Cisco.com

# IPv6 Basics

**Tony Hain**

**Cisco Systems**

**ahain@cisco.com**

# Outline

- Protocol Background
- Technology Differences
- Enhanced Capabilities
- Inaccuracies & Speculation
- Transition Technologies

# Protocol Background

# Why a New IP?

- 1991 – ALE WG studied projections about address consumption rate showed exhaustion by 2008.

- Bake-off in mid-1994 selected approach of a new protocol over multiple layers of encapsulation.

# What Ever Happened to IPv5?

| | | | |
|---|---|---|---|
| 0 | IP | March 1977 version | (deprecated) |
| 1 | IP | January 1978 version | (deprecated) |
| 2 | IP | February 1978 version A | (deprecated) |
| 3 | IP | February 1978 version B | (deprecated) |
| 4 | IPv4 | September 1981 version | (current widespread) |
| 5 | ST | Stream Transport | (not a new IP, little use) |
| 6 | IPv6 | December 1998 version | (formerly SIP, SIPP) |
| 7 | CATNIP | IPng evaluation | (formerly TP/IX; deprecated) |
| 8 | Pip | IPng evaluation | (deprecated) |
| 9 | TUBA | IPng evaluation | (deprecated) |
| 10-15 | | unassigned | |

# Do We Really Need a Larger Address Space?

- Internet Users or PC

  ~530 million users in Q2 CY2002, ~945 million by 2004

  (Source: Computer Industry Almanac)

  Emerging population/geopolitical and Address space

- PDA, Pen-Tablet, Notepad,…

  ~20 million in 2004

- Mobile phones

  Already 1 billion mobile phones delivered by the industry

- Transportation

  1 billion automobiles forecast for 2008

  Internet access in Planes

- Consumer devices

  Billions of Home and Industrial Appliances

# Explosion of New Internet Appliances

# IP Address Allocation History

Cisco.com

1981 - IPv4 protocol published

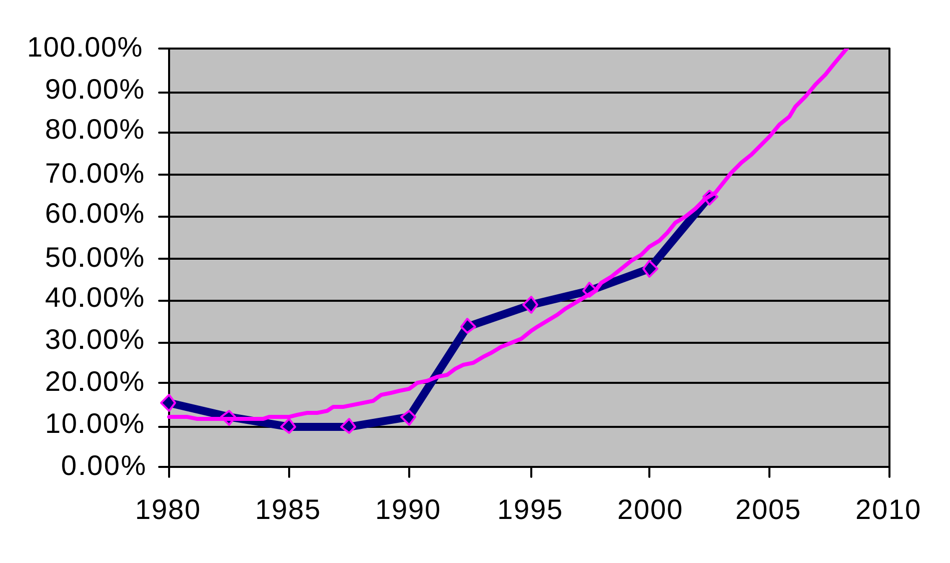
1985 ~ 1/16 of total space

1990 ~ 1/8 of total space

1995 ~ 1/3 of total space

2000 ~ 1/2 of total space

2002.5 ~ 2/3 of total space



- This despite increasingly intense conservation efforts

  PPP / DHCP address sharing

  CIDR (classless inter-domain routing)

  NAT (network address translation)

  plus some address reclamation

- Theoretical limit of 32-bit space: ~4 billion devices
  Practical limit of 32-bit space: ~250 million devices (RFC 3194)

# What were the goals of a new IP design?

- Expectation of a resurgence of “always-on” technologies

  xDSL, cable, Ethernet-to-the-home, Cell-phones, etc.

- Expectation of new users with multiple devices.

  China, India, etc. as new growth

  Consumer appliances as network devices

  ($10^{15}$ endpoints)

- Expectation of millions of new networks.

  Expanded competition and structured delegation.

  ($10^{12}$ sites)

# Why was 128 bits chosen as the IPv6 address size?

## Proposals for fixed-length, 64-bit addresses

- Accommodates $10^{12}$ sites, $10^{15}$ nodes, at .0001 allocation efficiency (3 orders of mag. more than IPng requirement)
- Minimizes growth of per-packet header overhead
- Efficient for software processing on current CPU hardware

## Proposals for variable-length, up to 160 bits

- Compatible with deployed OSI NSAP addressing plans
- Accommodates auto-configuration using IEEE 802 addresses
- Sufficient structure for projected number of service providers

## Settled on fixed-length, 128-bit addresses

- (340,282,366,920,938,463,463,374,607,431,768,211,456 in all!)

# Benefits of 128 bit Addresses

- Room for many levels of structured hierarchy and routing aggregation
- Easier address management and delegation than IPv4
- Easy address auto-configuration
- Ability to deploy end-to-end IPsec
  (NATs removed as unnecessary)

# Incidental Benefits of New Deployment

- Chance to eliminate some complexity in IP header

  improve per-hop processing

- Chance to upgrade functionality

  multicast, QoS, mobility

- Chance to include new features

  binding updates

# IPv6 & Geo-Politics

- Japan

  Formal announcement of IPv6 in the “e-Japan Initiative” plan, 2000

  IPv6 Promotion council Tax incentive program, 2002-2003

- Korea

  Looking for advanced services (consumer VoIP) on wide scale DSL

- China

  Is establishing an IPv6 collaboration with Japan

- Europe

  European IPv6 Task Force, www.ipv6-taskforce.org

  IPv6 2005 roadmap recommendations – Jan. 2002

  European Commission IPv6 project funding: 6NET & EuroIX

- U.S.

  North-America IPv6 Task Force

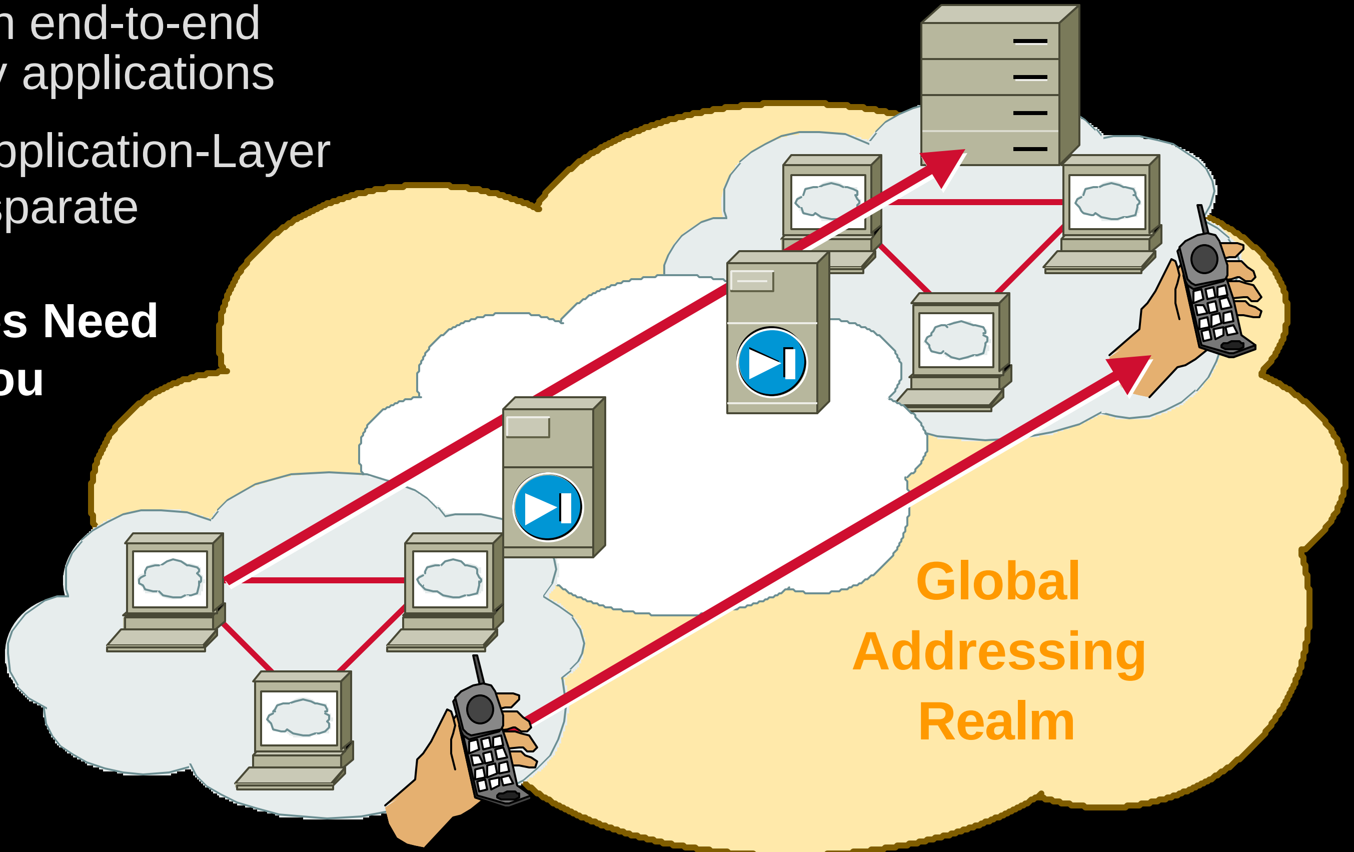
# Coming Back to an End-to-End Architecture

**New Technologies/Applications for Home Users**

**'Always-on'—Cable, DSL, Ethernet-to-the-home, Wireless,…**

- Internet started with end-to-end connectivity for any applications
- Today, NAT and Application-Layer Gateways connect disparate networks
- **Always-on Devices Need an Address When You Call Them**, eg.

- Mobile Phones
- Gaming
- Residential Voice over IP gateway
- IP Fax



# IPv6 Technology Scope

| *IP Service* | *IPv4 Solution* | *IPv6 Solution* |
|---|---|---|
| Addressing Range | 32-bit, Network Address Translation | 128-bit, Multiple Scopes |
| Autoconfiguration | DHCP | Serverless, Reconfiguration, DHCP |
| Security | IPSec | IPSec Mandated, works End-to-End |
| Mobility | Mobile IP | Mobile IP with Direct Routing |
| Quality-of-Service | Differentiated Service, Integrated Service | Differentiated Service, Integrated Service |
| IP Multicast | IGMP/PIM/Multicast BGP | MLD/PIM/Multicast BGP,Scope Identifier |

# Summary of Main IPv6 Benefits

- Expanded addressing capabilities
- Structured hierarchy to manage routing table growth
- Serverless autoconfiguration and reconfiguration
- Streamlined header format and flow identification
- Improved support for options / extensions

# IPv6 Advanced Features

- Security - Built-in, strong IP-layer encryption and authentication
- Mobility - More efficient and robust mechanisms
- Quality of Service
- Privacy Extensions for Stateless Address Autoconfiguration (RFC 3041)
- Source address selection

# How to get an IPv6 allocation?

- IPv6 address space at ARIN

  http://www.arin.net/registration/ipv6/index.html

- Allocation policies

  http://www.arin.net/policy/ipv6_policy.html

The minimum registry allocation for IPv6 is ::/32

The resulting customer allocations are expected to be:

::/48 in the general case, except for very large subscribers

::/64 when it is known that only one subnet is needed by design

::/128 when it is absolutely known that only one device is connecting without privacy expectations

# IPv6 Prefix Allocations: APNIC

## (whois.apnic.net) – Sept 2002

| | |
|---|---|
| WIDE-JP-19990813 | 2001:0200::/35 |
| NUS-SG-19990827 | 2001:0208::/35 |
| CONNECT-AU-19990916 | 2001:0210::/35 |
| NTT-JP-19990922 | 2001:0218::/32 |
| KT-KR-19991006 | 2001:0220::/35 |
| JENS-JP-19991027 | 2001:0228::/35 |
| ETRI-KRNIC-KR-19991124 | 2001:0230::/32 |
| HINET-TW-20000208 | 2001:0238::/32 |
| IIJ-JPNIC-JP-20000308 | 2001:0240::/32 |
| IMNET-JPNIC-JP-20000314 | 2001:0248::/35 |
| CERNET-CN-20000426 | 2001:0250::/32 |
| INFOWEB-JPNIC-JP-2000502 | 2001:0258::/35 |
| BIGLOBE-JPNIC-JP-20000719 | 2001:0260::/35 |
| BIGLOBE-JPNIC-JP-20000719 | 2001:0260::/32 |
| DION6-JPNIC-JP-20000829 | 2001:0268::/32 |
| DACOM-BORANET-20000908 | 2001:0270::/35 |
| ODN-JPNIC-JP-20000915 | 2001:0278::/32 |
| KOLNET-KRNIC-KR-20000927 | 2001:0280::/32 |
| TANET-TWNIC-TW-20001006 | 2001:0288::/35 |
| HANANET-KRNIC-KR-20001030 | 2001:0290::/32 |
| SONYTELECOM-JPNIC-JP-20001207 | 2001:0298::/32 |
| POWEREDCOM-JPNIC-JP-20001208 | 2001:02A0::/35 |
| CCCN-JPNIC-JP-20001228 | 2001:02A8::/35 |
| KORNET-KRNIC-KR-20010102 | 2001:02B0::/35 |
| NGINET-KRNIC-KR-20010115 | 2001:02B8::/32 |
| INFOSPHERE-JPNIC-JP-20010208 | 2001:02C0::/32 |
| OMP-JPNIC-JP-20010208 | 2001:02C8::/35 |
| ZAMA-AP-20010320 | 2001:02D0::/35 |
| SKTELECOMNET-KRNIC-KR-20010406 | 2001:02D8::/32 |
| HKNET-HK-20010420 | 2001:02E0::/35 |
| DTI-JPNIC-JP-20010702 | 2001:02E8::/32 |
| MEX-JPNIC-JP-20010801 | 2001:02F0::/32 |
| SINET-JPNIC-JP-20010809 | 2001:02F8::/32 |
| PANANET-JPNIC-JP-20010810 | 2001:0300::/35 |
| HTCN-JPNIC-JP-20010814 | 2001:0308::/32 |
| CWIDC-JPNIC-JP-20010815 | 2001:0310::/32 |
| STCN-JPNIC-JP-20010817 | 2001:0318::/32 |
| KREONET2-KRNIC-KR-20010823 | 2001:0320::/32 |
| MANIS-MY-20010824 | 2001:0328::/32 |
| SAMSUNGNETWORKS-KRNIC-KR-20010920 | 2001:0330::/32 |
| U-NETSURF-JPNIC-JP-20011005 | 2001:0338::/35 |
| FINE-JPNIC-JP-20011030 | 2001:0340::/35 |
| QCN-JPNIC-JP-20011031 | 2001:0348::/32 |
| MCNET-JPNIC-JP-20011108 | 2001:0350::/35 |
| MIND-JPNIC-JP-20011115 | 2001:0358::/35 |
| V6TELSTRAINTERNET-AU-20011211 | 2001:0360::/32 |
| MEDIAS-JPNIC-JP-20011212 | 2001:0368::/32 |
| GCTRJP-NET-20011212 | 2001:0370::/35 |
| THRUNET-KRNIC-KR-20011218 | 2001:0378::/35 |
| OCN-JP-JPNIC-JP-20020115 | 2001:0380::/32 |
| AARNET-IPV6-20020117 | 2001:0388::/32 |
| HANINTERNET-KRNIC-KR-20020207 | 2001:0390::/32 |
| HOTNET-JPNIC-JP-20020215 | 2001:0398::/32 |
| MULTIFEED-JPNIC-JP-20020319 | 2001:03A0::/35 |
| GNGIDC-KRNIC-KR-20020402 | 2001:03A8::/32 |
| KMN-IPV6-JPNIC-JP-20020403 | 2001:03B0::/32 |
| SO-NET-JPNIC-JP-20020409 | 2001:03B8::/35 |
| TOCN-20020513 | 2001:03C0::/35 |
| UNINET-TH-20020513 | 2001:03C8::/35 |
| PTOP-JPNIC-JP-20020521 | 2001:03D0::/35 |
| XEPHION-JPNIC-JP-20020523 | 2001:03D8::/32 |
| FBDC-JPNIC-JP-20020524 | 2001:03E0::/32 |
| INTEROP-JP-20020617 | 2001:03E8::/35 |
| KCOM-V6-JPNIC-JP-20020704 | 2001:03F0::/32 |
| BIIV6-CN-20020704 | 2001:03F8::/32 |
| INET-TH-20020711 | 2001:0C00::/32 |
| ASNET-TWNIC-TW-20020711 | 2001:0C08::/32 |
| SINGTEL-IXV6-20020718 | 2001:0C10::/32 |
| ARCNET6-20020723 | 2001:0C18::/32 |
| SINGNET-V6-SG-20020724 | 2001:0C20::/32 |
| ASAHI-NET-JPNIC-JP-20020730 | 2001:0C28::/32 |
| JCNET-JPNIC-JP-20020801 | 2001:0C30::/32 |
| CATIPV6-20020707 | 2001:0C38::/32 |
| GCIX-JPNIC-JP-20020808 | 2001:0C40::/32 |
| DREAMX-KRNIC-KR-20020812 | 2001:0C48::/32 |
| TTN-TWNIC-TW-20020812 | 2001:0C50::/32 |
| SIXREN-TWNIC-TW-20020827 | 2001:0C58::/32 |
| TIARE-PG-20020827 | 2001:0C60::/32 |
| Chinanet | 2001:0C68::/32 |
| CWJ-JPNIC-JP-20020910 | 2001:0C70::/32 |
| NTTIP-AU-20020910 | 2001:0C78::/32 |
| Allocated Prefixes: | 81 |

# IPv6 Prefix Allocations: RIPE-NCC
## (whois.ripe.net) – Sept 2002

| | |
|---|---|
| EU-UUNET-19990810 | 2001:0600::/35 |
| DE-SPACE-19990812 | 2001:0608::/32 |
| NL-SURFNET-19990819 | 2001:0610::/32 |
| UK-BT-19990903 | 2001:0618::/32 |
| CH-SWITCH-19990903 | 2001:0620::/35 |
| AT-ACONET-19990920 | 2001:0628::/35 |
| UK-JANET-19991019 | 2001:0630::/32 |
| DE-DFN-19991102 | 2001:0638::/35 |
| RU-FREENET-19991115 | 2001:0640::/35 |
| GR-GRNET-19991208 | 2001:0648::/35 |
| DE-ECRC-19991223 | 2001:0650::/32 |
| DE-TRMD-20000317 | 2001:0658::/32 |
| FR-RENATER-20000321 | 2001:0660::/35 |
| EU-NACNET-20000403 | 2001:0668::/35 |
| EU-EUNET-20000403 | 2001:0670::/35 |
| DE-JIPPII-20000426 | 2001:0678::/35 |
| DE-XLINK-20000510 | 2001:0680::/35 |
| FR-TELECOM-20000623 | 2001:0688::/32 |
| PT-RCCN-20000623 | 2001:0690::/32 |
| SE-SWIPNET-20000828 | 2001:0698::/35 |
| PL-ICM-20000905 | 2001:06A0::/35 |
| BE-BELNET-20001101 | 2001:06A8::/35 |
| SE-SUNET-20001218 | 2001:06B0::/32 |
| IT-CSELT-20001221 | 2001:06B8::/32 |
| SE-TELIANET-20010102 | 2001:06C0::/35 |
| DK-TELEDANMARK-20010131 | 2001:06C8::/35 |
| RU-ROSNIIROS-20010219 | 2001:06D0::/35 |
| PL-CYFRONET-20010221 | 2001:06D8::/35 |
| NL-INTOUCH-20010307 | 2001:06E0::/35 |
| FI-TELIVO-20010321 | 2001:06E8::/32 |
| SE-DIGITAL-20010321 | 2001:06F0::/35 |
| UK-EASYNET-20010322 | 2001:06F8::/32 |
| NO-UNINETT-20010406 | 2001:0700::/32 |
| FI-FUNET-20010503 | 2001:0708::/35 |
| UK-INS-20010518 | 2001:0710::/35 |
| CZ-TEN-34-20010521 | 2001:0718::/35 |
| ES-REDIRIS-20010521 | 2001:0720::/32 |
| UK-VERIO-20010717 | 2001:0728::/32 |
| AT-TELEKABEL-20010717 | 2001:0730::/32 |
| HU-HUNGARNET-20010717 | 2001:0738::/32 |
| DE-VIAG-20010717 | 2001:0740::/32 |
| DE-ROKA-20010817 | 2001:0748::/35 |
| IT-EDISONTEL-20010906 | 2001:0750::/35 |
| UK-NETKONECT-20010918 | 2001:0758::/35 |
| IT-GARR-20011004 | 2001:0760::/35 |
| DE-CYBERNET-20011008 | 2001:0768::/32 |
| IE-HEANET-20011008 | 2001:0770::/35 |
| LT-LITNET-20011115 | 2001:0778::/35 |
| DE-NORIS-20011203 | 2001:0780::/35 |
| FI-SONERA-20011231 | 2001:0788::/32 |
| EU-CARRIER1-20020102 | 2001:0790::/35 |
| EU-DANTE-20020131 | 2001:0798::/32 |
| DE-TELEKOM-20020228 | 2001:07A0::/35 |
| FR-NERIM-20020313 | 2001:07A8::/35 |
| DE-COMPLETEL-20020313 | 2001:07B0::/35 |
| NL-BIT-20020405 | 2001:07B8::/32 |
| DE-BELWUE-20020411 | 2001:07C0::/32 |
| IE-ISI-20020515 | 2001:07C8::/35 |
| EE-ESTPAK-20020516 | 2001:07D0::/35 |
| FI-KOLUMBUS-20020528 | 2001:07D8::/32 |
| UK-OPALNET-20020530 | 2001:07E0::/32 |
| LU-PT-20020605 | 2001:07E8::/35 |
| EU-LAMBDANET-20020605 | 2001:07F0::/35 |
| ES-TTD-20020705 | 2001:0800::/32 |
| PL-POZMAN-20020710 | 2001:0808::/32 |
| FR-SDV-20020710 | 2001:0810::/32 |
| PT-TELECEL-20020711 | 2001:0818::/32 |
| NO-WEBONLINE-20020712 | 2001:0820::/32 |
| NL-PROSERVE-20020712 | 2001:0828::/32 |
| DE-MAINLAB-20020724 | 2001:0830::/32 |
| NL-CONCEPTS-20020724 | 2001:0838::/32 |
| NO-POWERTECH-20020725 | 2001:0840::/32 |
| IT-CSP-20020725 | 2001:0848::/32 |
| AT-ATNET-20020725 | 2001:0850::/32 |
| AT-SIL-20020725 | 2001:0858::/32 |
| FR-GROLIER-20020725 | 2001:0860::/32 |
| DE-IPHH-20020725 | 2001:0868::/32 |
| AT-EUROPEANTELECOM-20020725 | 2001:0870::/32 |
| DK-DENET-20020801 | 2001:0878::/32 |
| DE-KOMPLEX-20020801 | 2001:0880::/32 |
| NL-XS4ALL-20020807 | 2001:0888::/32 |
| AT-TELEKOM-20020812 | 2001:0890::/32 |
| NL-WIDEXS-20020812 | 2001:0898::/32 |
| PT-TELEPAC-20020814 | 2001:08A0::/32 |
| CH-CYBERLINK-20020816 | 2001:08A8::/32 |
| UK-AA-20020820 | 2001:08B0::/32 |
| FI-RSLCOM-20020822 | 2001:08B8::/32 |
| NO-CATCHIP-20020823 | 2001:08C0::/32 |
| YU-VERAT-20020829 | 2001:08C8::/32 |
| DE-CELOX-20020829 | 2001:08D0::/32 |
| DE-SCHLUND-20020910 | 2001:08D8::/32 |
| Allocated Prefixes: | 91 |

# IPv6 Prefix Allocations: ARIN (whois.arin.net) – Sept 2002

| | |
|---|---|
| ESNET-V6 | 2001:0400::/32 |
| VBNS-IPV6 | 2001:0408::/32 |
| CANET3-IPV6 | 2001:0410::/32 |
| VRIO-IPV6-0 | 2001:0418::/32 |
| CISCO-IPV6-1 | 2001:0420::/32 |
| QWEST-IPV6-1 | 2001:0428::/32 |
| DISN-LES-V6 | 2001:0430::/35 |
| ABOVENET-IPV6 | 2001:0438::/35 |
| SPRINT-V6 | 2001:0440::/32 |
| UNAM-IPV6 | 2001:0448::/32 |
| GBLX-V6 | 2001:0450::/35 |
| STEALTH-IPV6-1 | 2001:0458::/35 |
| NET-CW-10BLK | 2001:0460::/35 |
| ABILENE-IPV6 | 2001:0468::/32 |
| HURRICANE-IPV6 | 2001:0470::/32 |
| EP-NET | 2001:0478::/32 |
| DREN-V6 | 2001:0480::/35 |
| AVANTEL-IPV6-1 | 2001:0488::/35 |
| NOKIA-1 | 2001:0490::/35 |
| ITESM-IPV6 | 2001:0498::/32 |
| IPV6-RNP | 2001:04A0::/32 |
| AXTEL-IPV6-1 | 2001:04A8::/35 |
| AOLTIMEWARNER | 2001:04B0::/32 |
| WAYPORT-IPV6 | 2001:04B8::/32 |
| PROTEL-RED-1-V6 | 2001:04C0::/35 |
| UNINET-NETV6-1 | 2001:04C8::/35 |
| NASA-PCCA-V6 | 2001:04D0::/35 |
| DOTNET-001 | 2001:04D8::/35 |
| WISCNET-V6 | 2001:04E0::/32 |
| SHAWIPV6 | 2001:04E8::/32 |
| ENTERZONE-V6 | 2001:04F0::/32 |
| ISC6-1 | 2001:04F8::/32 |
| Allocated Prefixes: | 32 |

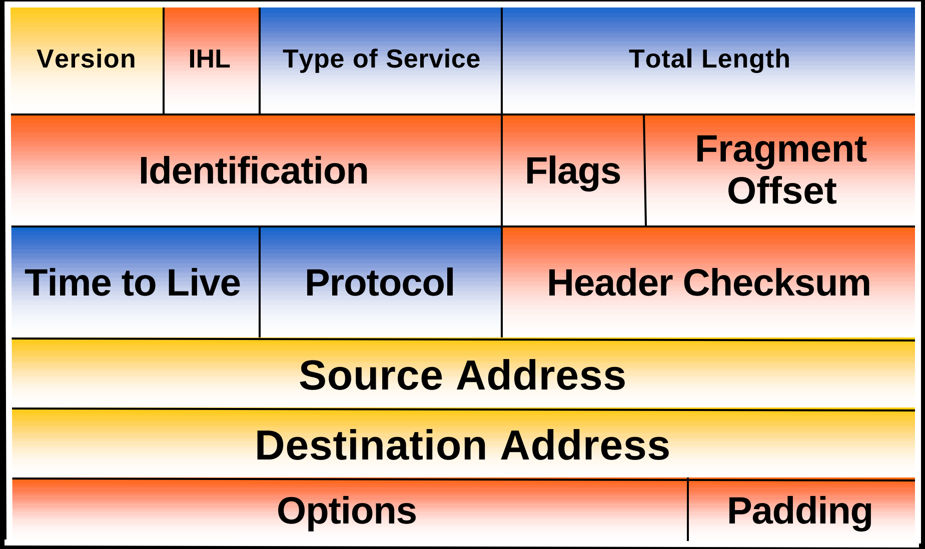
# Outline

- Protocol Background
- **Technology Differences**
- Enhanced Capabilities
- Inaccuracies & Speculation
- Transition Technologies

# A new Header

# IPv4 & IPv6 Header Comparison

## IPv4 Header



Version | IHL | Type of Service | Total Length

Identification | Flags | Fragment Offset

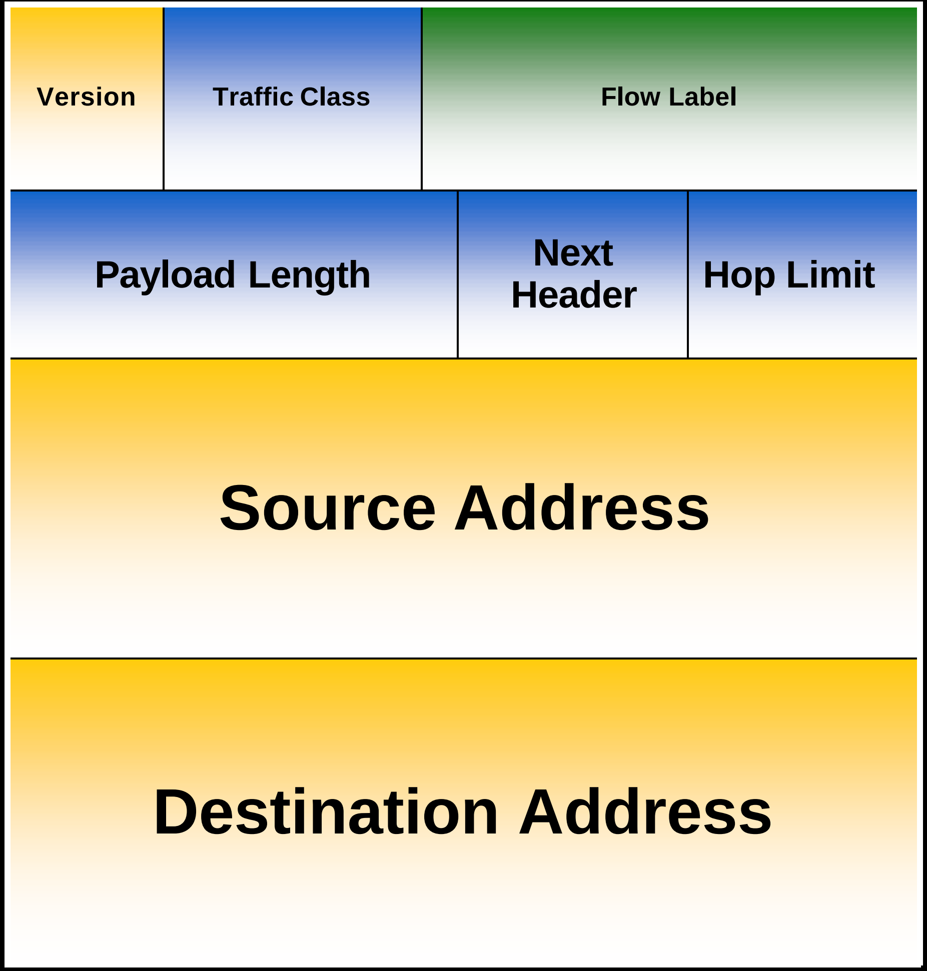
Time to Live | Protocol | Header Checksum

Source Address

Destination Address

Options | Padding

## IPv6 Header



Version | Traffic Class | Flow Label

Payload Length | Next Header | Hop Limit

Source Address

Destination Address

Legend

🟨 - field’s name kept from IPv4 to IPv6

🟧 - fields not kept in IPv6

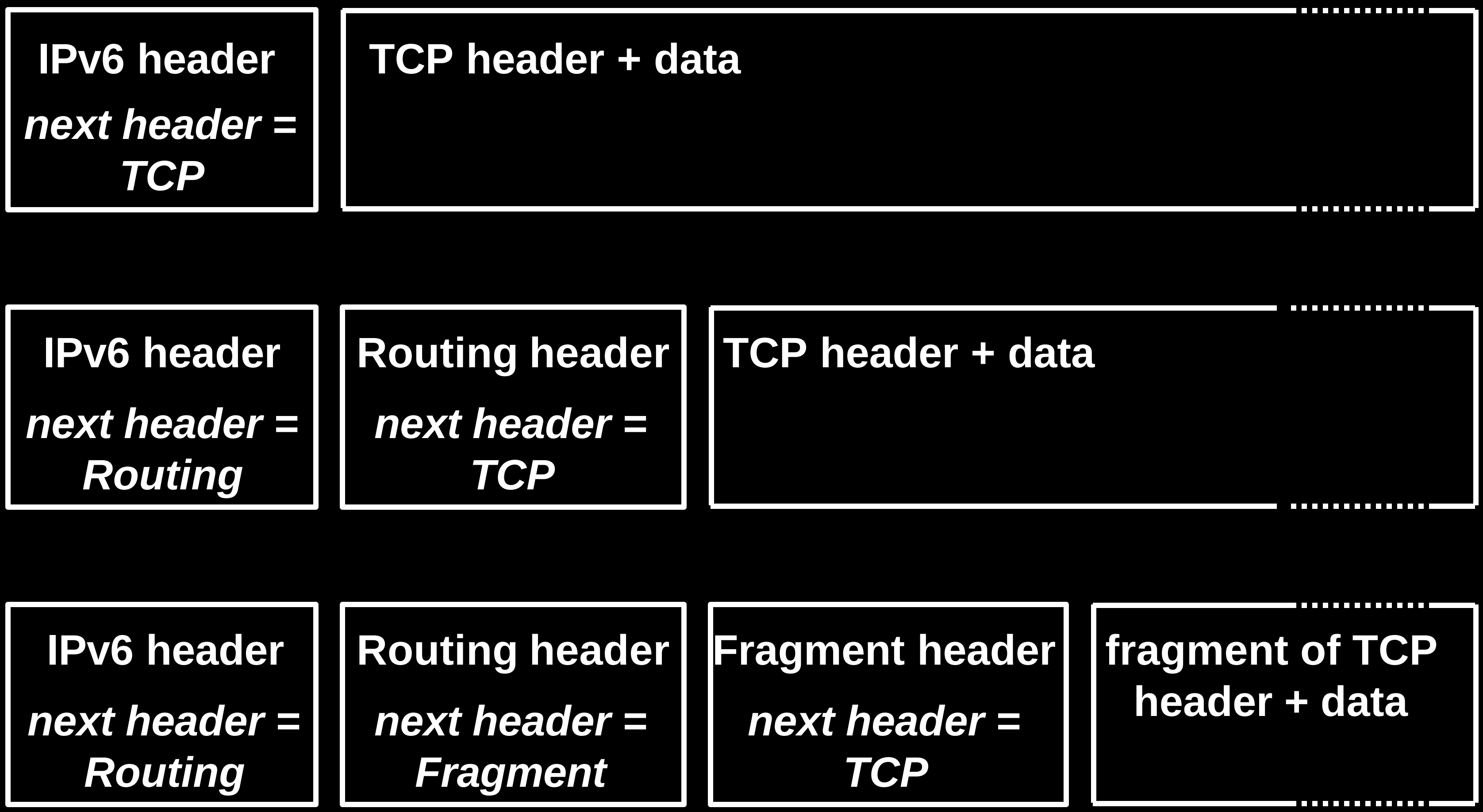
🟦 - Name & position changed in IPv6

🟩 - New field in IPv6

# Summary of Header Changes between IPv4 & IPv6

- **Streamlined**
  - Fragmentation fields moved out of base header
  - IP options moved out of base header
  - Header Checksum eliminated
  - Header Length field eliminated
  - Length field excludes IPv6 header
  - Alignment changed from 32 to 64 bits
- **Revised**
  - Time to Live 🏷 Hop Limit
  - Protocol 🏷 Next Header
  - Precedence & TOS 🏷 Traffic Class
  - Addresses increased 32 bits 🏷 128 bits
- **Extended**
  - Flow Label field added

# Extension Headers



IPv6 header

*next header = TCP*

TCP header + data

IPv6 header

*next header = Routing*

Routing header

*next header = TCP*

TCP header + data

IPv6 header

*next header = Routing*

Routing header

*next header = Fragment*

Fragment header

*next header = TCP*

fragment of TCP header + data

# Extension Headers (cont.)

- Generally processed only by node identified in IPv6 Destination Address field => much lower overhead than IPv4 options processing

  exception: Hop-by-Hop Options header

- Eliminated IPv4's 40-byte limit on options

  in IPv6, limit is total packet size, or Path MTU in some cases

- Currently defined extension headers:

  Fragment, Hop-by-Hop Options, Routing, Authentication, Encryption, Destination Options

# Fragment Header

| | | | |
|---|---|---|---|
| Next Header | Reserved | Fragment Offset | 0 0 M |
| Original Packet Identifier | | | |

- IPv6 fragmentation & reassembly is an end-to-end function;

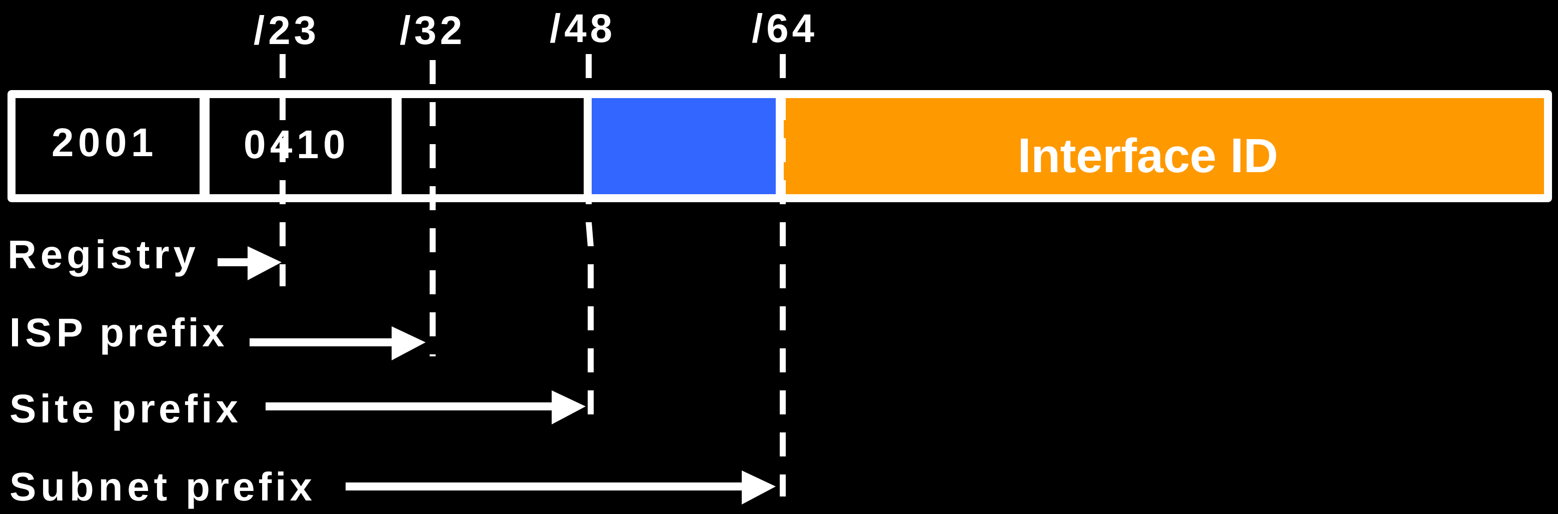
**routers do not fragment packets**

if packet is too big they send ICMP "packet too big"

- though discouraged, can use IPv6 Fragment header to support upper layers that do not (yet) do path MTU discovery

# Addressing

# Address Allocation



- The allocation process was recently updated by the registries:

  IANA allocates from 2001::/16 to regional registries

  Each regional registry allocation is a ::/23

  ISP allocations from the regional registry is a ::/36 (immediate allocation) or ::/32 (initial allocation) or shorter with justification

  Policy expectation that an ISP allocates a ::/48 prefix to each customer

# Some Terminology

| | |
|---|---|
| node | a protocol module that implements IPv6 |
| router | a node that forwards IPv6 packets not explicitly addressed to itself |
| host | any node that is not a router |
| link | a communication facility or medium over which nodes can communicate at the link layer,<br>i.e., the layer immediately below IPv6 |
| neighbors | nodes attached to the same link |
| interface | a node’s attachment to a link |
| address | an IPv6-layer identifier for an interface or a set of interfaces |

# Text Representation of Addresses

| | |
|---|---|
| "Preferred" form: | 1080:0:FF:0:8:800:200C:417A |
| Compressed form: | FF01:0:0:0:0:0:0:43<br>becomes FF01::43 |
| IPv4-mapped: | 0:0:0:0:0:FFFF:10.1.68.3<br>or ::FFFF:10.1.68.3 |

# IPv6 - Addressing Model

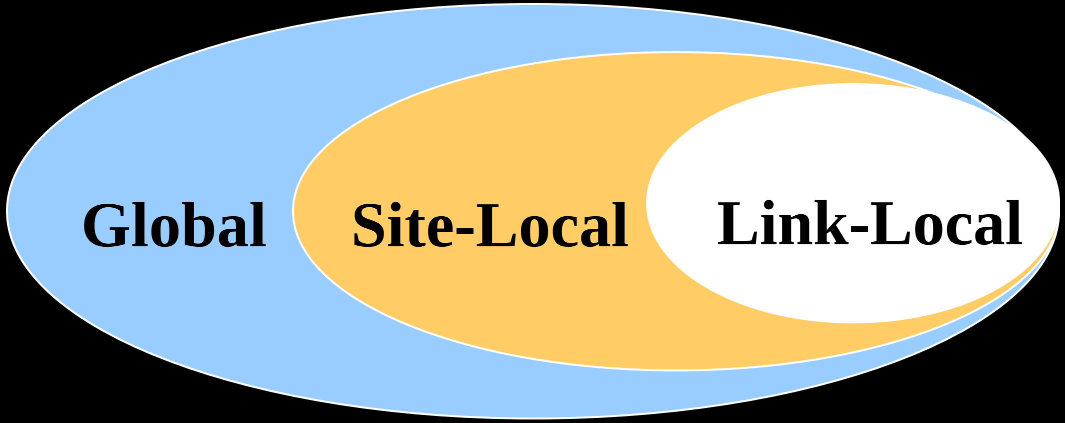
Addresses are assigned to interfaces

- change from IPv4 model :

Interface 'expected' to have multiple addresses

Addresses have scope

- Link Local
- Site Local
- Global



Addresses have lifetime

- Valid and Preferred lifetime

# Types of IPv6 Addresses

- **Unicast**

  **One address on a single interface**

  **Delivery to single interface**

- **Multicast**

  **Address of a set of interfaces**

  **Delivery to all interfaces in the set**

- **Anycast**

  **Address of a set of interfaces**

  **Delivery to a single interface in the set**

**No broadcast addresses**

# Interface Address set

- Loopback (only assigned to a single interface per node)
- Link local (required on all interfaces)
- Site local
- Auto-configured 6to4 (if IPv4 public is address available)
- Auto-configured IPv4 compatible (operationally discouraged)
- Solicited node Multicast (required for neighbor discovery)
- All node multicast
- Global anonymous
- Global published

# Source Address Selection Rules

- **Rule 1:** Prefer same address
- **Rule 2:** Prefer appropriate scope
  - Smallest matching scope
- **Rule 3:** Avoid deprecated addresses
- **Rule 4:** Prefer home addresses
- **Rule 5:** Prefer outgoing interface
- **Rule 6:** Prefer matching label from policy table
  - Native IPv6 source > native IPv6 destination
  - 6to4 source > 6to4 destination
  - IPv4-compatible source > IPv4-compatible destination
  - IPv4-mapped source> IPv4-mapped destination
- **Rule 7:** Prefer temporary addresses
- **Rule 8:** Use longest matching prefix

Local policy may override

# Destination Address Selection Rules

- **Rule 1:** Avoid unusable destinations
- **Rule 2:** Prefer matching scope
- **Rule 3:** Avoid dst with matching deprecated src address
- **Rule 4:** Prefer home addresses
- **Rule 5:** Prefer matching label from policy table
  - Native IPv6 source > native IPv6 destination
  - 6to4 source > 6to4 destination
  - IPv4-compatible source > IPv4-compatible destination
  - IPv4-mapped source> IPv4-mapped destination
- **Rule 6:** Prefer higher precedence
- **Rule 7:** Prefer smaller scope
- **Rule 8:** Use longest matching prefix
- **Rule 9:** Order returned by DNS

Local policy may override

# Address Type Prefixes

| Address type | Binary prefix |
|---|---|
| IPv4-compatible | 0000...0 (96 zero bits) |
| global unicast | 001 |
| link-local unicast | 1111 1110 10 |
| site-local unicast | 1111 1110 11 |
| multicast | 1111 1111 |

- all other prefixes reserved (approx. 7/8ths of total)
- anycast addresses use unicast prefixes

# Unicast Address Formats

## Link Local

| FP (10bits) | RESERVED (54bits) | Interface ID (64bits) |
|---|---|---|
| 1111111010 | MUST be 0 | MAC derived |

## Site Local

| FP (10bits) | Subnet (38bits) | Subnet (16bits) | Interface ID (64bits) |
|---|---|---|---|
| 1111111011 | Locally Administered | Locally Administered | MAC derived or Locally Administered |

## Global

| FP (3bits) | Registry / provider assigned (45bits) | Subnet (16bits) | Interface ID (64bits) |
|---|---|---|---|
| 001 | Provider Administered | Locally Administered | MAC derived or Locally Administered or Random |

# Tunneling Unicast Address Formats

## Compatible

| FP (96bits) | IPv4 ID (32bits) |
|---|---|
| MUST be 0 | Locally administered |

## 6to4

| FP (16bits) | IPv4 (32bits) | SLA (16bits) | Interface ID (64bits) |
|---|---|---|---|
| 00100010 | Provider Administered | Locally Administered | MAC derived or Locally Administered or Random |

## ISATAP

| Any (48bits) | SLA (16bits) | Interface ID (64bits) |
|---|---|---|
| Provider Administered | Locally Administered | IPv4 derived |

# Some Special-Purpose Unicast Addresses

- The unspecified address, used as a placeholder when no address is available:

  0:0:0:0:0:0:0:0

- The loopback address, for sending packets to self:

  0:0:0:0:0:0:0:1

# Multicast Address Format

| FP (8bits) | Flags (4bits) | Scope (4bits) | RESERVED (80bits) | Group ID (32bits) |
|---|---|---|---|---|
| 11111111 | 000T | Lcl/Sit/Gbl | MUST be 0 | Locally administered |

- **flag field**

  low-order bit indicates permanent/transient group

  (three other flags reserved)

- **scope field:**

  1 - node local
  2 - link-local
  5 - site-local

  8 - organization-local
  B - community-local
  E - global

  (all other values reserved)

- **map IPv6 multicast addresses directly into low order 32 bits of the IEEE 802 MAC**

# Multicast Address Format Unicast-Prefix based

| FP (8bits) | Flags (4bits) | Scope (4bits) | reserved (8bits) | plen (8bits) | Network Prefix (64bits) | Group ID (32bits) |
|---|---|---|---|---|---|---|
| 11111111 | 00PT | Lcl/Sit/Gbl | MUST be 0 | Locally administered | Unicast prefix | Auto configured |

- P = 1 indicates a multicast address that is assigned based on the network prefix
- plen indicates the actual length of the network prefix
- Source-specific multicast addresses is accomplished by setting
  P = 1
  plen = 0
  network prefix = 0

draft-ietf-ipngwg-uni-based-mcast-01.txt

# IPv6 Routing

# IPv6 routing

IPv6 still uses the longest-prefix match routing algorithm.

- RIPv2, supports split-horizon with poisoned reverse (RFC 2080)
- OSPFv3 (RFC 2740)
- ISIS (draft-ietf-isis-ipv6-02)
- BGP4+ (RFC 2858 and RFC 2545)

# Outline

- Protocol Background
- Technology Differences
- **Enhanced Capabilities**
- Inaccuracies & Speculation
- Transition Technologies

Cisco.com

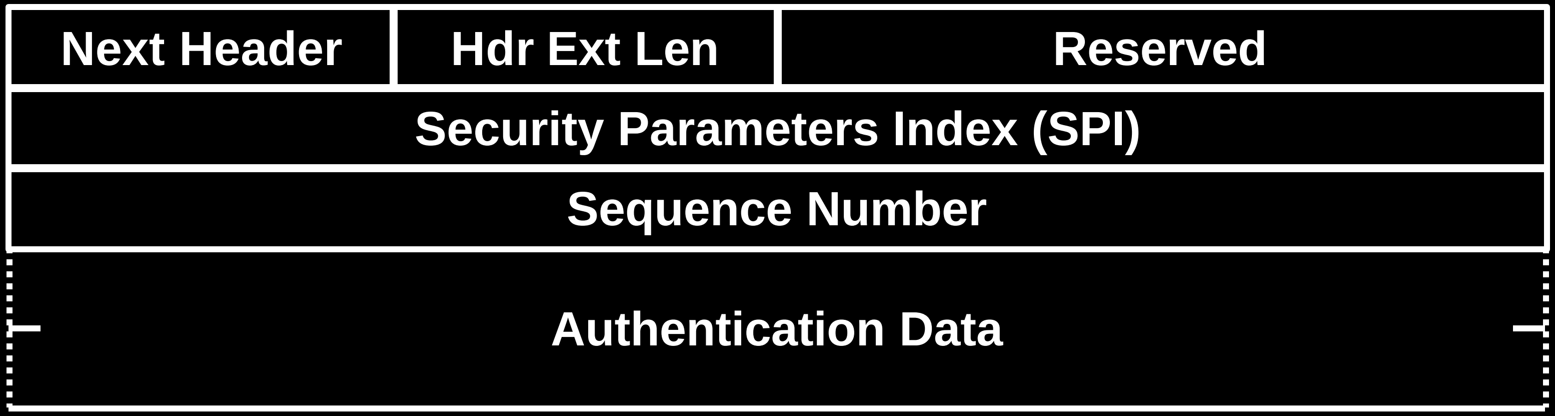
# Security

# IPv6 Security

Cisco.com

- All implementations required to support authentication and encryption headers (“IPsec”)
- Authentication separate from encryption for use in situations where encryption is prohibited or prohibitively expensive
- Key distribution protocols are under development (independent of IP v4/v6)
- Support for manual key configuration required

# Authentication Header



Next Header

Hdr Ext Len

Reserved

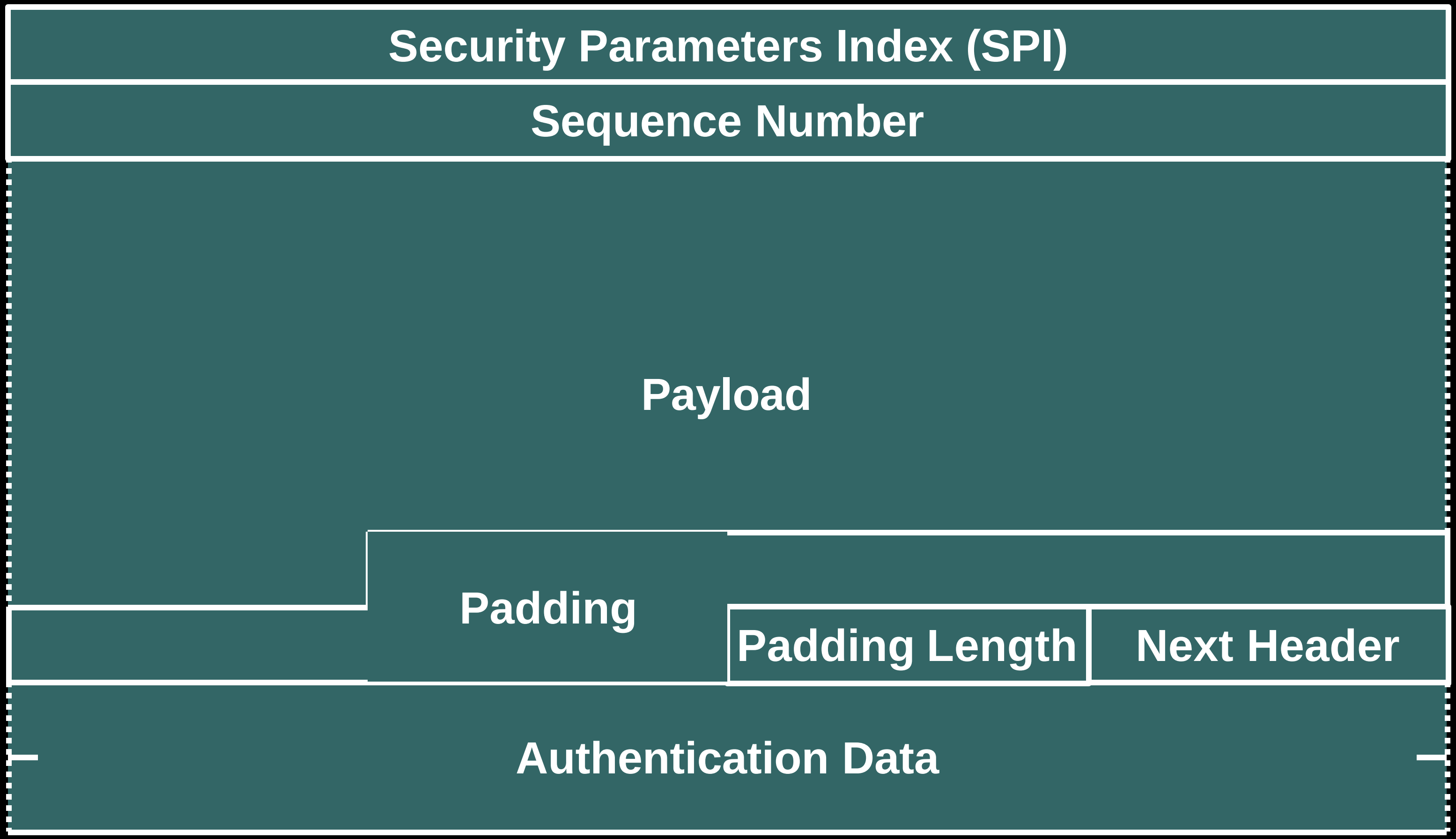
Security Parameters Index (SPI)

Sequence Number

Authentication Data

- Destination Address + SPI identifies security association state (key, lifetime, algorithm, etc.)
- Provides authentication and data integrity for all fields of IPv6 packet that do not change en-route
- Default algorithm is Keyed MD5

# Encapsulating Security Payload (ESP)



Security Parameters Index (SPI)

Sequence Number

Payload

Padding

Padding Length

Next Header

Authentication Data

# Quality of Service

# IP Quality of Service Approaches

Two basic approaches developed by IETF:

- “Differentiated Service” (diff-serv)

  coarse-grain (per-class), qualitative promises (e.g., higher priority), no explicit signaling

- “Integrated Service” (int-serv)

  fine-grain (per-flow), quantitative promises (e.g., x bits per second), uses RSVP signaling

# IPv6 Support for Diff-Serv

8-bit Traffic Class field to identify specific classes of packets needing special QoS

- same as dscp definition of IPv4 Type-of-Service byte
- may be initialized by source or by router enroute; may be rewritten by routers enroute
- traffic Class value of 0 used when no special QoS requested (the common case today)

# IPv6 Support for Int-Serv

20-bit Flow Label field to identify specific flows needing special QoS

- each source chooses its own Flow Label values; routers use Source Addr + Flow Label to identify distinct flows
- Flow Label value of 0 used when no special QoS requested (the common case today)
- this part of IPv6 is not standardized yet, and may well change semantics in the future

# Compromise

- **Signaled diff-serv (RFC 2998)**
  - **– uses RSVP for signaling with course-grained qualitative aggregate markings**
  - **– allows for policy control without requiring per-router state overhead**

# Mobility

# IPv6 Mobility

- Mobile hosts have one or more home address

  relatively stable; associated with host name in DNS

- A Host will acquire a care-of address when it discovers it is in a foreign subnet (i.e., not its home subnet)

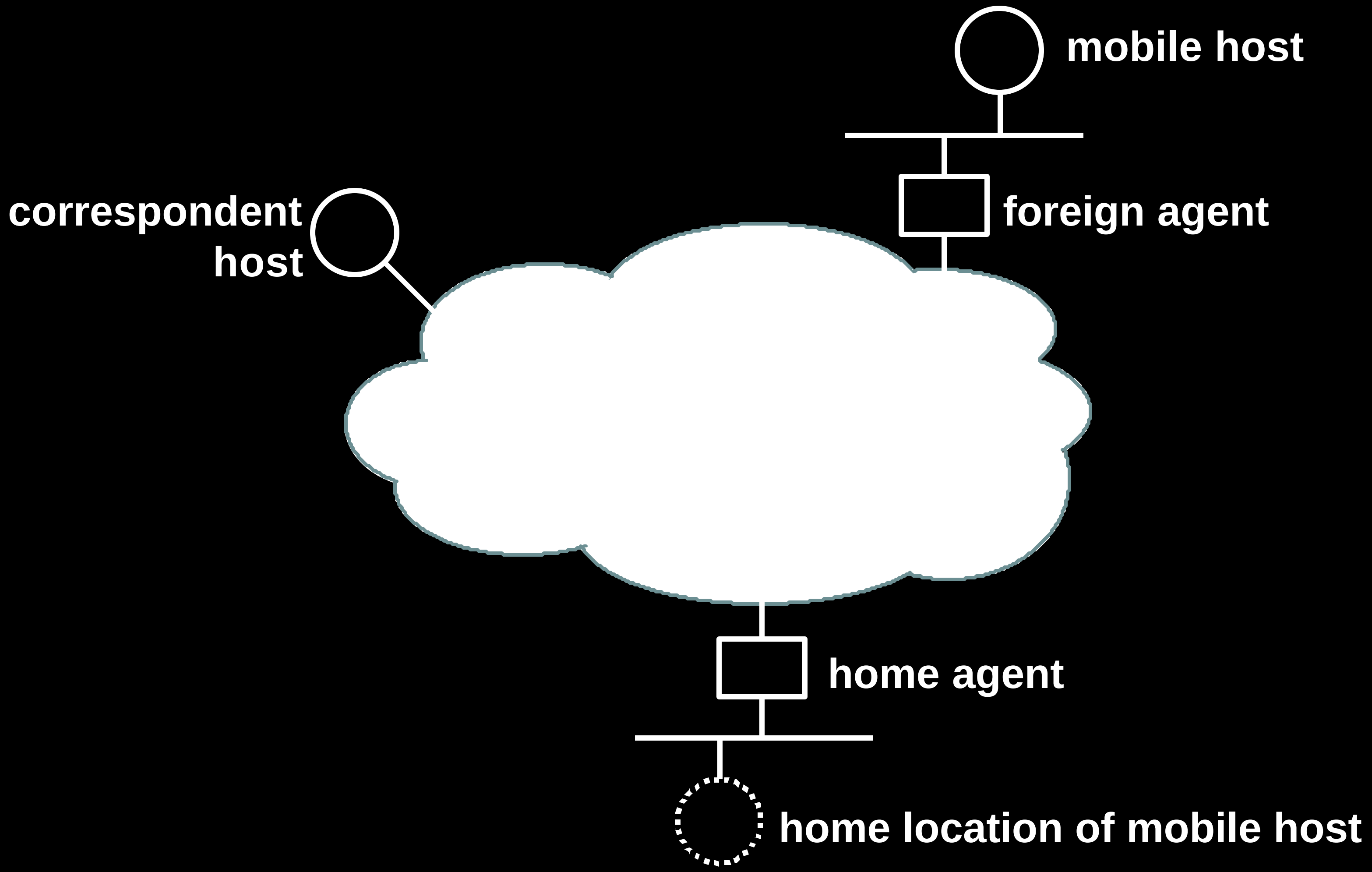
  uses auto-configuration or local policy to get the address

  registers the care-of address with a home agent,
  i.e, a router on its home subnet

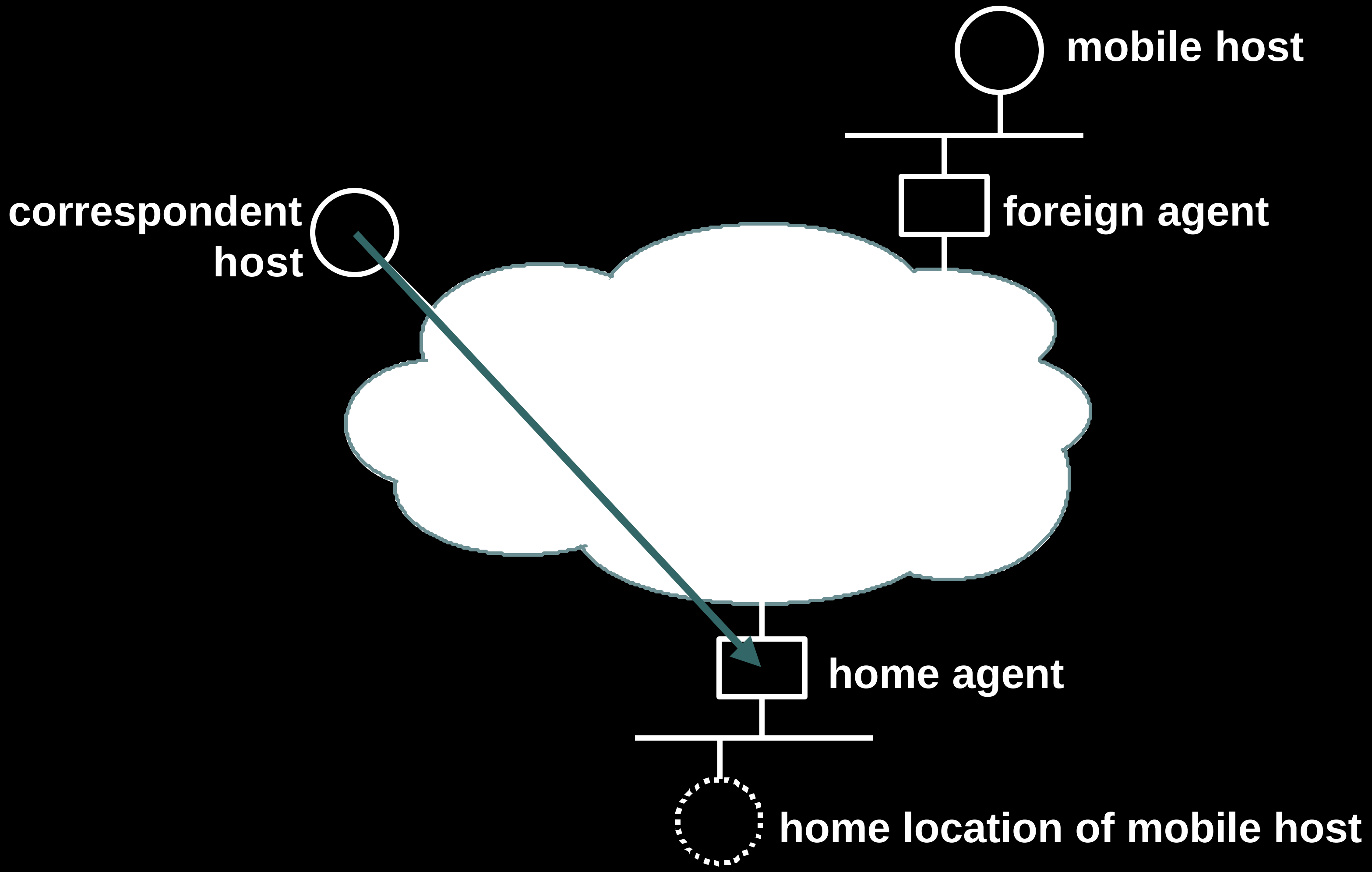
- Packets sent to the mobile's home address(es) are intercepted by home agent and forwarded to the care-of address, using encapsulation

- Mobile IPv6 hosts sends binding-updates to correspondent to remove home agent from flow

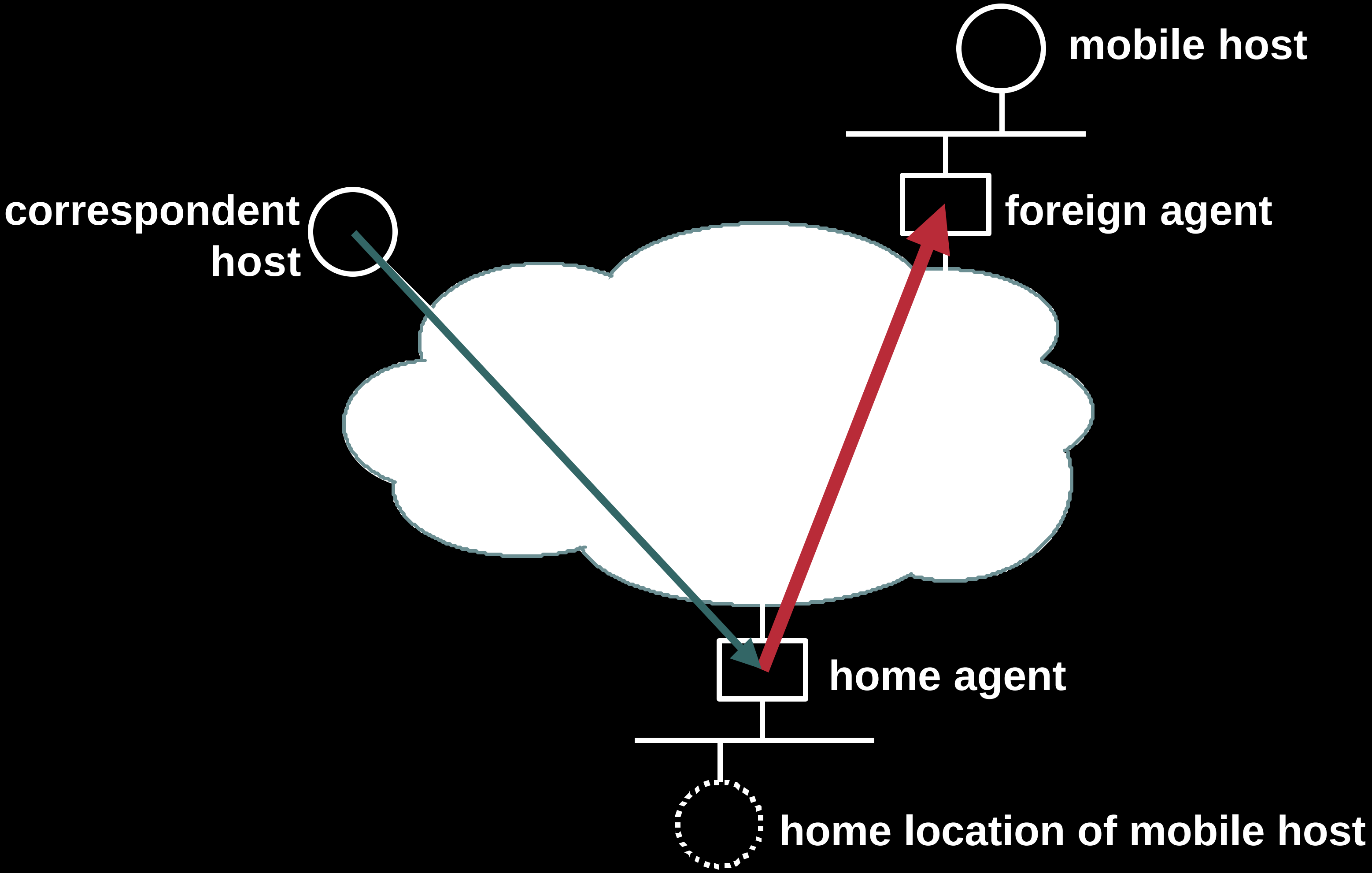
# Mobile IP (v4 version)



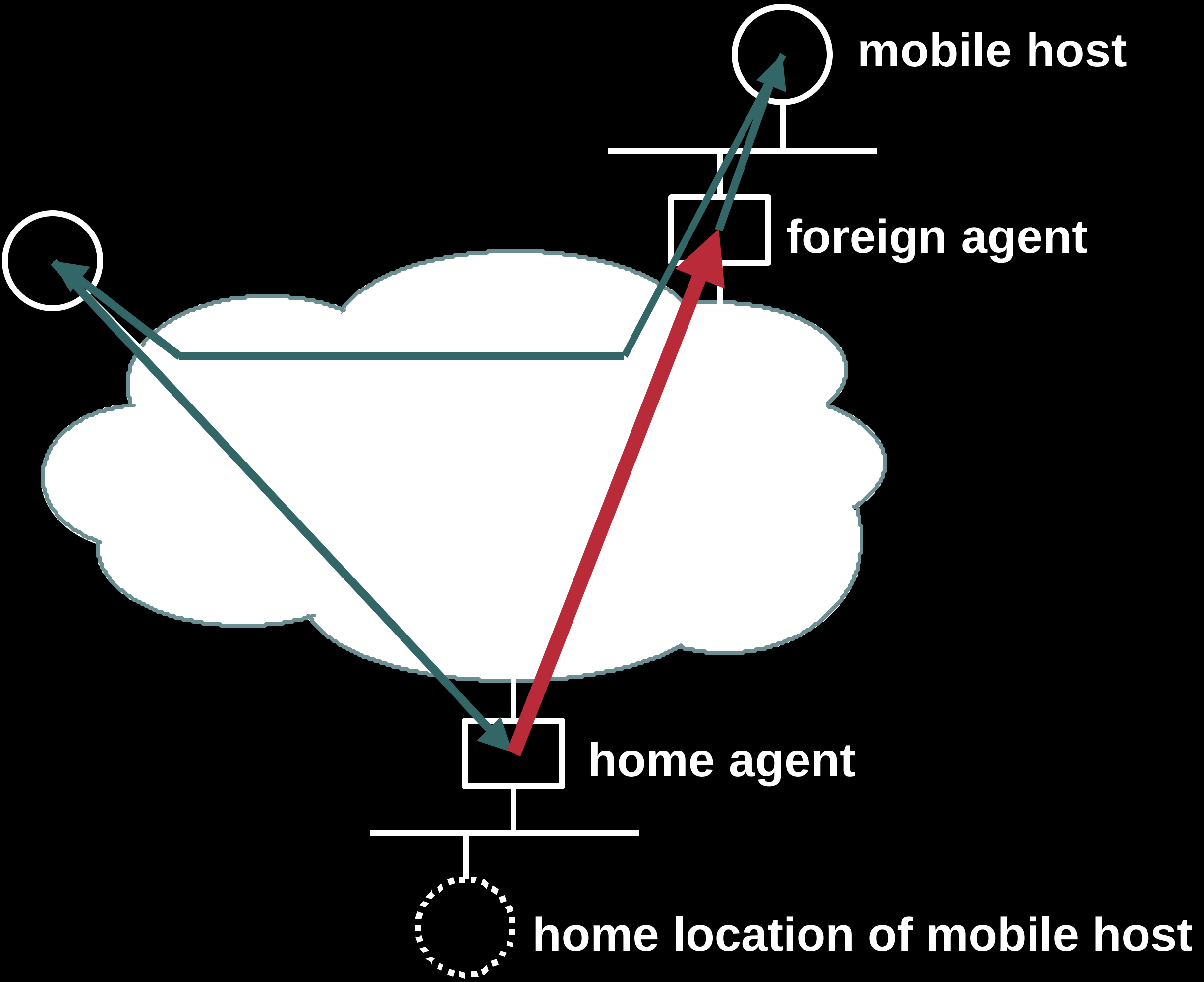
# Mobile IP (v4 version)



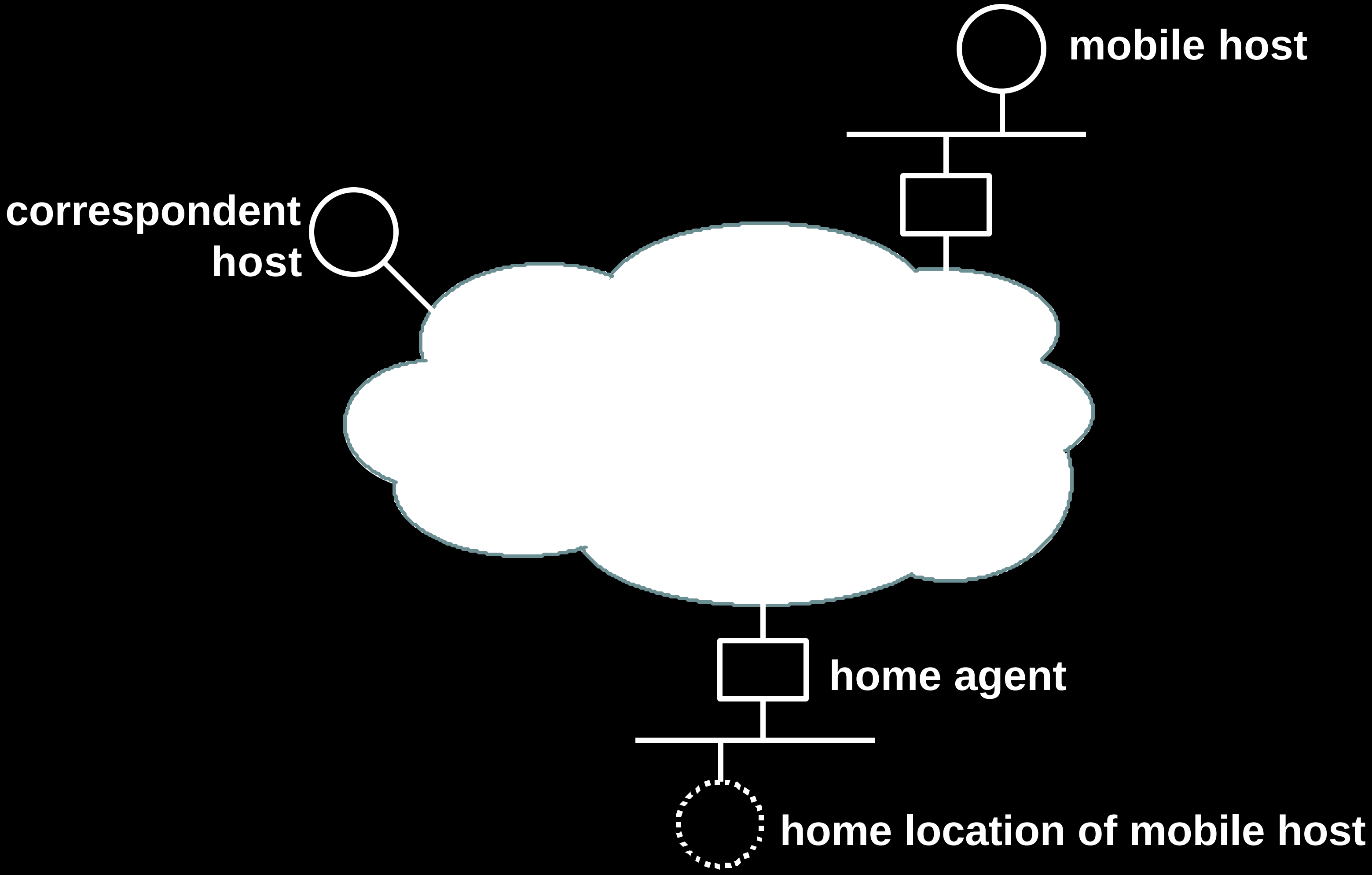
# Mobile IP (v4 version)



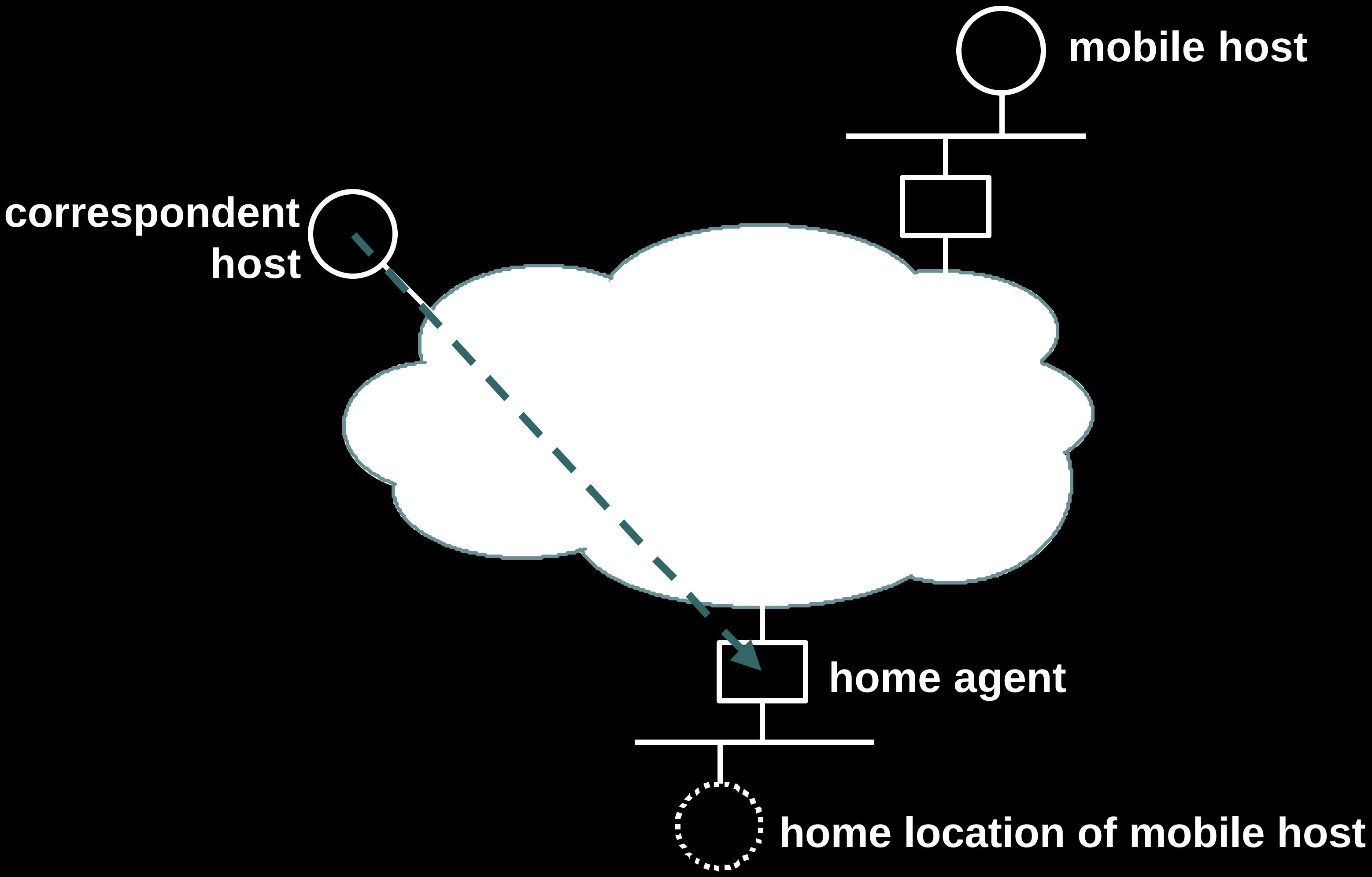
# Mobile IP (v4 version)



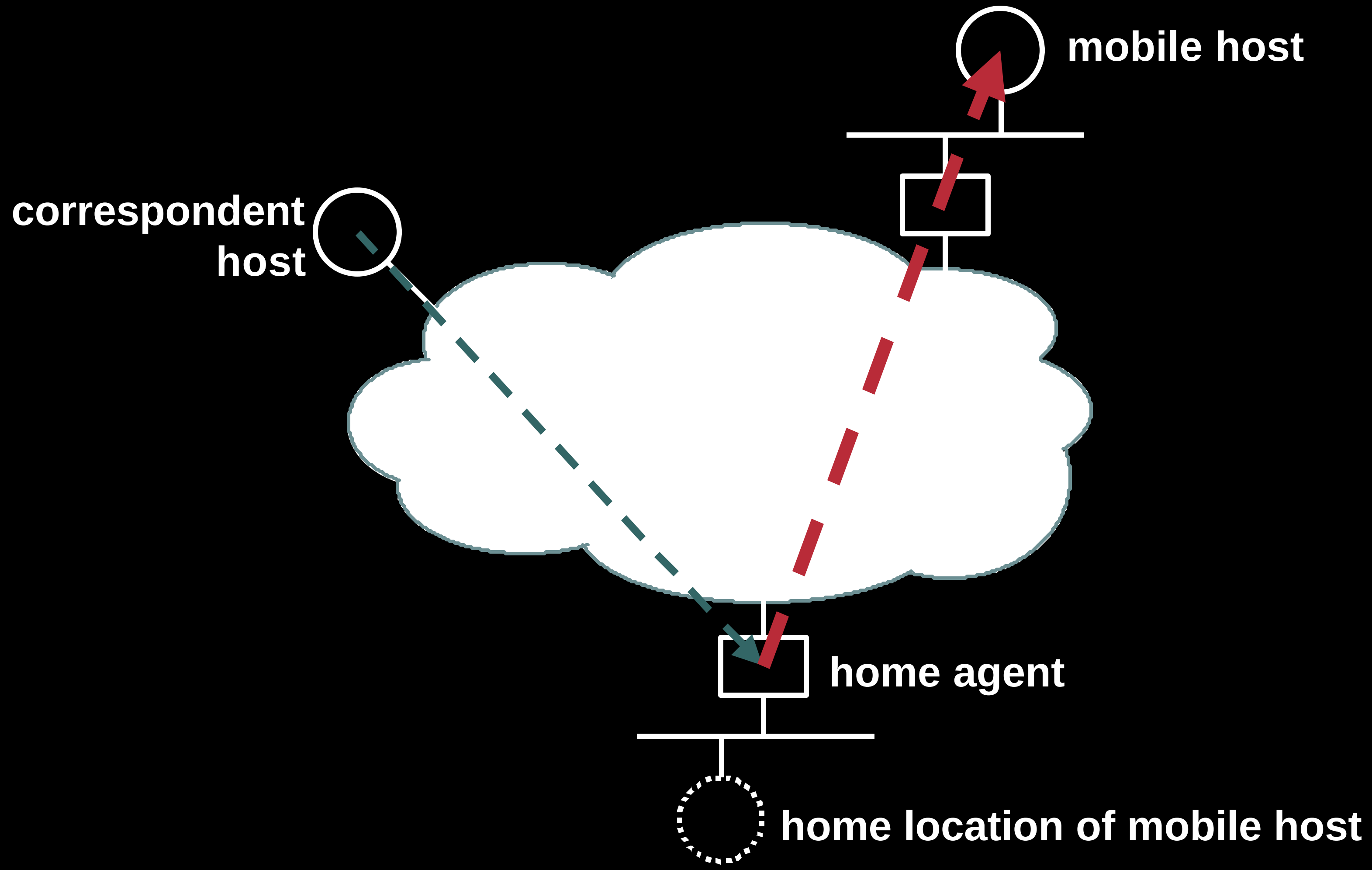
# Mobile IP (v6 version)



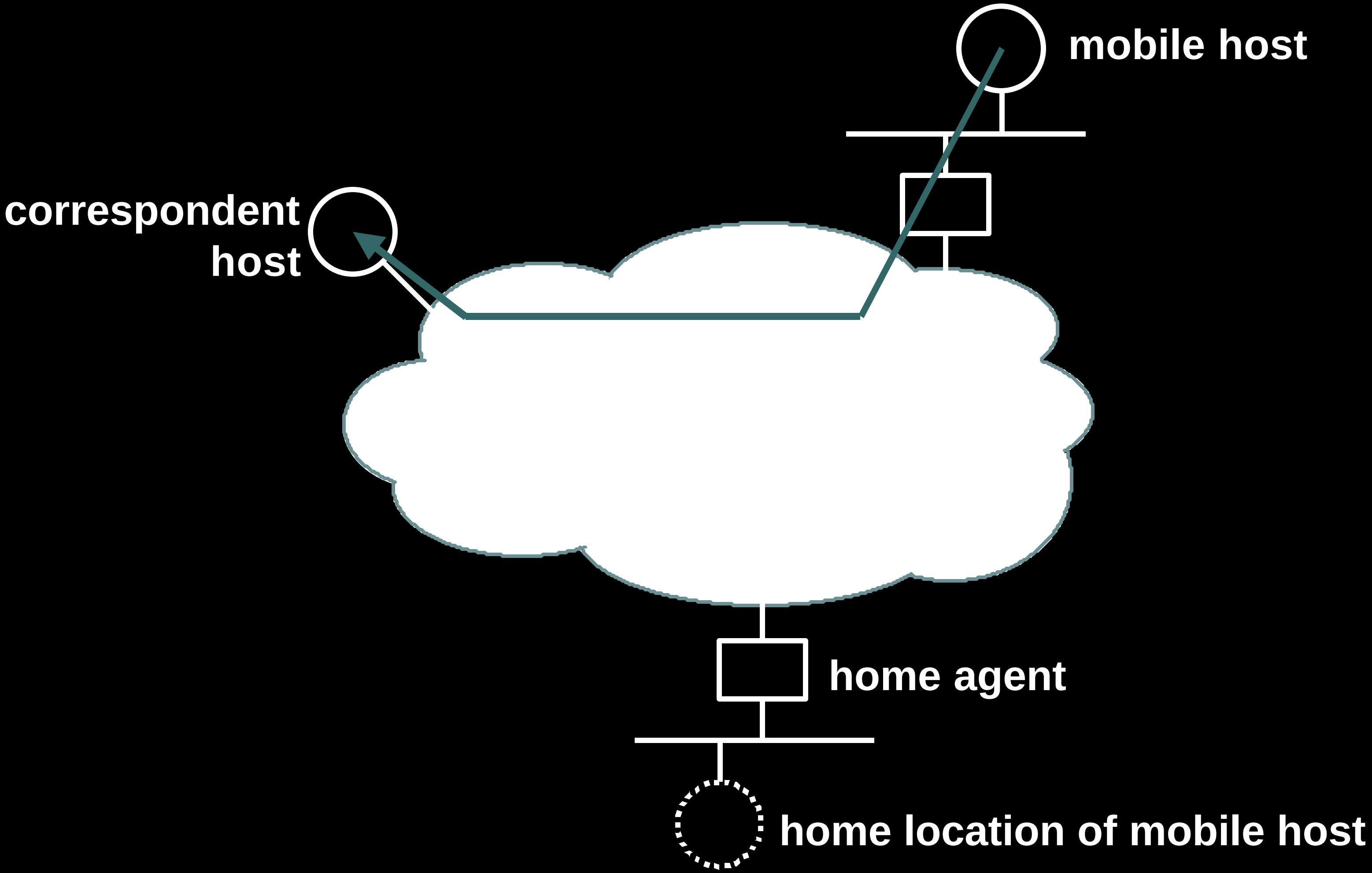
# Mobile IP (v6 version)



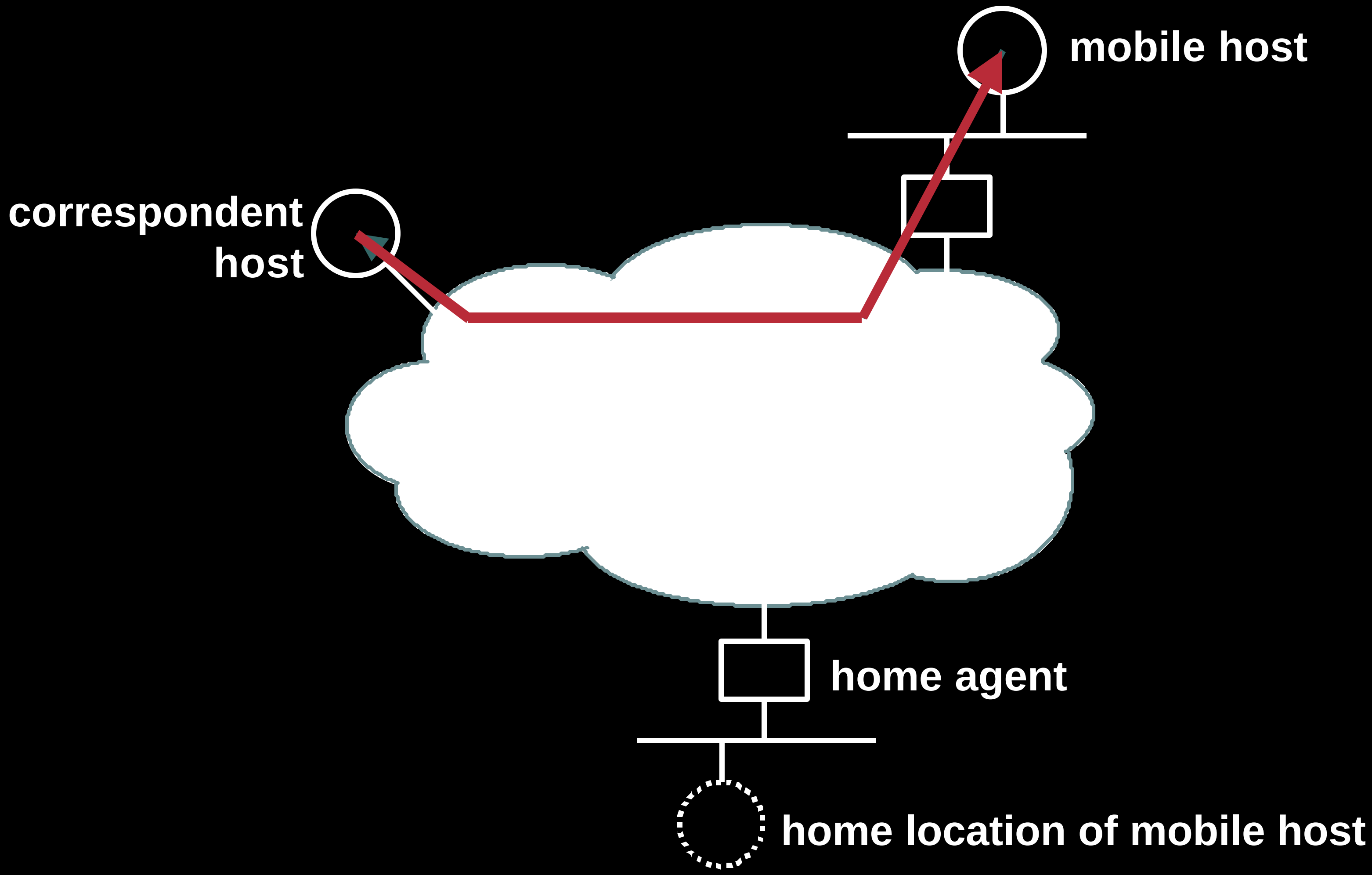
# Mobile IP (v6 version)



# Mobile IP (v6 version)



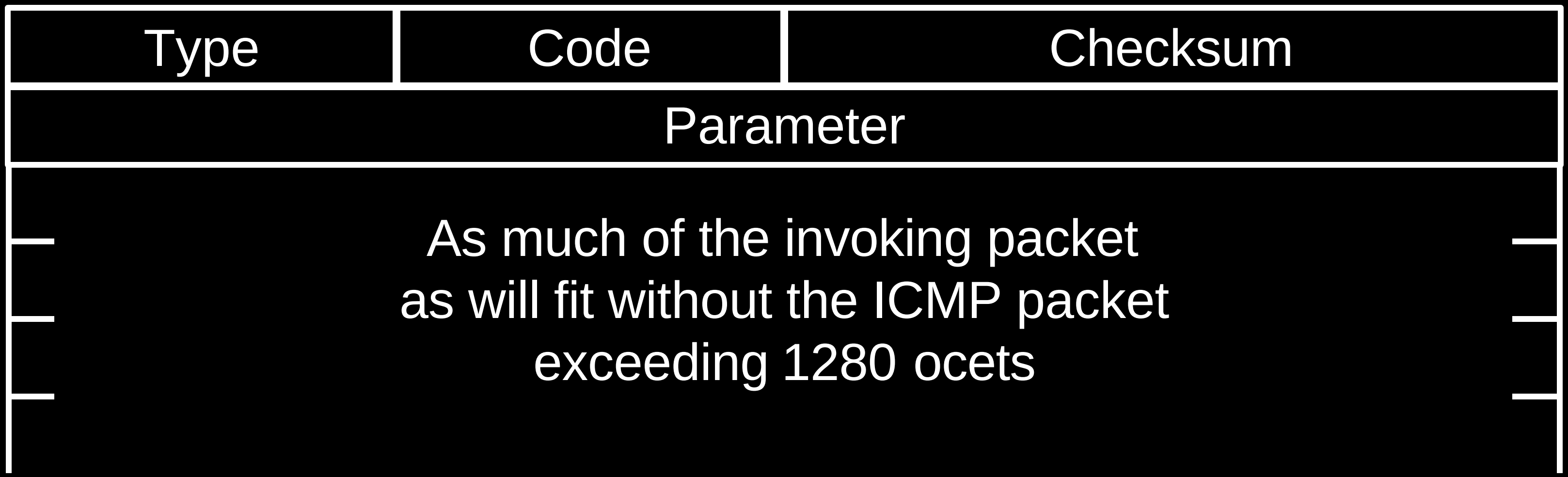
# Mobile IP (v6 version)



# ICMP / Neighbor Discovery

# ICMP Error Messages

**common format:**



Type | Code | Checksum

Parameter

As much of the invoking packet
as will fit without the ICMP packet
exceeding 1280 ocets

(code and parameter are type-specific)

# ICMP Error Message Types

- destination unreachable
  - no route
  - administratively prohibited
  - address unreachable
  - port unreachable
- packet too big
- time exceeded
- parameter problem
  - erroneous header field
  - unrecognized next header type
  - unrecognized option

# ICMP Informational Messages

- Echo request & reply (same as IPv4)
- Multicast listener discovery messages:
  query, report, done (like IGMP for IPv4):

<table>
<tr><td>Type</td><td>Code</td><td colspan="2">Checksum</td></tr>
<tr><td colspan="2">Maximum Response Delay</td><td colspan="2">Reserved</td></tr>
<tr><td colspan="4">Multicast Address</td></tr>
</table>

# Neighbor Discovery

## ICMP message types:

- router solicitation
- router advertisement
- neighbor solicitation
- neighbor advertisement
- redirect

## Functions performed:

- router discovery
- prefix discovery
- autoconfiguration of address & other parameters
- duplicate address detection (DAD)
- neighbor unreachability detection (NUD)
- link-layer address resolution
- first-hop redirect

# Router Advertisements

- Periodically multicast by router to all-nodes multicast address (link scope)
- Contents:
  - “I am a router” (implied)
  - lifetime as default (1 sec – 18 hr)
  - “get addresses from DHCP” flag
  - “get other stuff from DHCP” flag
  - router’s link-layer address
  - link MTU
  - suggested hop limit
  - list of:
    - » prefix
    - » prefix length
    - » valid lifetime
    - » preferred lifetime
    - » on-link flag
    - » autoconfig OK flag
- Not sent frequently enough for unreachability detection

# Other Neighbor Discovery Messages

- Router solicitations
  - sent only at host start-up, to solicit immediate router advert.
  - sent to all-routers multicast address (link scope)
- Neighbor solicitations
  - for address resolution: sent to “solicited node” multicast addr.
  - for unreachability detection: sent to neighbor’s unicast addr.
- Neighbor advertisements
  - for address resolution: sent to unicast address of solicitor
  - for link-layer address change: sent to all-nodes multicast addr.
  - usable for proxy responses (detectable)
  - includes router/host flag

# Serverless Autoconfiguration (“Plug-n-Play”)

- Hosts generally will construct addresses from RA:

  subnet prefix(es) learned from periodic multicast advertisements from neighboring router(s)

  interface IDs generated locally

    MAC addresses : pseudo-random temporary

- Other IP-layer parameters also learned from router adverts (e.g., router addresses, recommended hop limit, etc.)
- Higher-layer info (e.g., DNS server and NTP server addresses) discovered by multicast / anycast-based service-location protocol [details being worked out]
- DHCP is available for those who want explicit control

# Auto-Reconfiguration (“Renumbering”)

- New address prefixes can be introduced, and old ones withdrawn

  we assume some overlap period between old and new, i.e., no “flash cut-over”

  hosts learn prefix lifetimes and preference order from router advertisements

  old TCP connections can survive until end of overlap; new TCP connections use longest preferred lifetime

- Router renumbering protocol, to allow domain-interior routers to learn of prefix introduction / withdrawal

# Minimum MTU

- **Definitions:**

  **link MTUa link’s maximum transmission unit,**
  **i.e., the max IP packet size that can**
  **be transmitted over the link**

  **path MTU the minimum MTU of all the links in a**
  **path between a source and a**
  **destination**

- **Minimum link MTU for IPv6 is 1280 octets (versus 68 octets for IPv4)**
- **On links with MTU < 1280, link-specific fragmentation and reassembly must be used**

**routers deliver packets without further fragmentation**

# Path MTU Discovery

- Implementations are expected to perform path MTU discovery to send packets bigger than 1280 octets:

    for each dest., start by assuming MTU of first-hop link

    if a packet reaches a link in which it cannot fit, will invoke ICMP “packet too big” message to source, reporting the link’s MTU; MTU is cached by source for specific destination

    occasionally discard cached MTU to detect possible increase

- Minimal implementation can omit path MTU discovery as long as all packets kept " 1280 octets

    e.g., in a boot ROM implementation

# Outline

- Protocol Background
- Technology Differences
- Enhanced Capabilities
- **Inaccuracies & Speculation**
- Transition Technologies

# Lack of demand

- There is no shortage of v4 space
- The only people who ask about IPv6 are people who have heard something about it
- IPv6 exhibits no added functionality over IPv4 + NAT
  - True for client/server apps with server on public side
  - False for peer-to-peer apps & servers behind nat

# Deployments

- IPv6 deployments will occur piecewise from the edge.

  Core infrastructure only moving when significant customer usage demands it.

  Consumers should never be exposed to which protocol they are running, so demand will be implicit.

  Platforms and products that are updated first need to address the lack of ubiquity. Whenever possible, devices and applications should be capable of both IPv4 & IPv6, to minimize the delays and potential failures inherent in translation points.

# Routing

- IPv6 routing will change drastically before it becomes production
- Routing is still a big problem in IPv6

  IPv6 allocations and routing are cidr based; massive aggregation through new allocations; <12k origin AS's for explicit policy

  - only problem is providers punching holes in their aggregates

# Multicast

- what happens to multicast routing when one million gamers setup half a million *,G and a few million S,G pairs

SSM w/unicast prefix based groups

# Interface IDs

- IPv6 has many privacy issues because it uses an interface ID derived from hardware

  Lowest-order 64-bit field of unicast address may be assigned in several different ways:

  – auto-configured from a 64-bit EUI-64, or expanded from a 48-bit MAC address (e.g., Ethernet address)

  – auto-generated pseudo-random number RFC3041

    (specifically designed to address privacy concerns)

  – assigned via DHCP

  – manually configured

  – possibly other methods in the future (crypto derived)

# Infrastructure impact

- RFC3041 addresses require frequent updating DNS forward & reverse

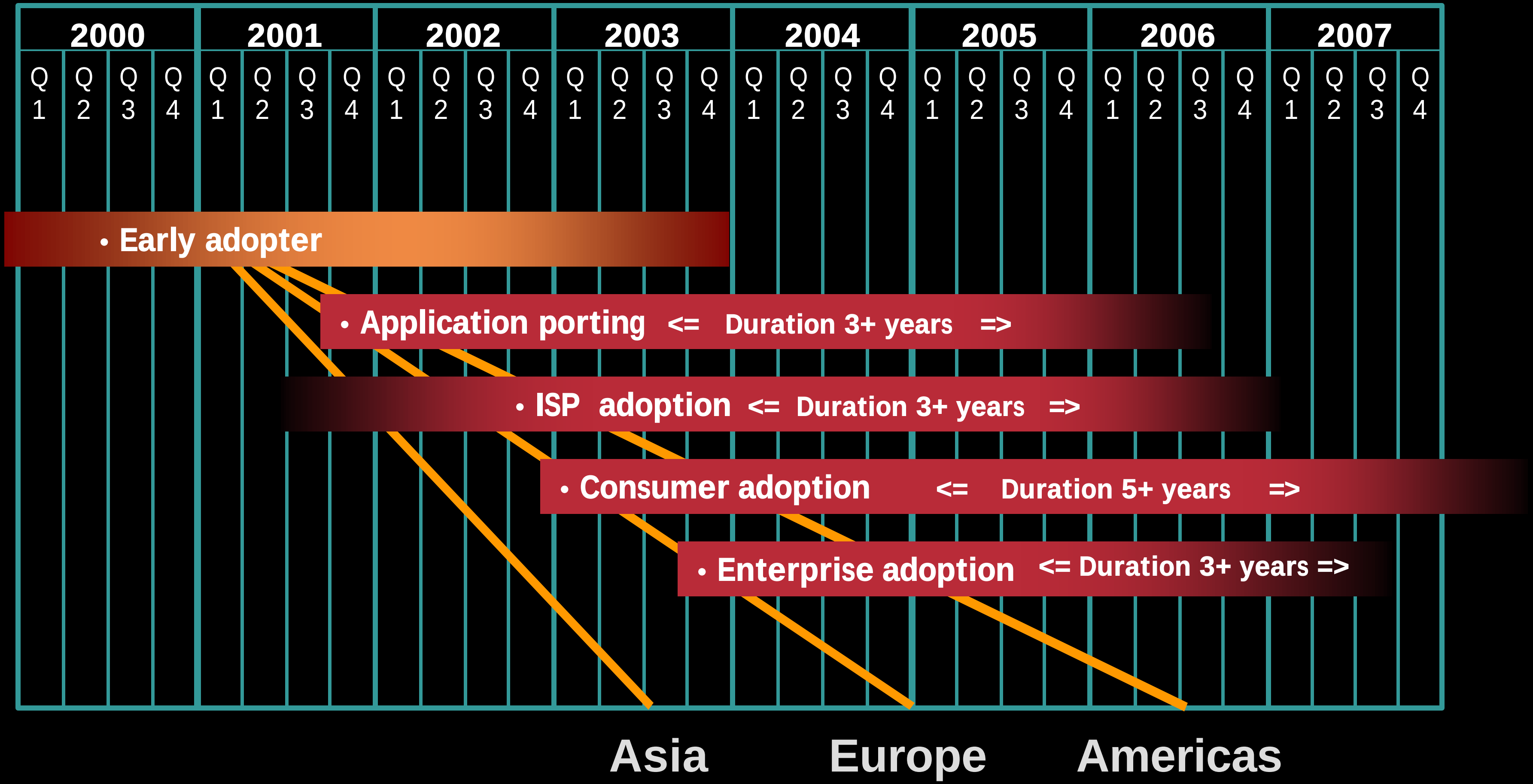
  Logical disconnect between a privacy function used in combination with a registration service

# Outline

- Protocol Background
- Technology Differences
- Enhanced Capabilities
- Inaccuracies & Speculation
- **Transition Technologies**

# IPv6 Timeline

## (A pragmatic projection)



- Early adopter
- Application porting <= Duration 3+ years =>
- ISP adoption <= Duration 3+ years =>
- Consumer adoption <= Duration 5+ years =>
- Enterprise adoption <= Duration 3+ years =>

Asia Europe Americas

# IPv4-IPv6 Transition / Co-Existence

A wide range of techniques have been identified and implemented, basically falling into three categories:

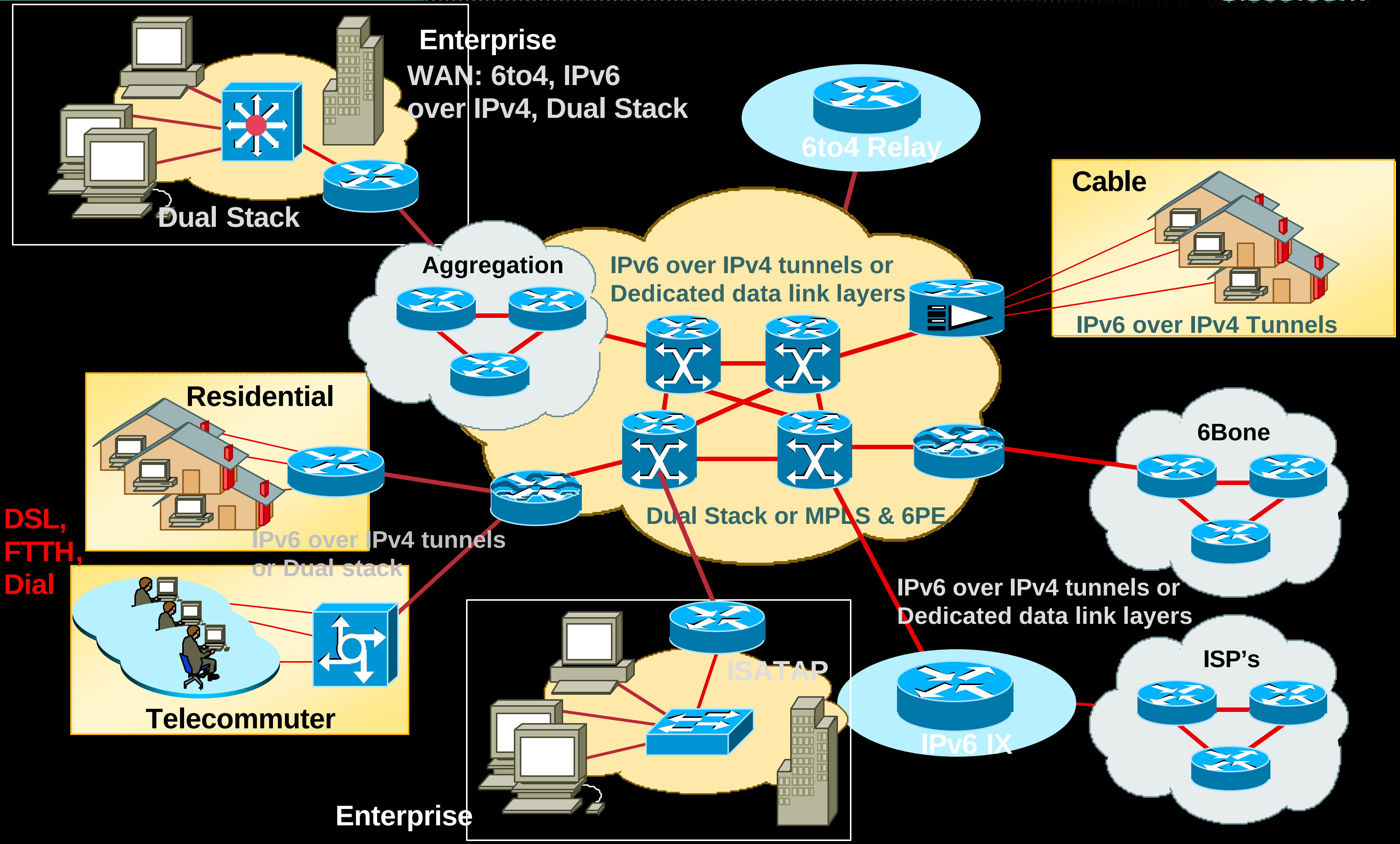
(1) **Dual-stack** techniques, to allow IPv4 and IPv6 to co-exist in the same devices and networks

(2) **Tunneling** techniques, to avoid order dependencies when upgrading hosts, routers, or regions

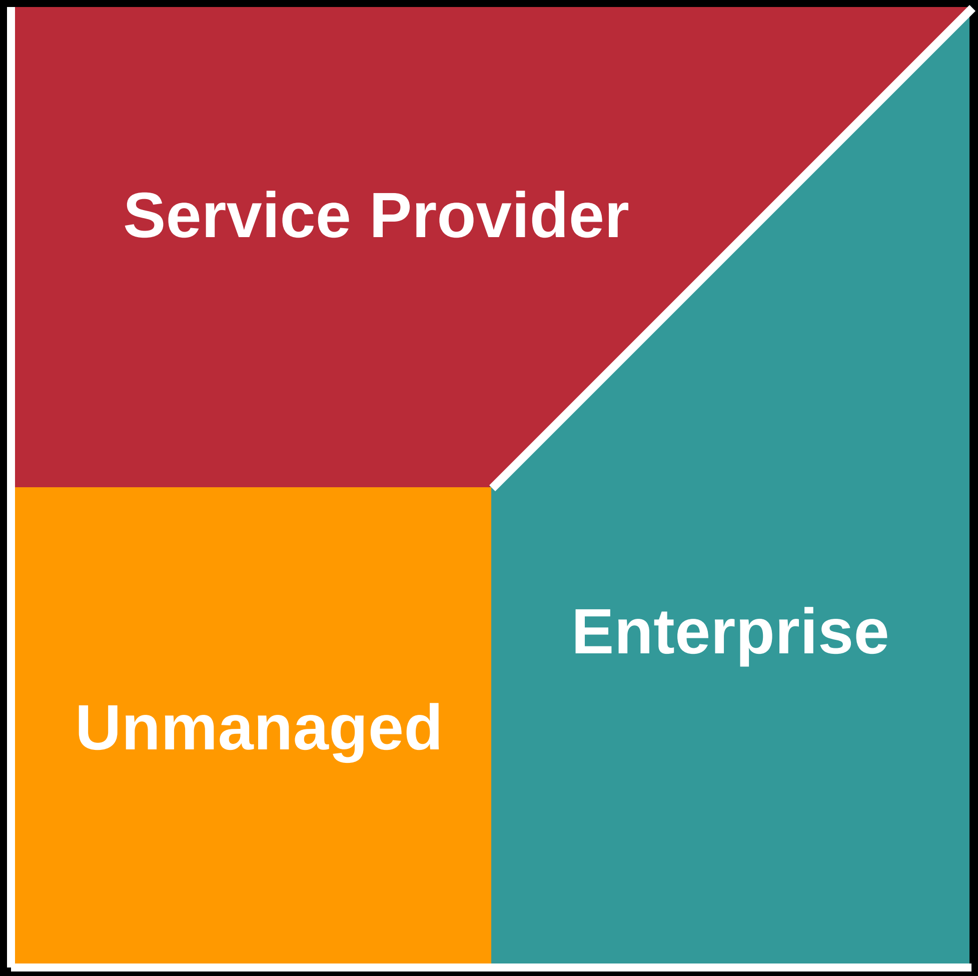
(3) **Translation** techniques, to allow IPv6-only devices to communicate with IPv4-only devices

Expect all of these to be used, in combination

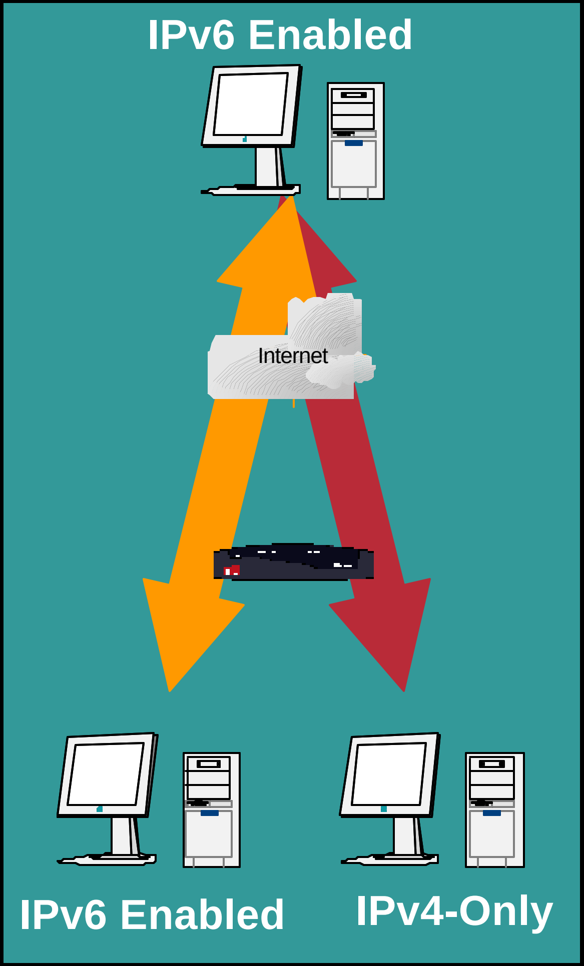
# Transition environments


Enterprise
WAN: 6to4, IPv6 over IPv4, Dual Stack
Dual Stack
6to4 Relay
Cable
IPv6 over IPv4 Tunnels
Aggregation
IPv6 over IPv4 tunnels or Dedicated data link layers
Residential
DSL, FTTH, Dial
IPv6 over IPv4 tunnels or Dual stack
Telecommuter
Dual Stack or MPLS & 6PE
6Bone
IPv6 over IPv4 tunnels or Dedicated data link layers
ISATAP
IPv6 IX
ISP's
Enterprise

# Environments

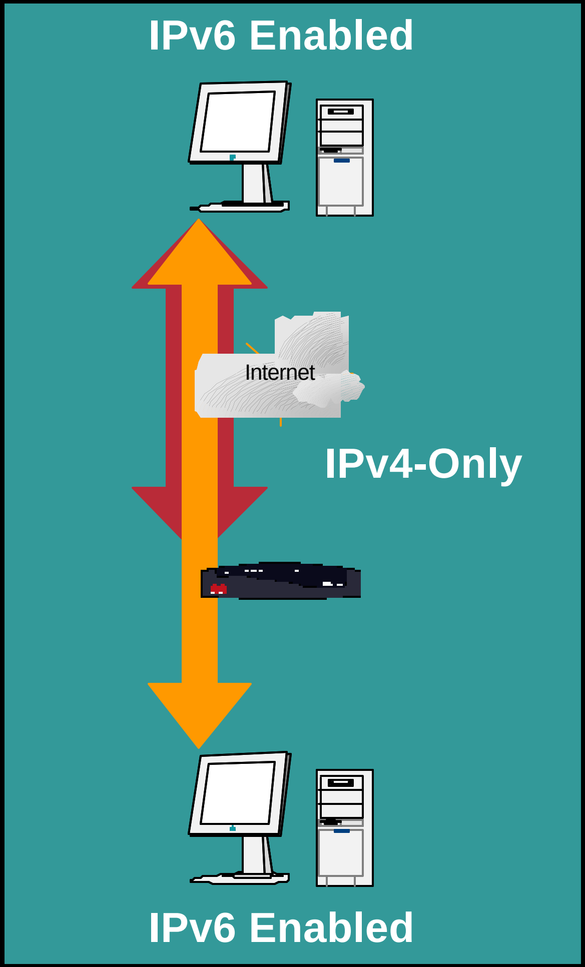


# Tools – Dual Stack



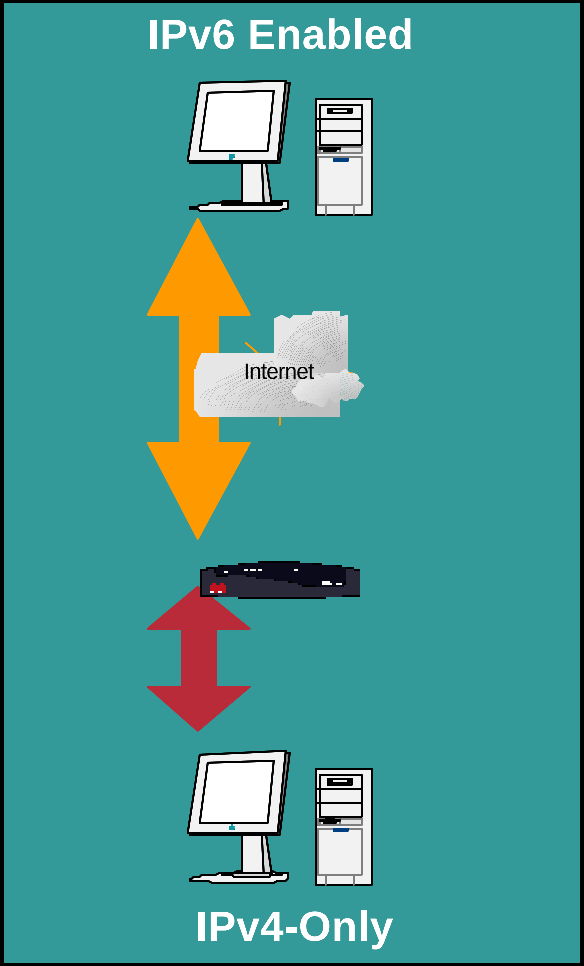
- **Primary tool**
- **Allows continued 'normal' operation with IPv4-only nodes**
- **Address selection rules generally prefer IPv6**
- **DSTM variant allows temporary use of IPv4 pool**

# Tools – Tunneling



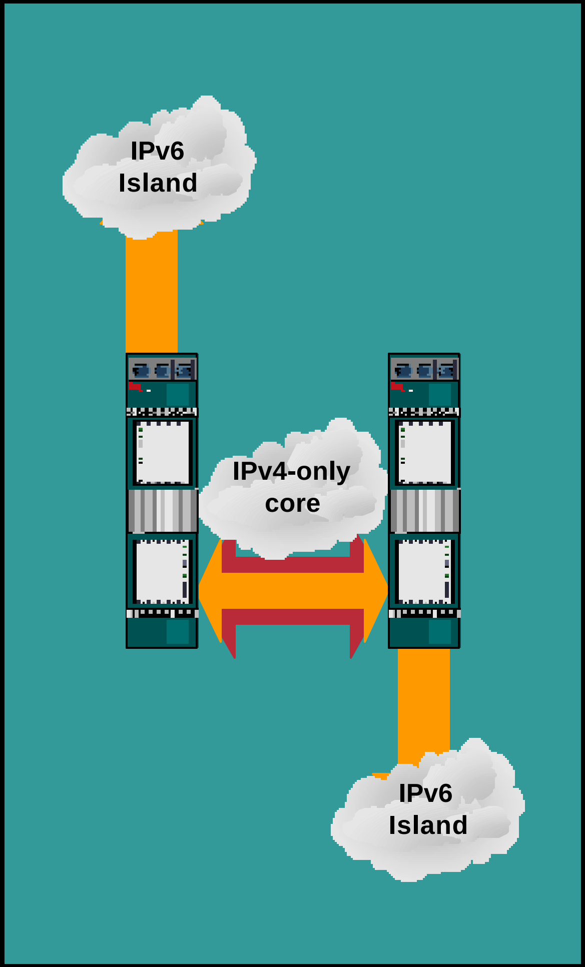
- Nodes view IPv4 network as a logical NBMA link-layer
- May be used in conjunction with dual-stack

# Tools – Translation



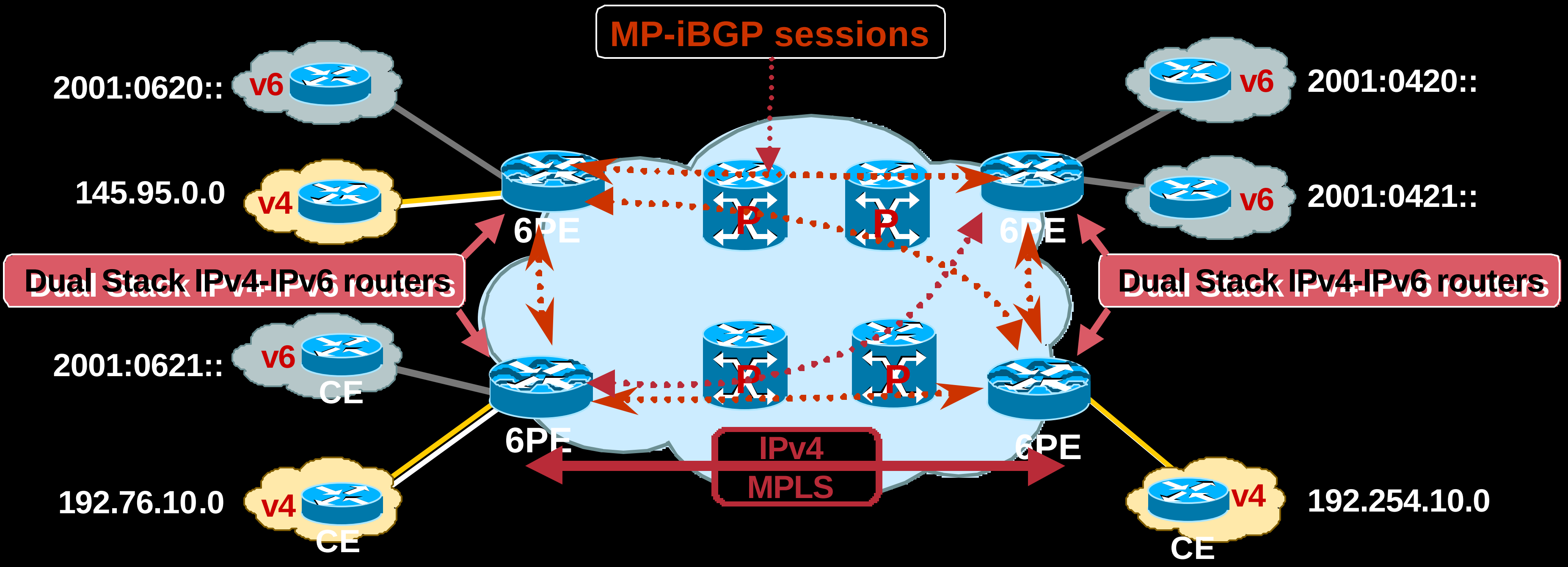
- Allows for the case where some components are IPv6-only while others are IPv4-only
- Tool of last resort
- Pay attention to scaling properties
- Same application issues as IPv4/IPv4 translation

# Tools – BGP tunnel



- Service provider can incrementally upgrade PE routers with active customers
- Sites are connected to Dual Stack MP-BGP-speaking edge router
- Transport across the IPv4 core can be any tunneling mechanism

# IPv6 Provider Edge Router (6PE) over MPLS



- IPv4 or MPLS Core Infrastructure is IPv6-unaware
- PEs are updated to support Dual Stack/6PE
- IPv6 reachability exchanged among 6PEs via iBGP (MP-BGP)
- IPv6 packets transported from 6PE to 6PE inside MPLS

# Questions?

Cisco Systems
®
Empowering the Internet Generation℠

Cisco.com

# Routing Header

# Routing

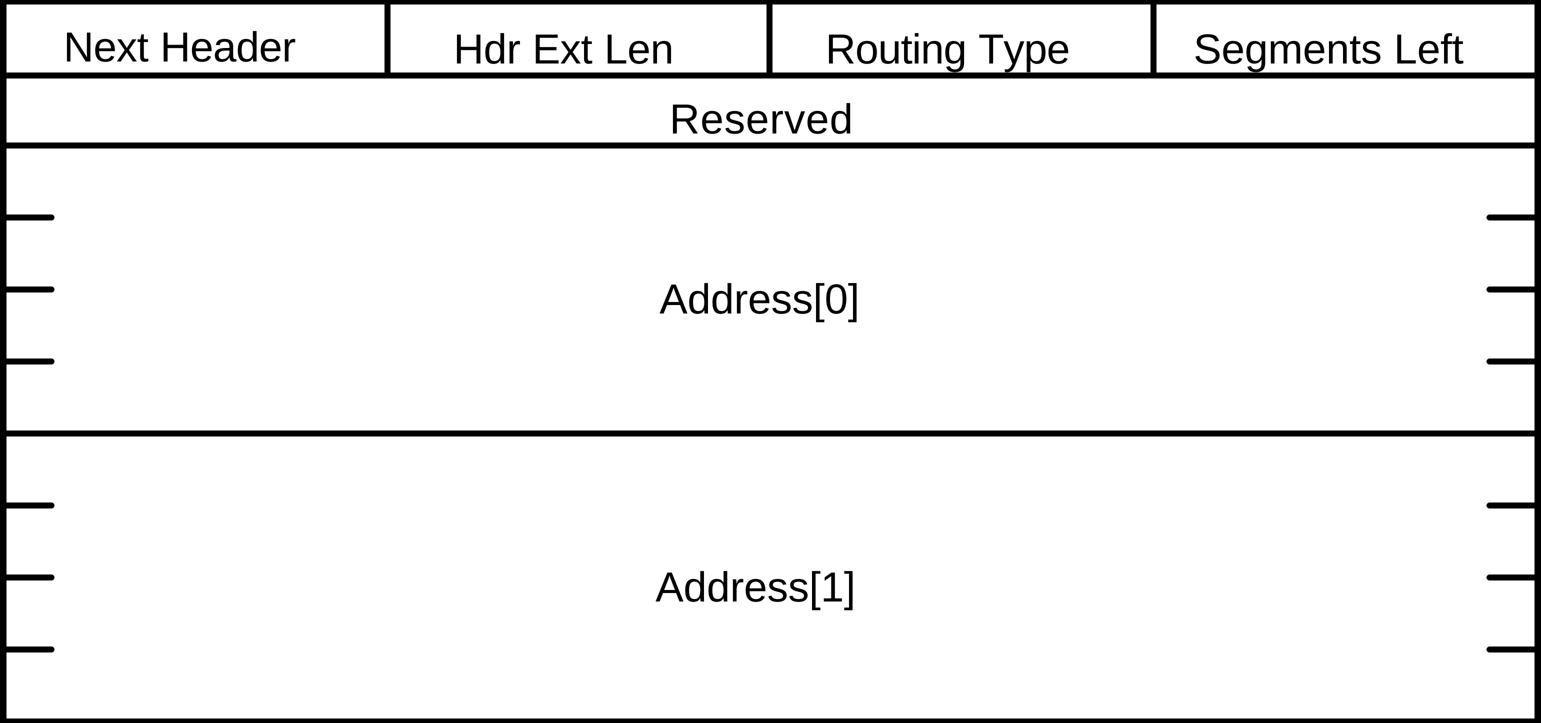
- Same “longest-prefix match” routing as IPv4 CIDR
- Straightforward changes to existing IPv4 routing protocols to handle bigger addresses

  unicast: OSPF, RIP-II, IS-IS, BGP4+, …

  multicast: MOSPF, PIM, …
- Use of Routing header with anycast addresses allows routing packets through particular regions

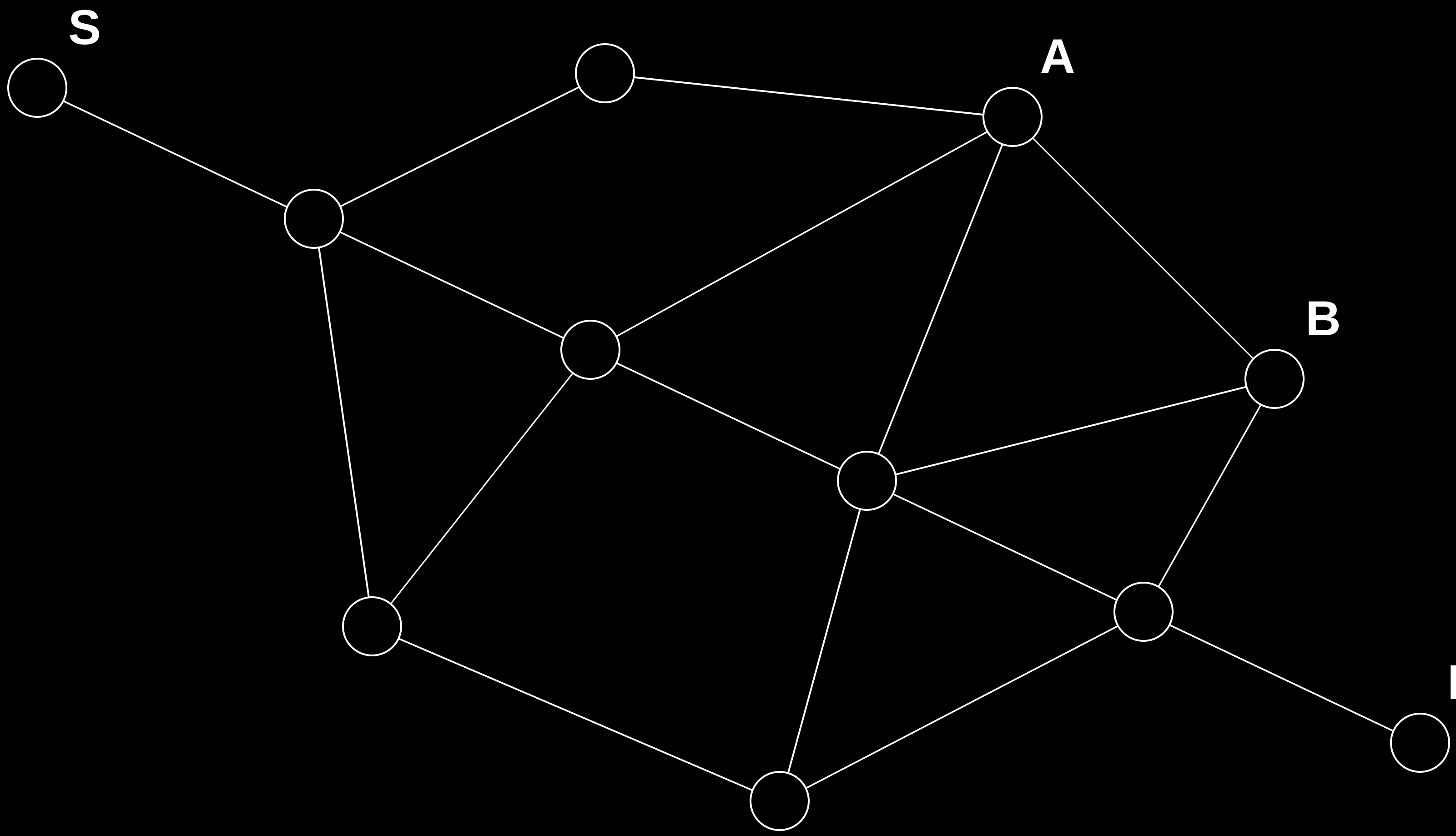
  e.g., for provider selection, policy, performance, etc.

# Routing Header

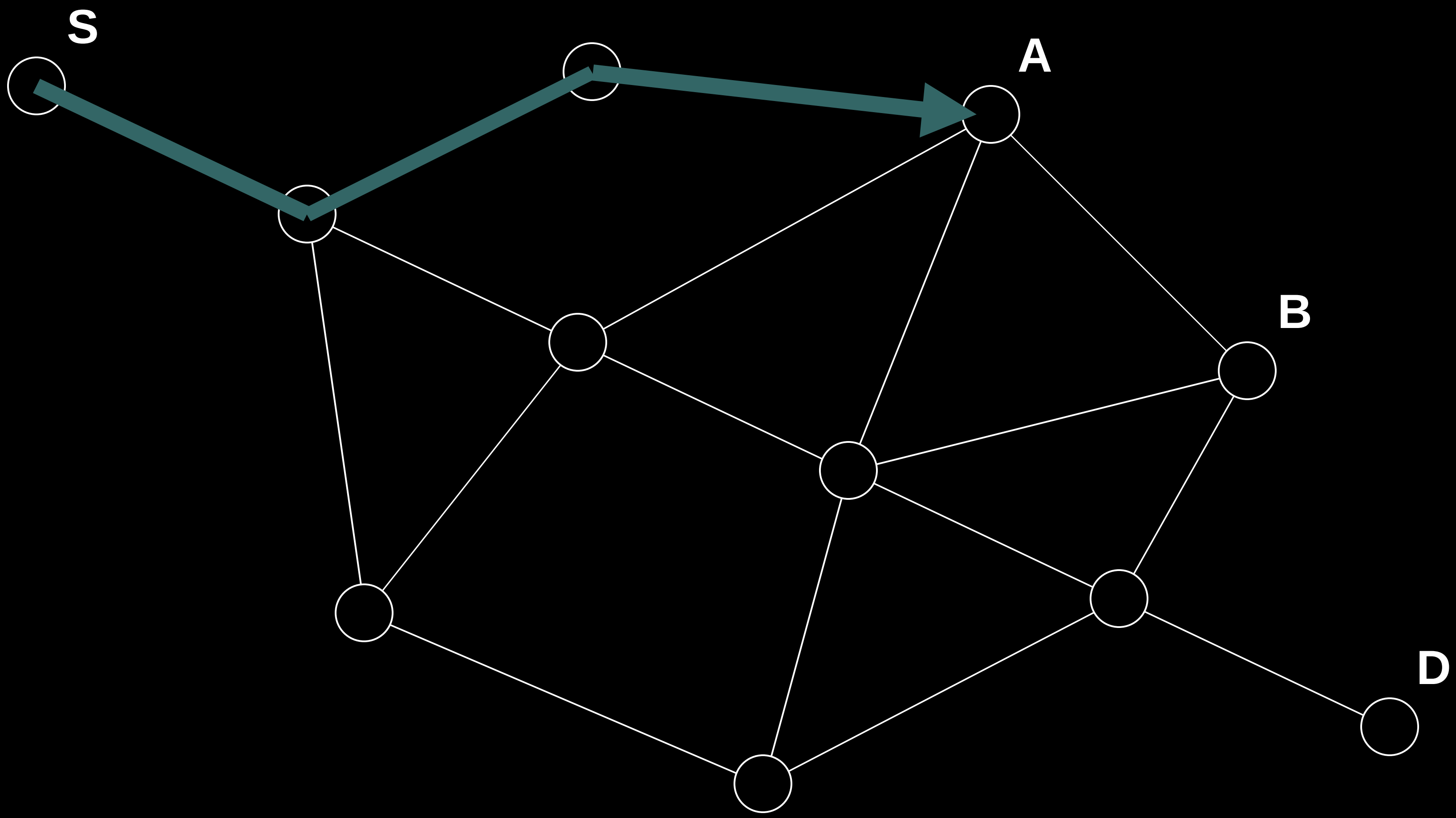


| Next Header | Hdr Ext Len | Routing Type | Segments Left |
|---|---|---|---|
| Reserved | | | |
| Address[0] | | | |
| Address[1] | | | |

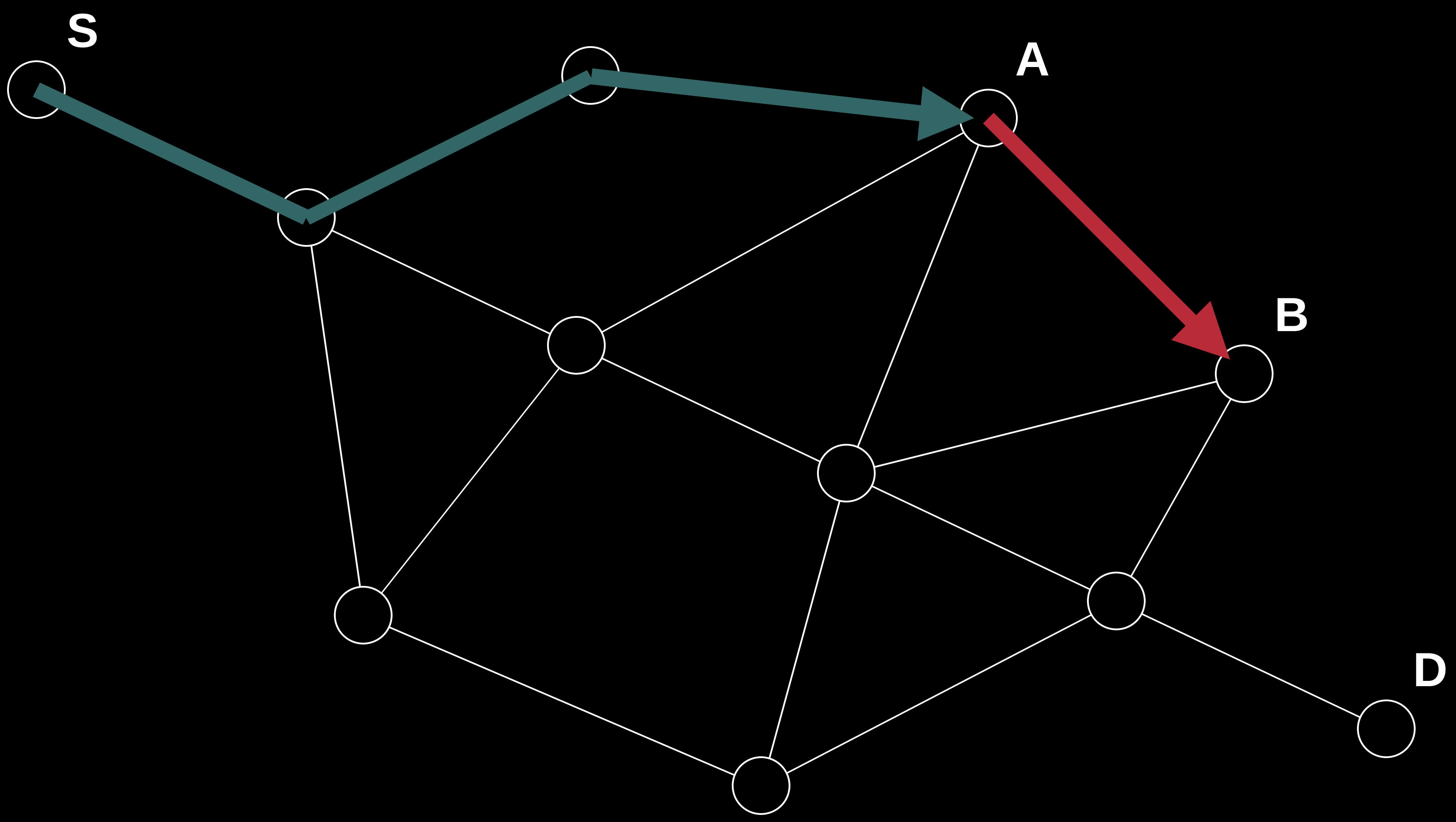
# Example of Using the Routing Header



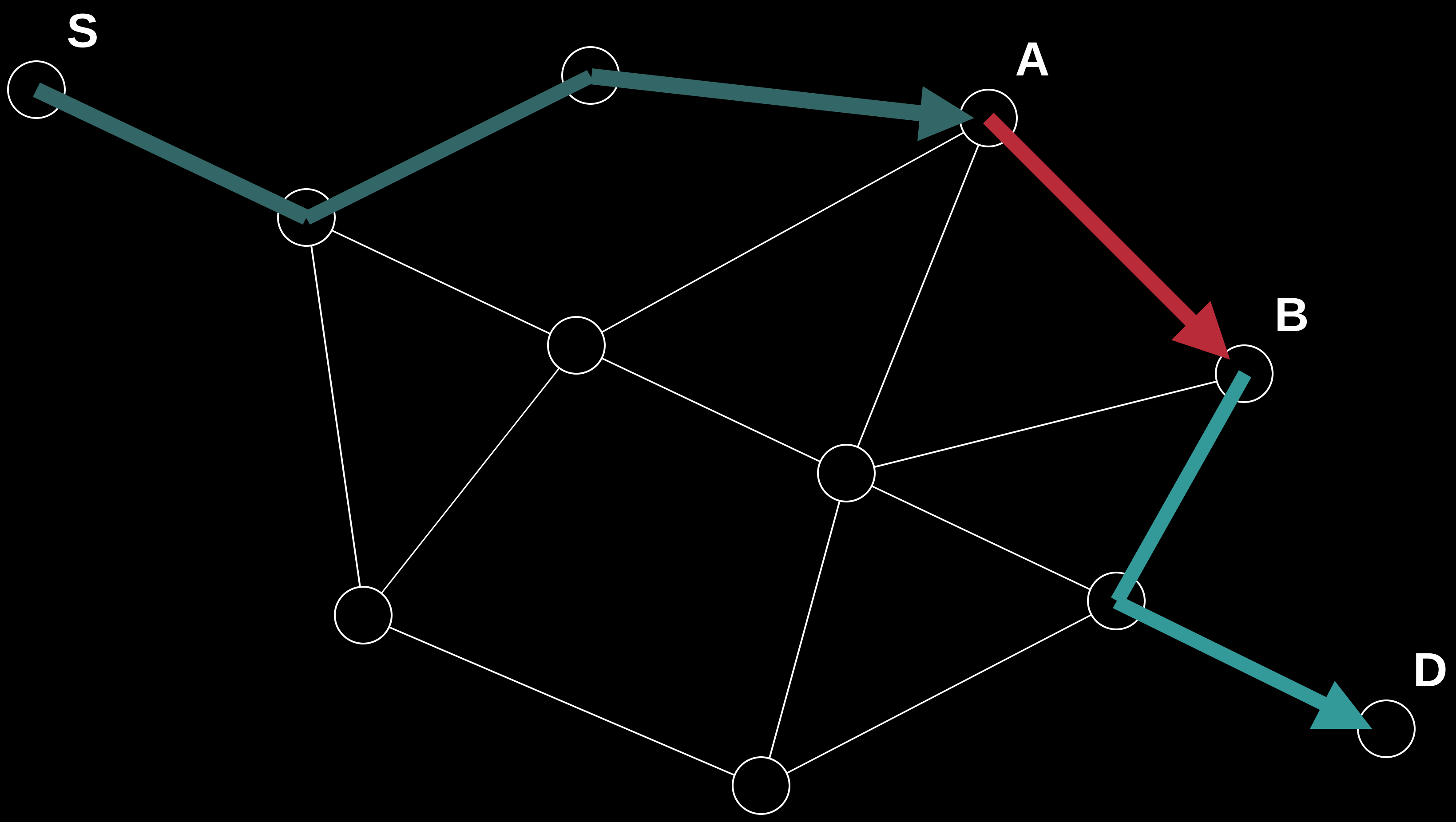
# Example of Using the Routing Header



# Example of Using the Routing Header



# Example of Using the Routing Header



# Porting Issues

# Effects on higher layers

- Changes TCP/UDP checksum “pseudo-header”
- Affects anything that reads/writes/stores/passes IP addresses (just about every higher protocol)
- Packet lifetime no longer limited by IP layer
  (it never was, anyway!)
- Bigger IP header must be taken into account when computing max payload sizes
- New DNS record type: AAAA and (new) A6
- …

# Sockets API Changes

- Name to Address Translation Functions
- Address Conversion Functions
- Address Data Structures
- Wildcard Addresses
- Constant Additions
- Core Sockets Functions
- Socket Options
- New Macros

# Core Sockets Functions

- Core APIs
  - Use IPv6 Family and Address Structures
  - socket() Uses PF_INET6
- Functions that pass addresses
  - bind()
  - connect()
  - sendmsg()
  - sendto()
- Functions that return addresses
  - accept()
  - recvfrom()
  - recvmsg()
  - getpeername()
  - getsockname()

# Name to Address Translation

- getaddrinfo()
  - Pass in nodename and/or servicename string
    - Can Be Address and/or Port
  - Optional Hints for Family, Type and Protocol
    - Flags – AI_PASSIVE, AI_CANNONNAME, AI_NUMERICHOST, AI_NUMERICSERV, AI_V4MAPPED, AI_ALL, AI_ADDRCONFIG
  - Pointer to Linked List of addrinfo structures Returned
    - Multiple Addresses to Choose From
- freeaddrinfo()

```
struct addrinfo {
     int ai_flags;
     int ai_family;
     int ai_socktype;
     int ai_protocol;
     size_t ai_addrlen;
     char *ai_canonname;
     struct sockaddr *ai_addr;
     struct addrinfo *ai_next;
     };
```

```
int getaddrinfo(
    IN const char FAR * nodename,
    IN const char FAR * servname,
    IN const struct addrinfo FAR * hints,
    OUT struct addrinfo FAR * FAR * res
    );
```

# Address to Name Translation

- getnameinfo()
  - Pass in address (v4 or v6) and port
    - Size Indicated by salen
    - Also Size for Name and Service buffers (NI_MAXHOST, NI_MAXSERV)
  - Flags
    - NI_NOFQDN
    - NI_NUMERICHOST
    - NI_NAMEREQD
    - NI_NUMERICSERV
    - NI_DGRAM

```
int getnameinfo(
    IN const struct sockaddr FAR * sa,
    IN socklen_t salen,
    OUT char FAR * host,
    IN size_t hostlen,
    OUT char FAR * serv,
    IN size_t servlen,
    IN int flags
    );
```

# Porting Environments

- Node Types
    - IPv4-only
    - IPv6-only
    - IPv6/IPv4
- Application Types
    - IPv6-unaware
    - IPv6-capable
    - IPv6-required
- IPv4 Mapped Addresses

# Porting Issues

- Running on ANY System
  - Including IPv4-only
- Address Size Issues
- New IPv6 APIs for IPv4/IPv6
- Ordering of API Calls
- User Interface Issues
- Higher Layer Protocol Changes

# Specific things to look for

- Storing IP address in 4 bytes of an array.
- Use of explicit dotted decimal format in UI.
- Obsolete / New:

| | | |
|---|---|---|
| AF_INET | replaced by | AF_INET6 |
| SOCKADDR_IN | replaced by | SOCKADDR_STORAGE |
| IPPROTO_IP | replaced by | IPPROTO_IPV6 |
| IP_MULTICAST_LOOP | replaced by | SIO_MULTIPOINT_LOOPBACK |
| gethostbyname | replaced by | getaddrinfo |
| gethostbyaddr | replaced by | getnameinfo |

# IPv6 literal addresses in URL’s

- From RFC 2732

Literal IPv6 Address Format in URL's Syntax To use a literal IPv6 address in a URL, the literal address should be enclosed in "[" and "]" characters. For example the following literal IPv6 addresses: FEDC:BA98:7654:3210:FEDC:BA98:7654:3210

3ffe:2a00:100:7031::1

::192.9.5.5

2010:836B:4179::836B:4179

would be represented as in the following example URLs: http://[FEDC:BA98:7654:3210:FEDC:BA98:7654:3210]:80/index.html

http://[3ffe:2a00:100:7031::1]

http://[::192.9.5.5]/ipng

http://[2010:836B:4179::836B:4179]

# Other Issues

- Renumbering & Mobility routinely result in changing IP Addresses –

  Use Names and Resolve, Don’t Cache

- Multihomed Servers

  More Common with IPv6

  Try All Addresses Returned

- Using New IPv6 Functionality

# Porting Steps -Summary

- Use IPv4/IPv6 Protocol/Address Family
- Fix Address Structures
  - in6_addr
  - sockaddr_in6
  - sockaddr_storage to allocate storage
- Fix Wildcard Address Use
  - in6addr_any, IN6ADDR_ANY_INIT
  - in6addr_loopback, IN6ADDR_LOOPBACK_INIT
- Use IPv6 Socket Options
  - IPPROTO_IPV6, Options as Needed
- Use getaddrinfo()
  - For Address Resolution

Cisco.com

# IPv4 - IPv6
# Co-Existence / Transition

# Impediments to IPv6 deployment

- Applications
- Applications
- Applications

**Move to the new APIs NOW**

# Dual-Stack Approach

- When adding IPv6 to a system, do not delete IPv4

  this multi-protocol approach is familiar and well-understood (e.g., for AppleTalk, IPX, etc.)

  note: in most cases, IPv6 will be bundled with new OS releases, not an extra-cost add-on

- Applications (or libraries) choose IP version to use

  when initiating, based on DNS response:

  Prefer scope match first, when equal IPv6 over IPv4

  when responding, based on version of initiating packet

- This allows indefinite co-existence of IPv4 and IPv6, and gradual app-by-app upgrades to IPv6 usage

# Tunnels to Get Through IPv6-Ignorant Routers

- Encapsulate IPv6 packets inside IPv4 packets (or MPLS frames)
- Many methods exist for establishing tunnels:

  manual configuration

  “tunnel brokers” (using web-based service to create a tunnel)

  automatic (depricated, using IPv4 as low 32bits of IPv6)

  “6-over-4” (intra-domain, using IPv4 multicast as virtual LAN)

  “6-to-4” (inter-domain, using IPv4 addr as IPv6 site prefix)
- Can view this as:

  IPv6 using IPv4 as a virtual NBMA link-layer, or

  an IPv6 VPN (virtual public network), over the IPv4 Internet

# Translation

- May prefer to use IPv6-IPv4 protocol translation for:
  - new kinds of Internet devices (e.g., cell phones, cars, appliances)
  - benefits of shedding IPv4 stack (e.g., serverless autoconfig)
- This is a simple extension to NAT techniques, to translate header format as well as addresses
  - IPv6 nodes behind a translator get full IPv6 functionality when talking to other IPv6 nodes located anywhere
  - they get the normal (i.e., degraded) NAT functionality when talking to IPv4 devices
  - drawback : minimal gain over IPv4/IPv4 NAT approach

# Current Status

# Standards

- core IPv6 specifications are IETF Draft Standards
  => well-tested & stable

  IPv6 base spec, ICMPv6, Neighbor Discovery, PMTU Discovery, IPv6-over-Ethernet, IPv6-over-PPP,...

- other important specs are further behind on the standards track, but in good shape

  mobile IPv6, header compression, A6 DNS support,...

  for up-to-date status:   playground.sun.com/ipng

- UMTS R5 cellular wireless standards mandate IPv6

# Implementations

- Most IP stack vendors have an implementation at some stage of completeness

  some are shipping supported product today,
  e.g., 3Com, *BSD(KAME), Cisco, Epilogue, Ericsson/Telebit, IBM, Hitachi, NEC, Nortel, Sun, Juniper, Trumpet

  others have beta releases now, supported products soon,
  e.g., HP / Compaq, Linux community, Microsoft

  others rumored to be implementing, but status unkown (to me), e.g., Apple, Bull, Mentat, Novell, SGI

  (see playground.sun.com/ipng for most recent status reports)

- Good attendance at frequent testing events

# Next Steps

# So what can I do?

- Begin porting NOW!

- Establish test networks to verify configurations, and application compatibility

# For More Information

- http://www.ietf.org/html.charters/ipngwg-charter.html
- http://www.ietf.org/html.charters/ngtrans-charter.html
- http://playground.sun.com/ipv6/
- http://www.6bone.net/ngtrans/

# For More Information

- http://www.6bone.net
- http://www.ipv6forum.com
- http://www.ipv6.org
- http://www.cisco.com/ipv6
- http://www.microsoft.com/ipv6

# For More Information

- BGP4+ References

  RFC2858 Multiprotocol extension to BGP
  RFC2545 BGP MP for IPv6
  RFC2842 Capability negotiation

- RIPng RFC2080

# Other Sources of Information

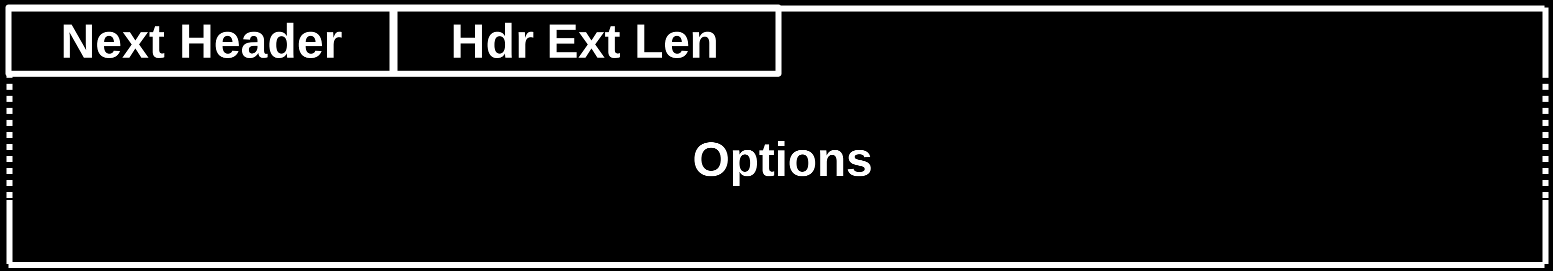
- **Books**

  **IPv6, The New Internet Protocol**
  **by Christian Huitema (Prentice Hall)**

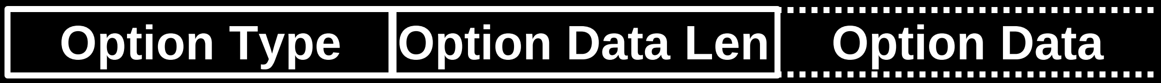
  **Internetworking IPv6 with Cisco Routers**
  **by Silvano Gai (McGraw-Hill)**

  **and many more... (14 hits at Amazon.com)**

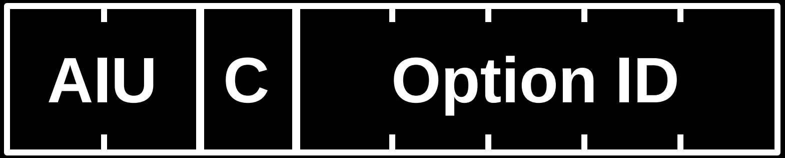
# Hop-by-Hop Options Header
# & Destination Options Header


Next Header
Hdr Ext Len
Options

**are containers for variable-length options:**


Option Type
Option Data Len
Option Data

# Option Type Encoding



AIU

C

Option ID

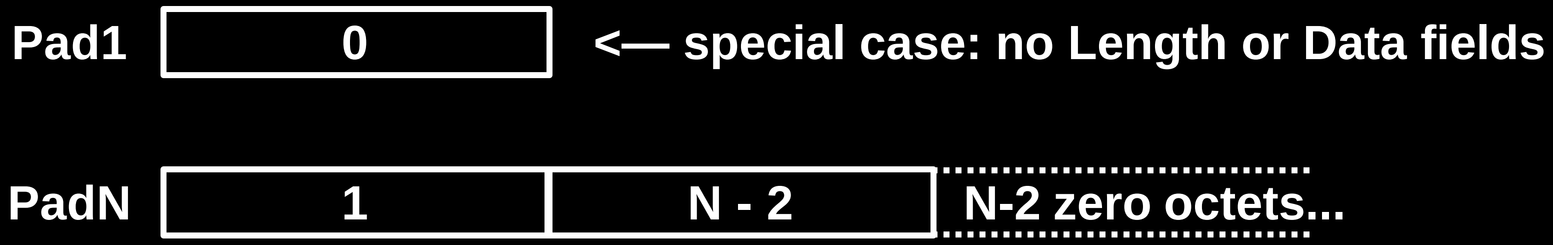
**AIU — action if unrecognized:**

**00 — skip over option**
**01 — discard packet**
**10 — discard packet &**
**send ICMP Unrecognized Type to source**
**11 — discard packet &**
**send ICMP Unrecognized Type to source**
**only if destination was not multicast**

**C — set if Option Data changes en-route**
**(Hop-by-Hop Options only)**

# Option Alignment and Padding

two padding options:



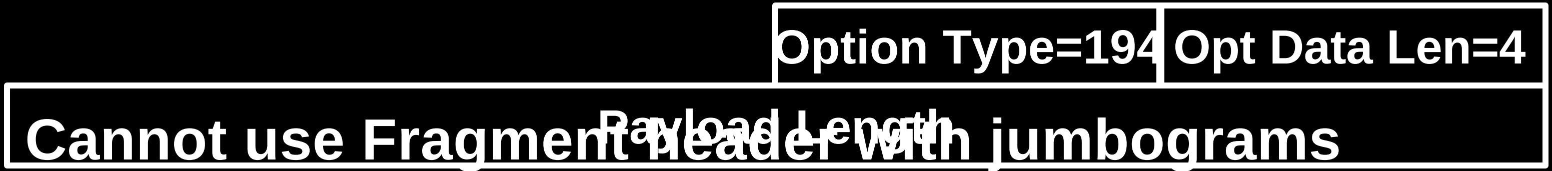
Pad1 | 0 | <— special case: no Length or Data fields

PadN | 1 | N - 2 | N-2 zero octets...

- used to align options so multi-byte data fields fall on natural binary boundaries
- used to pad out containing header to an integer multiple of 8 bytes

# Maximum Packet Size

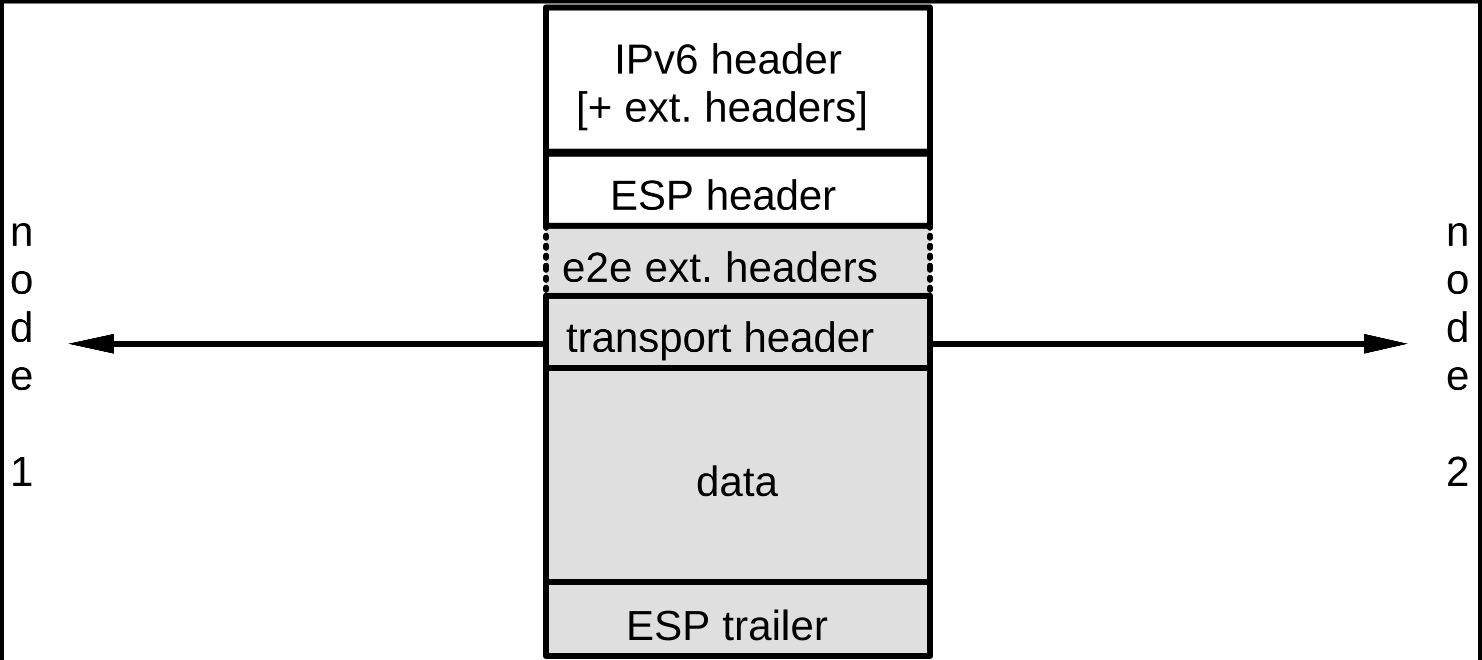
- Base IPv6 header supports payloads of up to 65,535 bytes (not including 40 byte IPv6 header)
- Jumbo payloads can be carried by setting IPv6 Payload Length field to zero, and adding the “jumbogram” hop-by-hop option:



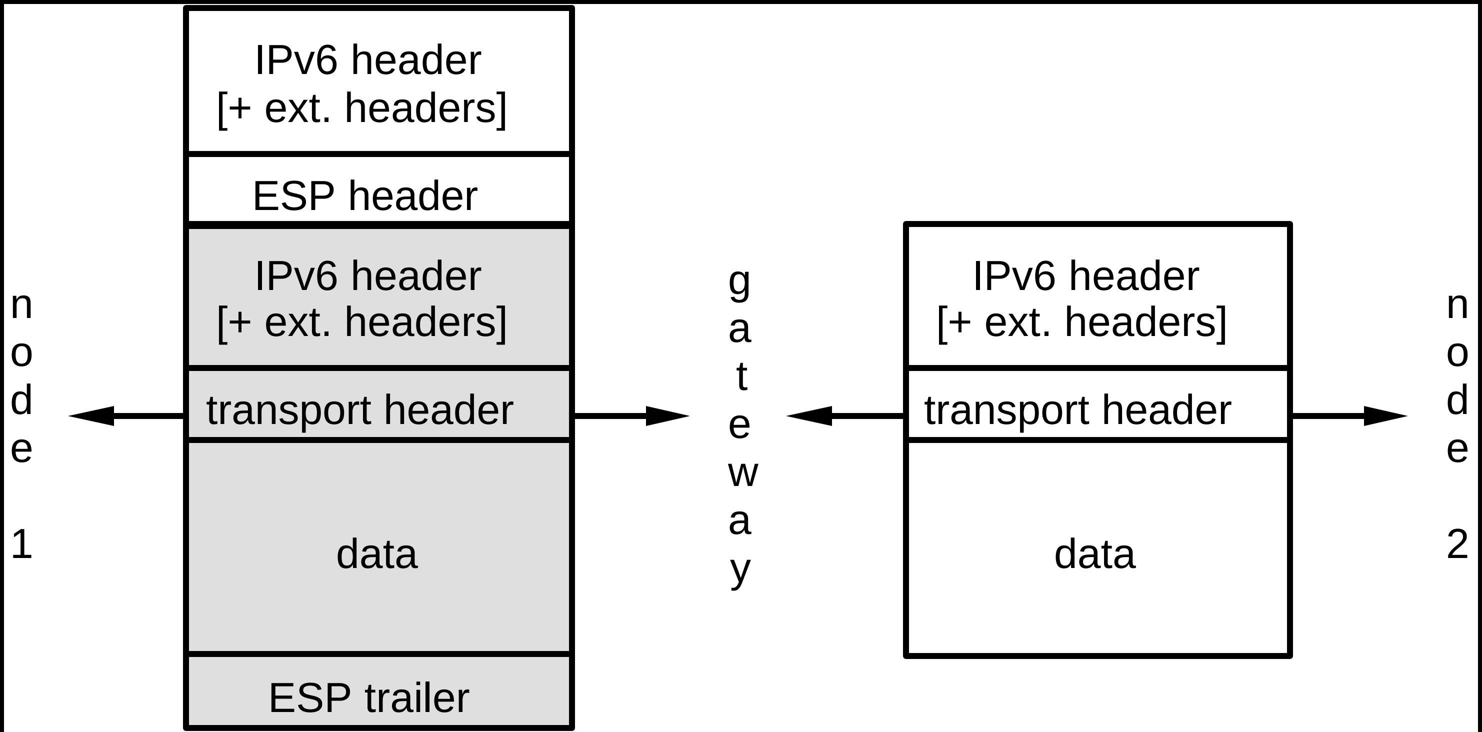
| | |
|---|---|
| Option Type=194 | Opt Data Len=4 |
| Payload Length | |

- Cannot use Fragment Header with jumbograms

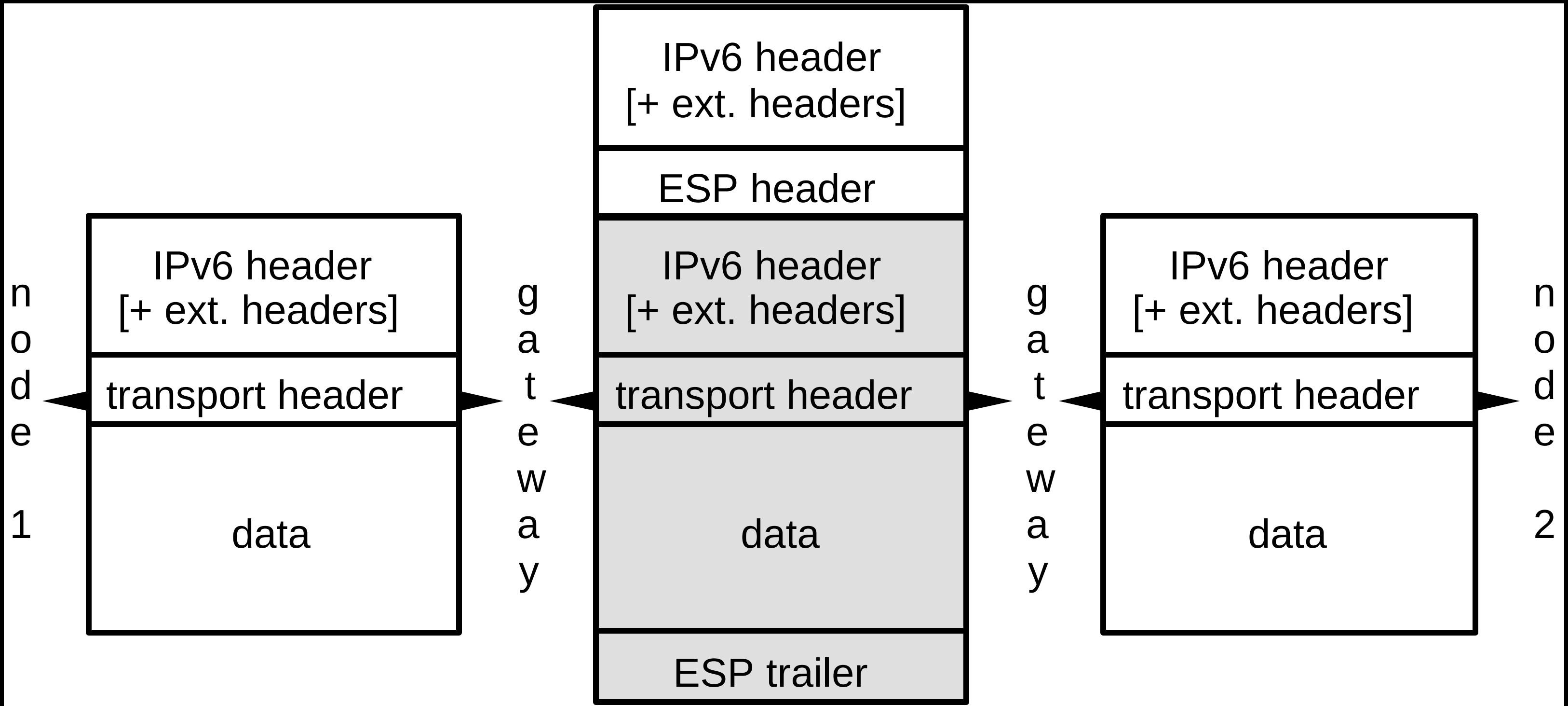
# Transport Mode ESP (End-to-End)



# Tunnel Mode ESP (End to Security Gateway)


node 1
IPv6 header [+ ext. headers]
ESP header
IPv6 header [+ ext. headers]
transport header
data
ESP trailer
gateway
IPv6 header [+ ext. headers]
transport header
data
node 2

# Tunnel Mode ESP (Gateway to Gateway)



node 1
IPv6 header [+ ext. headers]
transport header
data
gateway
IPv6 header [+ ext. headers]
ESP header
IPv6 header [+ ext. headers]
transport header
data
ESP trailer
gateway
IPv6 header [+ ext. headers]
transport header
data
node 2

# Deployment of IPv6 Services

- Satisfy Business Drivers
  - applications requiring end-to-end IPv6 traffic forwarding geographies with registry allocations issues
- No Flag Day
- No Performance Penalty
  - implementation must be scalable and reliable
- Minimize operational upgrade costs and training expenses
  - Investment Protection & Low startup cost
- Incremental Upgrade/Deployment
  - Preserve IPv6 - IPv4 connectivity/transparency

***Strategy that reflects this …***

***Starting with Edge upgrades enable IPv6 service offerings now***