Communication Networks

Spring 2020

Laurent Vanbever

nsg.ee.ethz.ch

ETH Zürich (D-ITET)

May 18 2020

Materials inspired from Scott Shenker and Jennifer Rexford

Last Monday on

Communication Networks

Web

Video Streaming

http://www.google.ch

HTTP-based

Web

Video Streaming

http://www.google.ch

HTTP performance goals vary depending on who you ask

	User	Network operators	Content provider
			
wish	fast downloads high availability	no overload	happy users cost-effective infrastructure
solution	Improve HTTP to compensate for TCP weakspots	Caching and Replication	

Considering the time to retrieve *n* small objects, pipelining wins

	# RTTS
one-at-a-time	~2*n*
M concurrent	~2*n*/M
persistent	~*n*+1
pipelined	2

Considering the time to retrieve *n* big objects,
there is no clear winners as bandwidth matters more

$$\frac{\#\ \text{RTTS}}{} \qquad \frac{\sim n * \text{avg. file size}}{\text{bandwidth}}$$

To limit staleness of cached objects,
HTTP enables a client to validate cached objects

Server hints when an object expires (kind of TTL)

as well as the last modified date of an object

Client conditionally requests a ressources

using the “if-modified-since” header in the HTTP request

Server compares this against “last modified” time

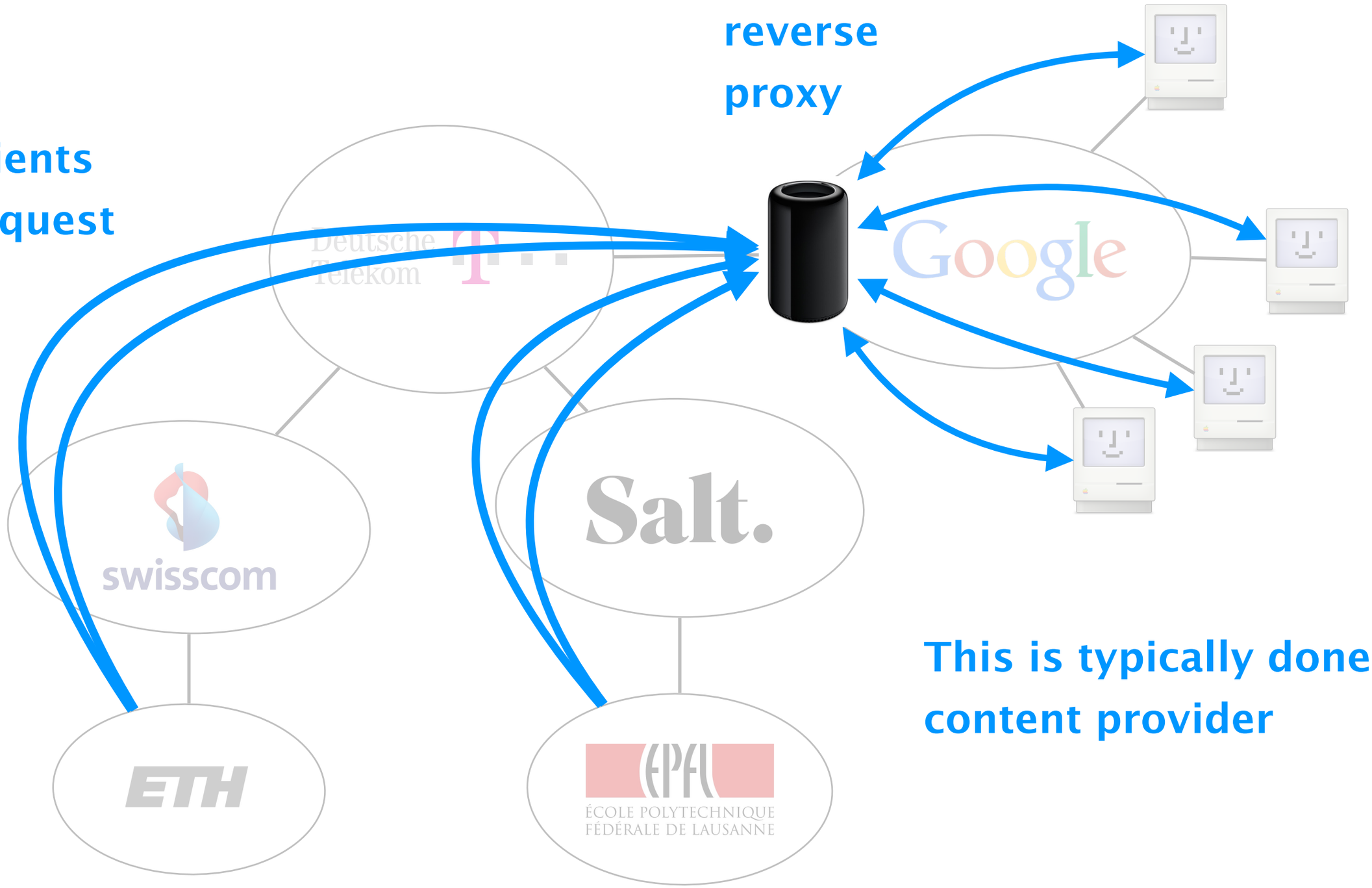
of the resource and returns:

- Not Modified if the resource has not changed
- OK with the latest version

Caching can and is performed at different locations

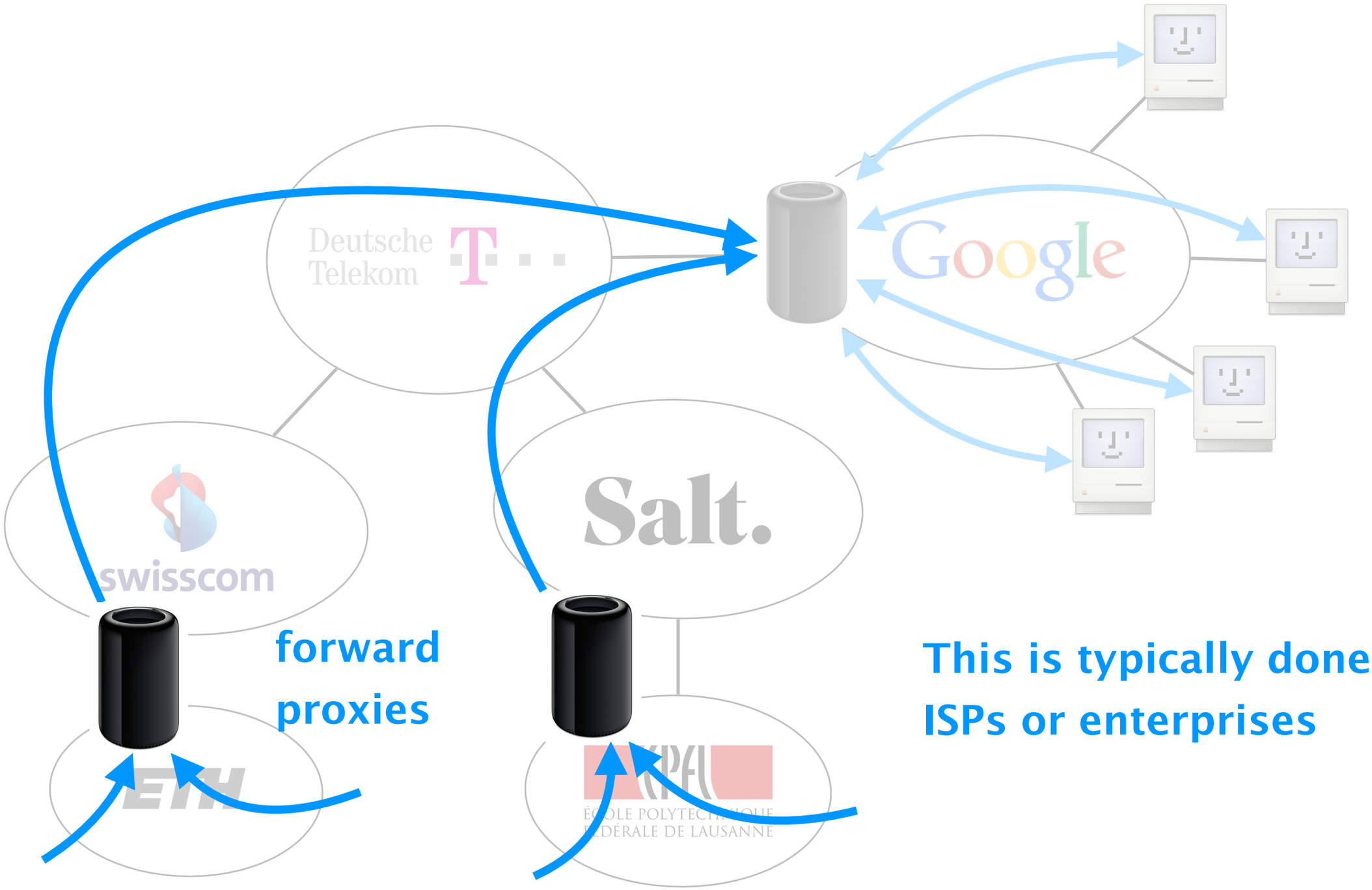
client	browser cache
close to the client	forward proxy
	Content Distribution Network (CDN)
close to the destination	reverse proxy

Reverse proxies cache documents close to servers, decreasing their load



This is typically done by content provider

Forward proxies cache documents close to clients, decreasing network traffic, server load and latencies



This is typically done by ISPs or enterprises

Web

Video Streaming

HTTP-based

The three steps behind most contemporary solutions

- Encode video in multiple bitrates
- Replicate using a content delivery network
- Video player picks bitrate adaptively
  - Estimate connection's available bandwidth
  - Pick a bitrate $\leq$ available bandwidth

Encoding

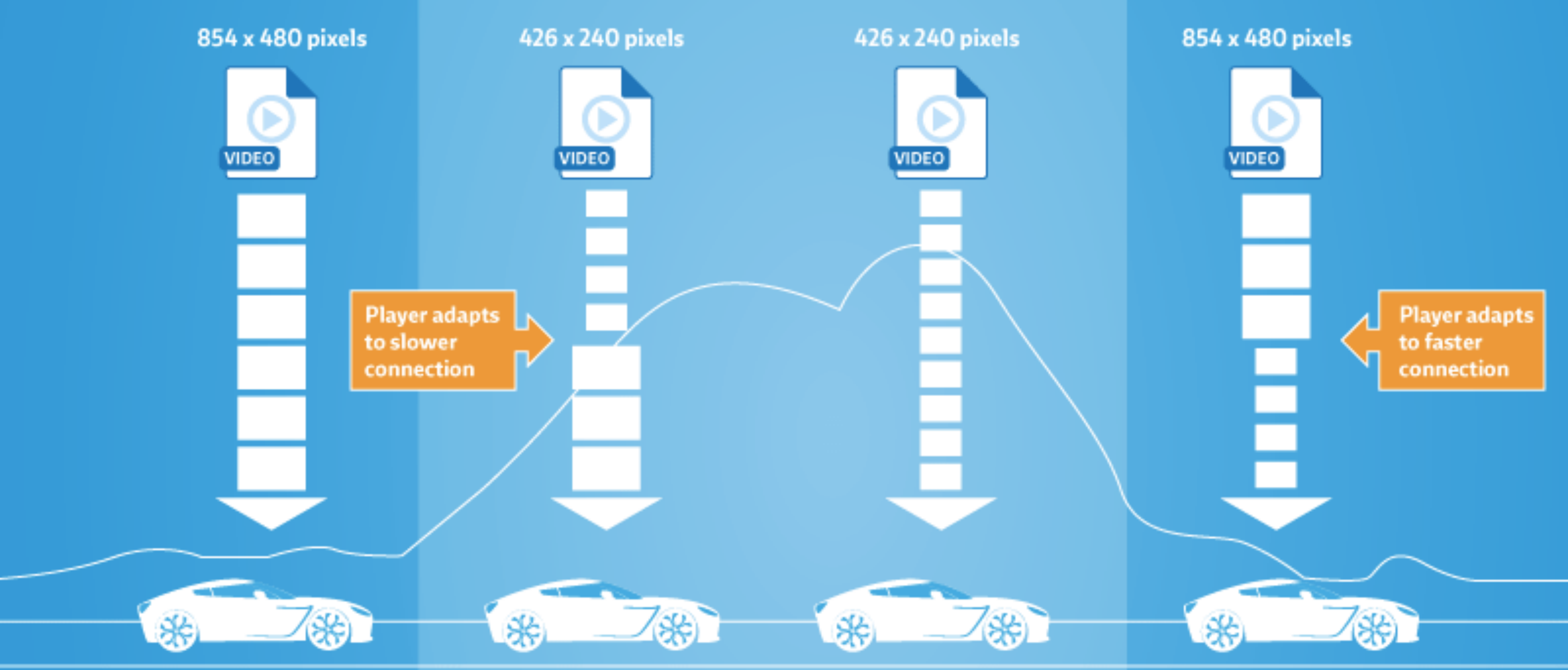
Replication

Adaptation

Encoding

Replication

Adaptation



[bitmovin.com]

Problem: this doesn't take into account the variability in the video content (slow moving vs. fast moving)

Bitrate (kbps)	Resolution
235	320x240
375	384x288
560	512x384
750	512x384
1050	640x480
1750	720x480
2350	1280x720
3000	1280x720
4300	1920x1080
5800	1920x1080

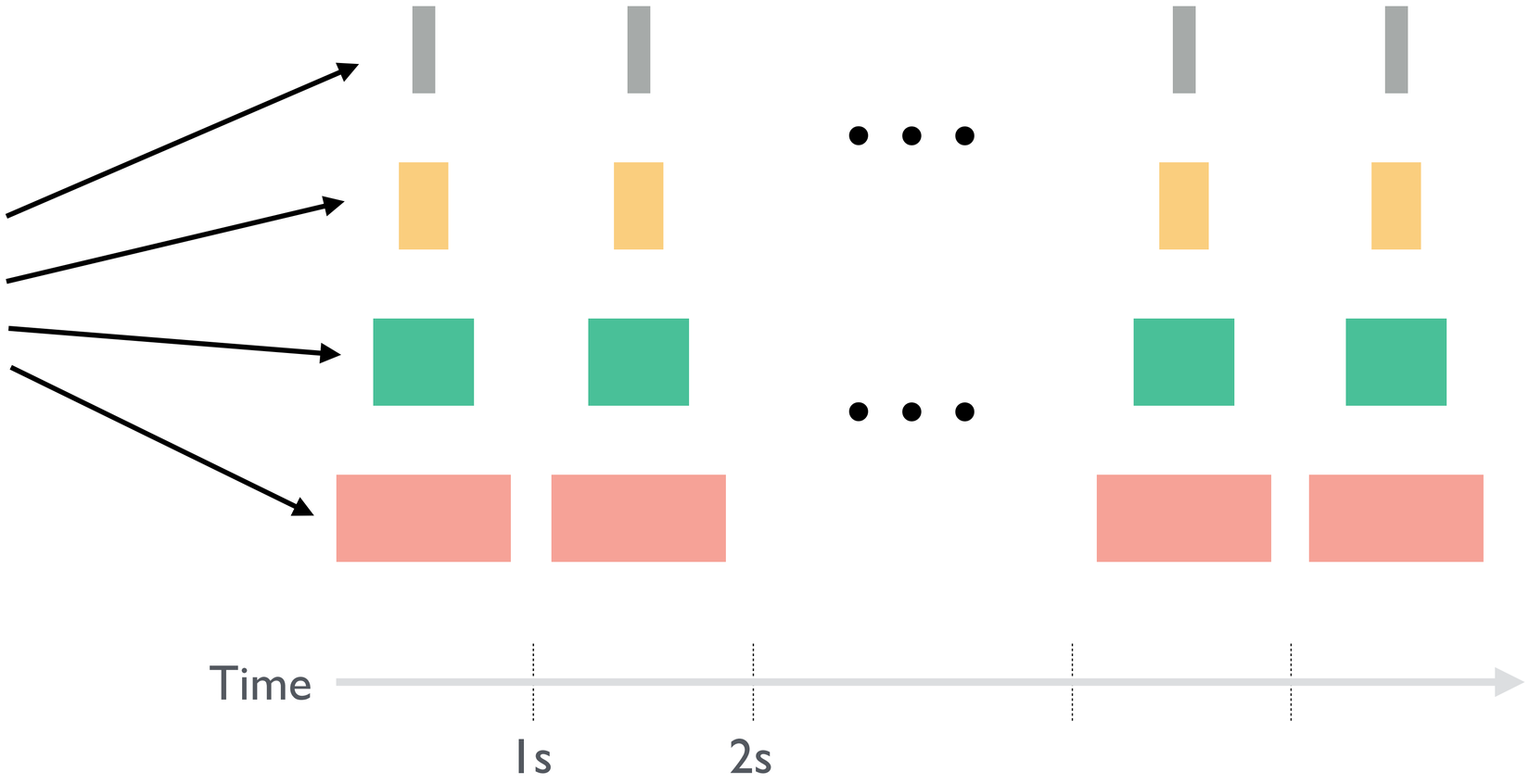
[netflix.com]

[netflix.com]

[bitmovin.com]

Your player download “chunks” of video at different bitrates

[netflix.com]



Depending on your network connectivity, your player fetches chunks of different qualities

[netflix.com]

Encoding

Replication

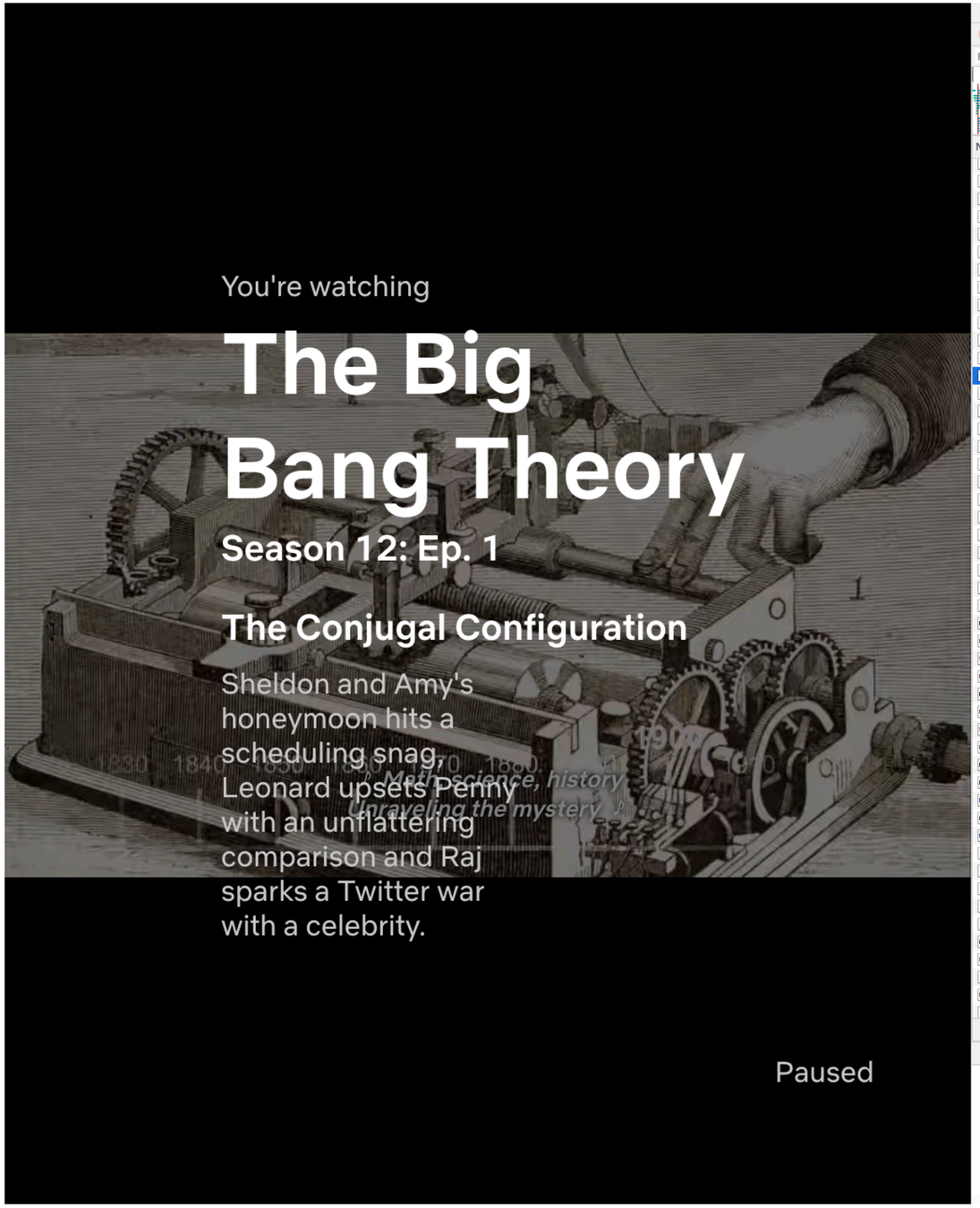
Adaptation

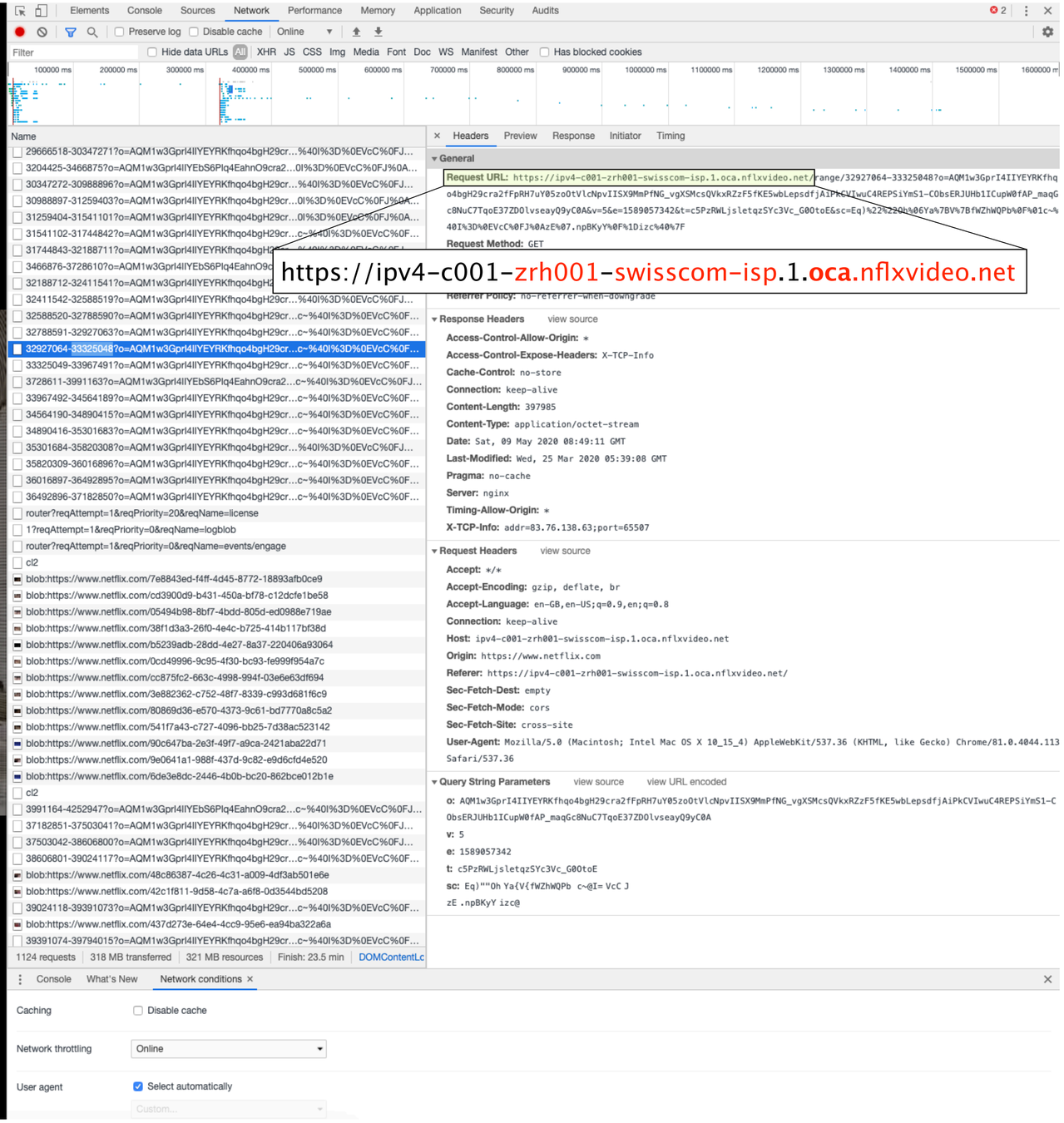
NETFLIX

Open Connect: Starting from a Greenfield (a mostly Layer 0 talk)

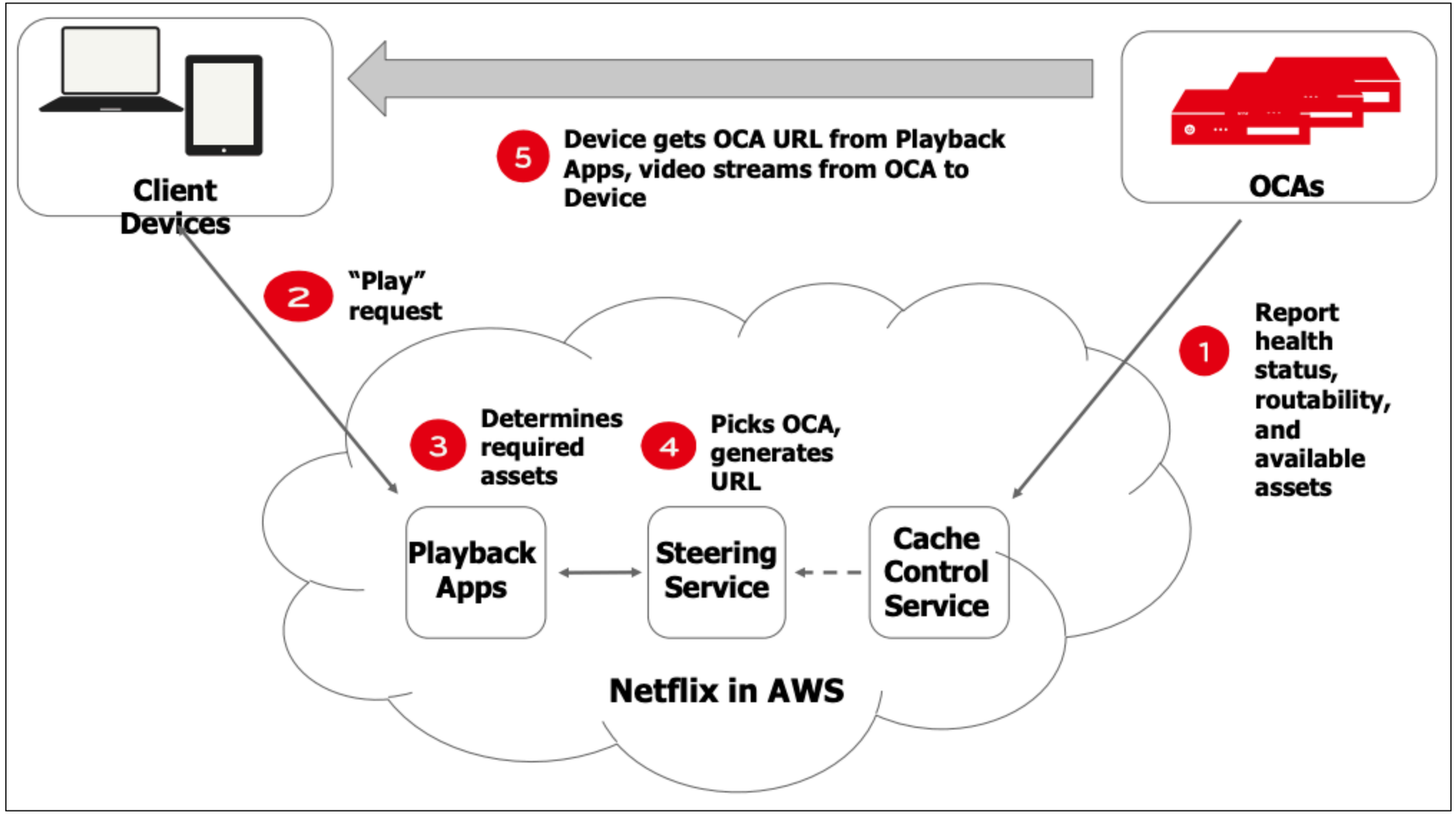
Dave Temkin

06/01/2015


You're watching
The Big Bang Theory
Season 12: Ep. 1
The Conjugal Configuration
Sheldon and Amy's honeymoon hits a scheduling snag, Leonard upsets Penny with an unflattering comparison and Raj sparks a Twitter war with a celebrity.
Paused


Elements Console Sources Network Performance Memory Application Security Audits
Preserve log Disable cache Online
Filter Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other Has blocked cookies
Name
Headers Preview Response Initiator Timing
General
Request URL: https://ipv4-c001-zrh001-swisscom-isp.1.oca.nflxvideo.net/range/32927064-33325048?o=AQM1w3GprI4IIYEYRKfhqo4bgH29cra2fFpRH7uY05zoOtVlcNpvIISX9MmPfNG_vgXSMcsQVkxRZzF5fKE5wbLepsdfjAiPkCVIwuC4REPSiYmS1-CObsERJUHb1ICupW0fAP_maqGc8NuC7TqoE37ZDOlvseayQ9yC0A&v=5&e=1589057342&t=c5PzRWLjsletqzSyc3Vc_G0OtoE&sc=Eq)%22%22Oh%06Ya%7BV%7BfWZhWQPb%0F%01c~%40I%3D%0EVcC%0FJ%0AzE%07.npBKyY%0F%1Dizc%40%7F
Request Method: GET
https://ipv4-c001-zrh001-swisscom-isp.1.oca.nflxvideo.net
Response Headers view source
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: X-TCP-Info
Cache-Control: no-store
Connection: keep-alive
Content-Length: 397985
Content-Type: application/octet-stream
Date: Sat, 09 May 2020 08:49:11 GMT
Last-Modified: Wed, 25 Mar 2020 05:39:08 GMT
Pragma: no-cache
Server: nginx
Timing-Allow-Origin: *
X-TCP-Info: addr=83.76.138.63;port=65507
Request Headers view source
Accept: */*
Accept-Encoding: gzip, deflate, br
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: keep-alive
Host: ipv4-c001-zrh001-swisscom-isp.1.oca.nflxvideo.net
Origin: https://www.netflix.com
Referer: https://ipv4-c001-zrh001-swisscom-isp.1.oca.nflxvideo.net/
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: cross-site
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.113 Safari/537.36
Query String Parameters view source view URL encoded
o: AQM1w3GprI4IIYEYRKfhqo4bgH29cra2fFpRH7uY05zoOtVlcNpvIISX9MmPfNG_vgXSMcsQVkxRZzF5fKE5wbLepsdfjAiPkCVIwuC4REPSiYmS1-CObsERJUHb1ICupW0fAP_maqGc8NuC7TqoE37ZDOlvseayQ9yC0A
v: 5
e: 1589057342
t: c5PzRWLjsletqzSyc3Vc_G0OtoE
sc: Eq)""Oh Ya{V{fWZhWQPb c~@I= VcC J
zE .npBKyY izc@
1124 requests 318 MB transferred 321 MB resources Finish: 23.5 min DOMContentLc
Console What's New Network conditions
Caching Disable cache
Network throttling Online
User agent Select automatically

Complete Playback Workflow @Netflix

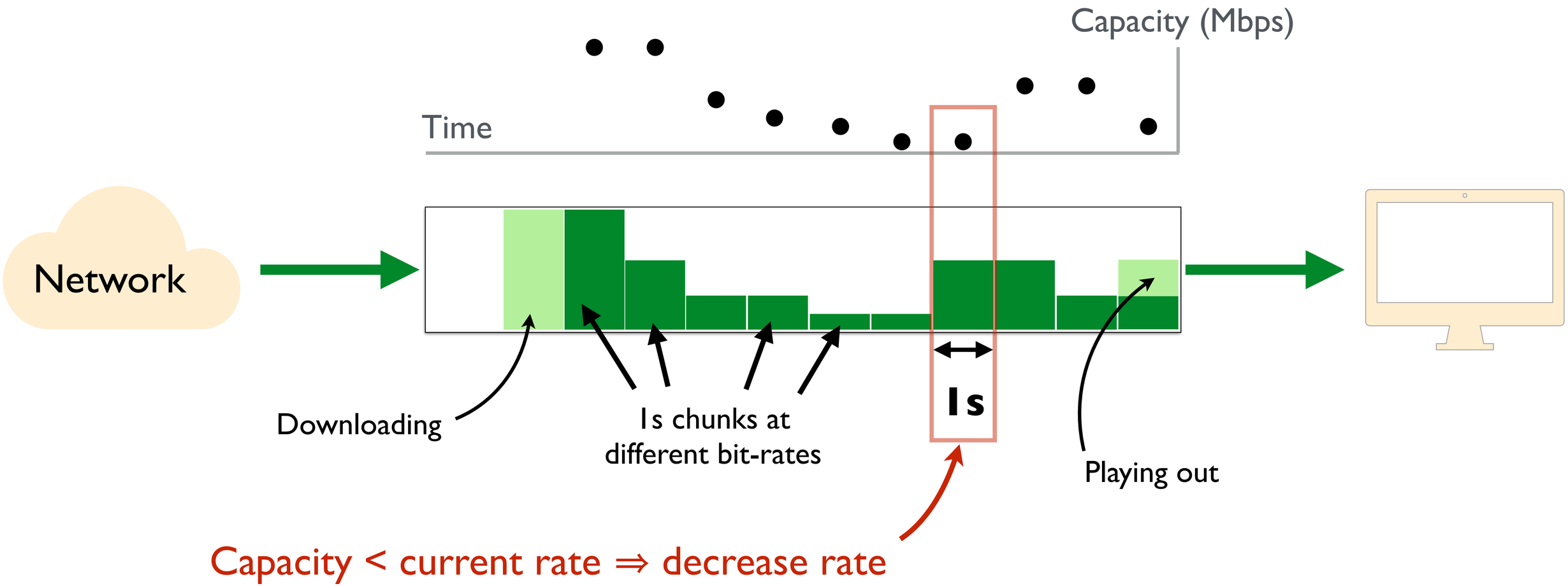


[more-ip-event.net]

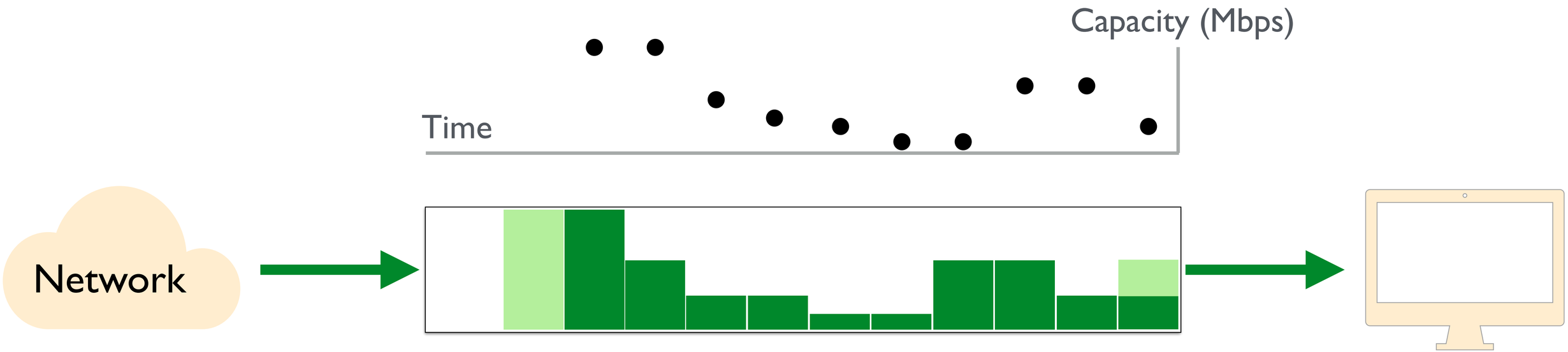
Encoding

Replication

Adaptation

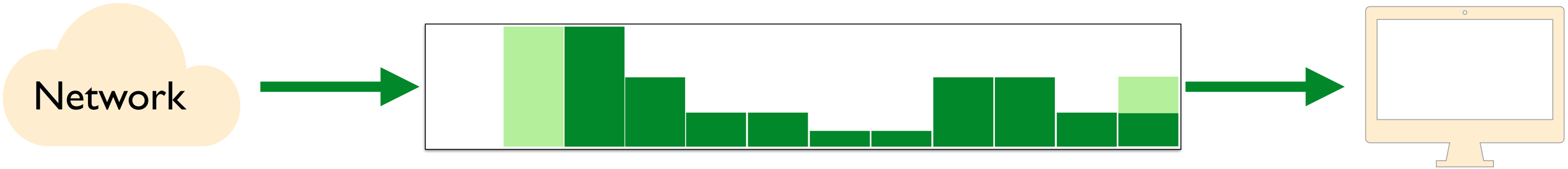

Capacity (Mbps)
Time
Network
Downloading
1s chunks at
different bit-rates
1s
Playing out
Capacity < current rate ⇒ decrease rate

Capacity estimation



Decide based on the buffer alone?

Buffer-based adaptation



Nearly full buffer ⇒ large rate

Buffer-based adaptation

Nearly empty buffer ⇒ small rate

Today on

Communication Networks

E-mail

MX, SMTP, POP, IMAP

IPv6

128-bits IPv4 addresses?

(the beginning)

Today on

Communication Networks

E-mail

MX, SMTP, POP, IMAP

IPv6

128-bits IPv4 addresses?

We'll study e-mail from three different perspectives

Content	Infrastructure/ Transmission	Retrieval
Format: Header/Content Encoding: MIME	SMTP: Simple Mail Transfer Protocol Infrastructure mail servers	POP: Post Office Protocol IMAP:Internet Message Access Protocol

Content

Infrastructure/
Transmission

Retrieval

Format: Header/Content

Encoding: MIME

An e-mail is composed of two parts

E-mail

A header, in 7-bit U.S. ASCII text

Header

From: Laurent Vanbever <lvanbever@ethz.ch>

To: Tobias Buehler <buehlert@ethz.ch>

Subject: [comm-net] Exam questions

A body, also in 7-bit U.S. ASCII text

From: Laurent Vanbever <lvanbever@ethz.ch>

To: Tobias Buehler <buehlert@ethz.ch>

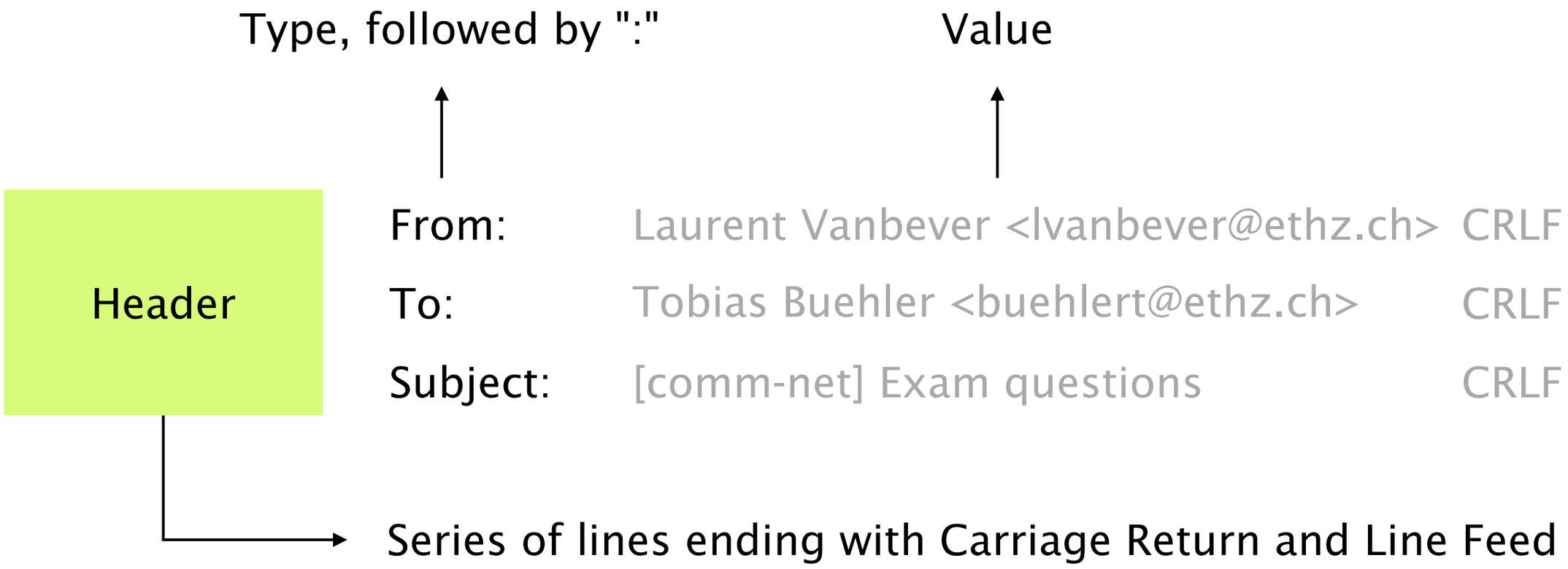
Subject: [comm-net] Exam questions

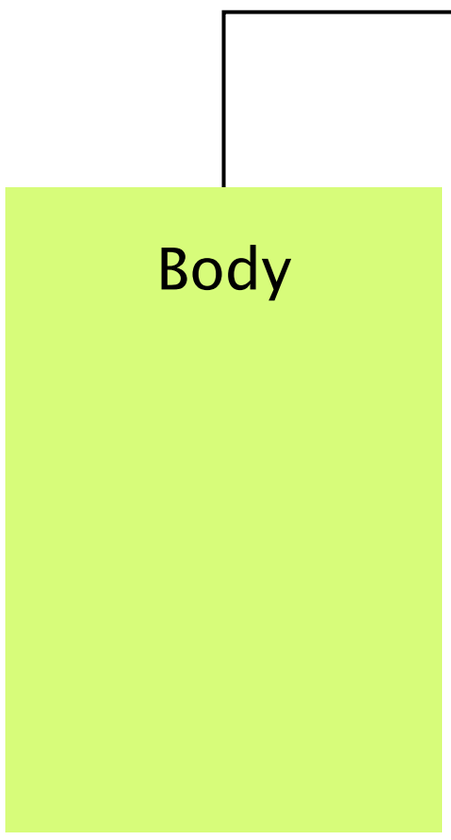
Body

Hi Tobias,

Here are some interesting questions...

Best,
Laurent


Type, followed by ":"
Value
Header
From: Laurent Vanbever <lvanbever@ethz.ch> CRLF
To: Tobias Buehler <buehlert@ethz.ch> CRLF
Subject: [comm-net] Exam questions CRLF
Series of lines ending with Carriage Return and Line Feed

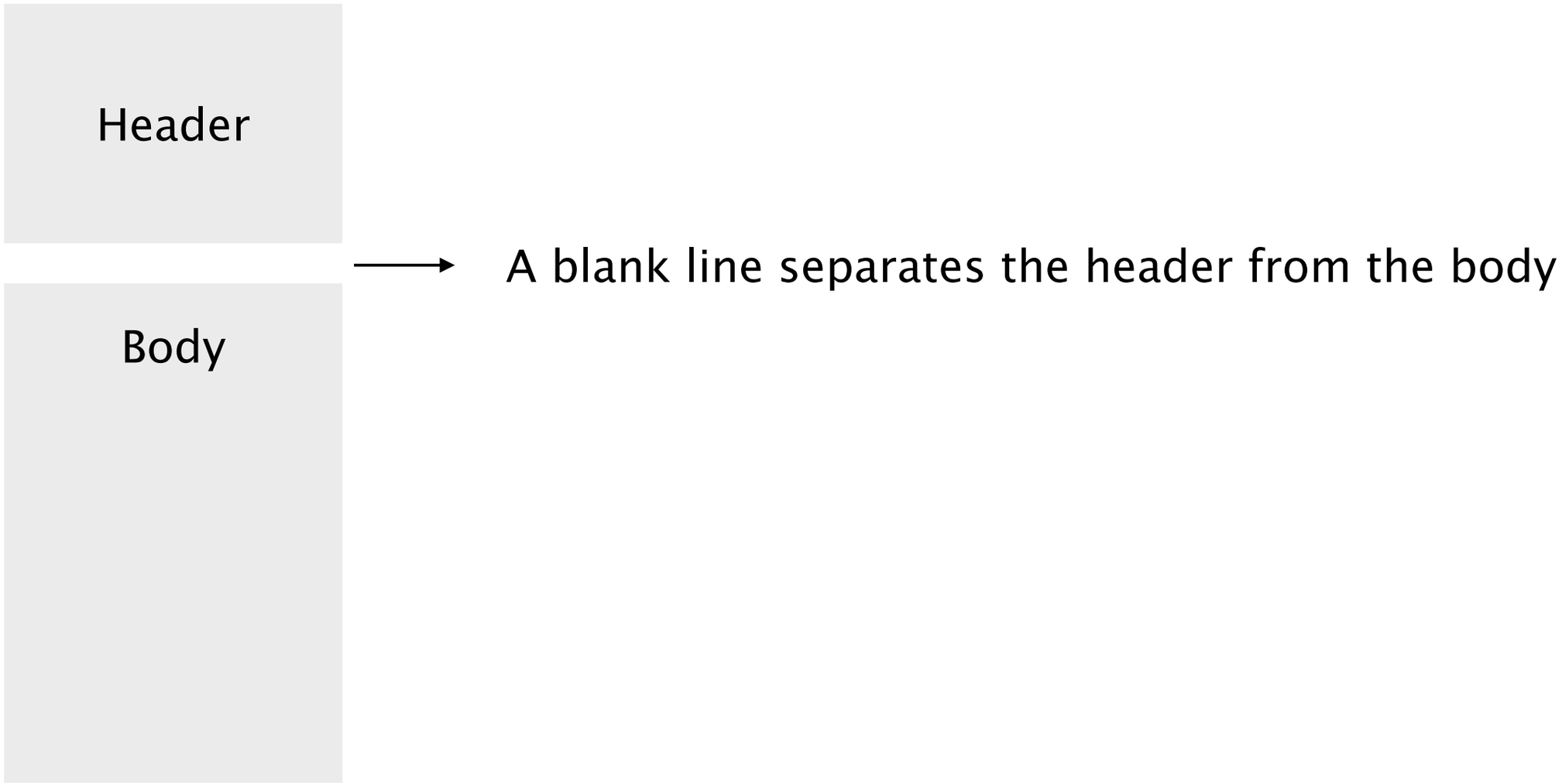


Series of lines with no structure/meaning

Hi Tobias,

Here are some interesting questions…

Best,
Laurent


Header
A blank line separates the header from the body
Body

Header

Body

.

→ A blank line separates the header from the body

→ A dot (".") on a new line ends the body

Email relies on 7-bit U.S. ASCII…

How do you send non-English text? Binary files?

Solution

Multipurpose Internet Mail Extensions

commonly known as MIME, standardized in RFC 822

MIME defines

- additional headers for the email body
- a set of content types and subtypes
- base64 to encode binary data in ASCII

MIME defines

- additional headers for the email body

MIME-Version: the version of MIME being used

Content-Type: the type of data contained in the message

Content-Transfer-Encoding: how the data is encoded

MIME defines

- additional headers for the email body
- a set of content types and subtypes

  e.g. image with subtypes gif or jpeg

  text with subtypes plain, html, and rich text

  application with subtypes postscript or msword

  multipart with subtypes mixed or alternative

The two most common types/subtypes for MIME are:
multipart/mixed and *multipart/alternative*

Content-Type	indicates that the message contains
multipart/mixed	multiple independent parts e.g. plain text *and* a binary file
multipart/alternative	multiple representation of the same content e.g. plain text *and* HTML

MIME defines

- additional headers for the email body
- a set of content types and subtypes
- base64 to encode binary data in ASCII

What kind of delimiter do we use?

multipart/mixed — **multiple** independent parts

multipart/alternative — **multiple** representation of the same content

Content-Type contains a parameter that specifies a string delimiter (chosen randomly by the client)

ensuring that the delimiter
does *not* appear in the email itself

```
From: Laurent Vanbever <lvanbever@ethz.ch>
To: Tobias Buehler <buehlert@ethz.ch>
Subject: [comm-net] Final exam
MIME-Version: 1.0
Content-Type: multipart/related;
boundary="_004_cc163051808f425a9b67b778666b785eeeethzch_";
   type="multipart/alternative"

--_004_cc163051808f425a9b67b778666b785eeeethzch_
Content-Type: multipart/alternative;
   boundary="_000_cc163051808f425a9b67b778666b785eeeethzch_"

--_000_cc163051808f425a9b67b778666b785eeeethzch_
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

Let's start the exam with ...

--_000_cc163051808f425a9b67b778666b785eeeethzch_
Content-Type: text/html; charset="utf-8"
Content-Transfer-Encoding: base64

PGh0bWwgeG1sbnM6dj0idX ...
```

MIME relies on Base64 as binary-to-text encoding scheme

Relies on 64 characters out of the 128 ASCII characters

the most common *and* printable ones, i.e. A-Z, a-z, 0-9, +, /

Divides the bytes to be encoded into sequences of 3 bytes

each group of 3 bytes is then encoded using 4 characters

Uses padding if the last sequence is partially filled

i.e. if the |sequence| to be encoded is not a multiple of 3

Binary input	`0x14fb9c03d97e`
8-bits	`00010100 11111011 10011100` `00000011 11011001 01111110`
6-bits	`000101 001111 101110 011100` `000000 111101 100101 111110`
Decimal	`5 15 46 28 0 61 37 62`
base64	`F P u c A 9 l +`

Value	Char	Value	Char	Value	Char	Value	Char
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/

If the length of the input is not a multiple of three, Base64 uses "=" as padding character

Binary input	0x14
8-bits	00010100
6-bits	000101 000000
Decimal	5 0
base64	F A = =

```
From: Laurent Vanbever <lvanbever@ethz.ch>
To: Tobias Buehler <buehlert@ethz.ch>
Subject: [comm-net] Final exam
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: multipart/mixed;
        boundary="123boundary"

This is a multipart message in MIME format.

--123boundary
Content-Type: text/plain

Hi Tobias, Please find the exam enclosed. Laurent

--123boundary
Content-Type: application/pdf;
Content-Disposition: attachment;
        filename="exam_2020.pdf"

base64 encoded data .....
.........................
......base64 encoded data
```

Content

Infrastructure/
Transmission

Retrieval

SMTP: Simple Mail
Transfer Protocol

Infrastructure
mail servers

An e-mail address is composed of two parts identifying the local mailbox and the domain

lvanbever @ ethz.ch

local mailbox

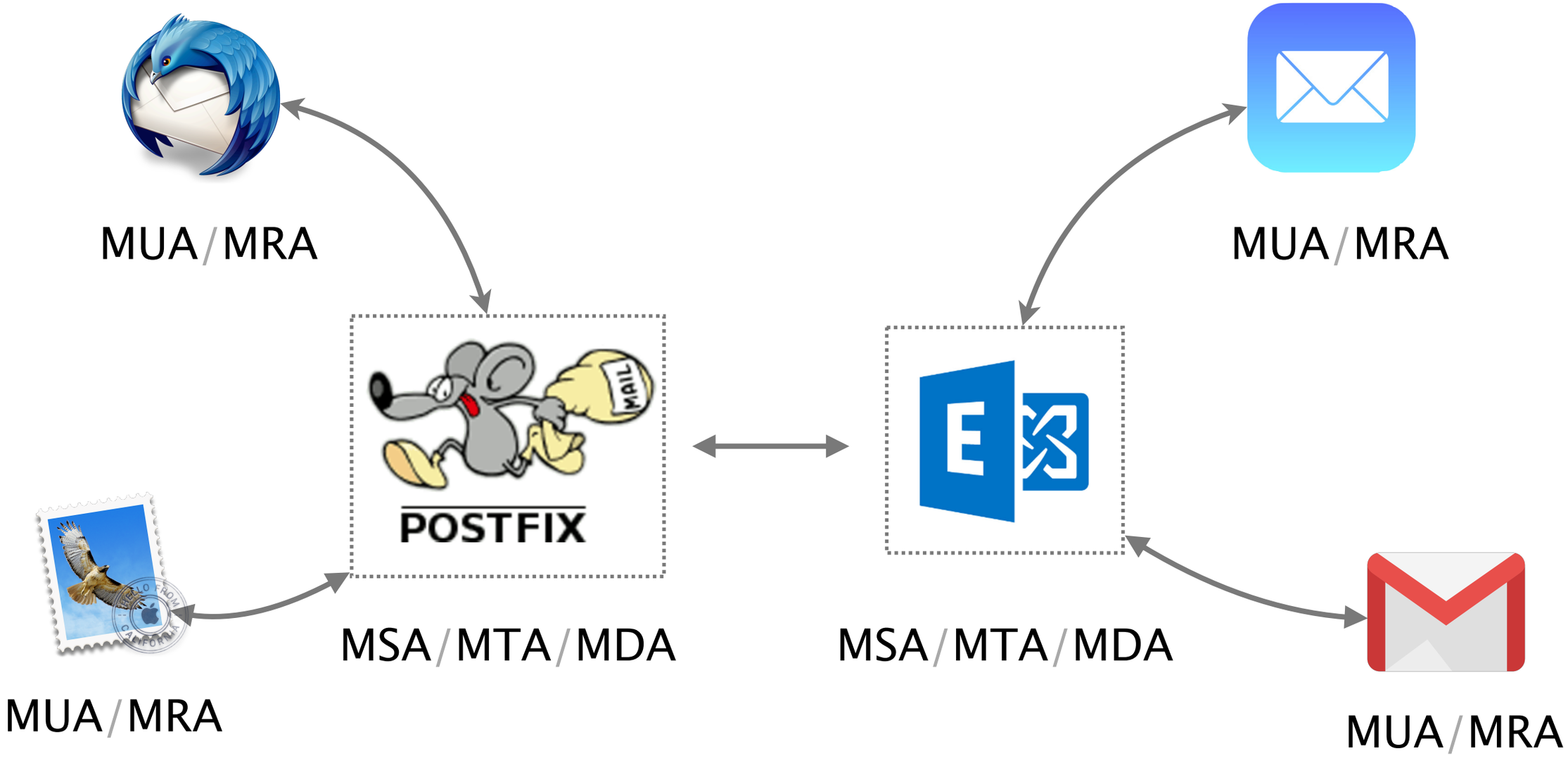
domain name

actual mail server is identified using a DNS query asking for MX records

We can divide the e-mail infrastructure into five functions

Mail	User	Agent	Use to read/write emails (mail client)
Mail	Submission	Agent	Process email and forward to local MTA
Mail	Transmission	Agent	Queues, receives, sends mail to other MTAs
Mail	Delivery	Agent	Deliver email to user mailbox
Mail	Retrieval	Agent	Fetches email from user mailbox

MSA/MTA/MDA and MRA/MUA are often packaged together leading to simpler workflows



Simple Mail Transfer Protocol (SMTP) is the current standard for transmitting e-mails

SMTP is a text-based, client-server protocol

client sends the e-mail, server receives it

SMTP uses reliable data transfer

built on top of TCP (port 25 and 465 for SSL/TLS)

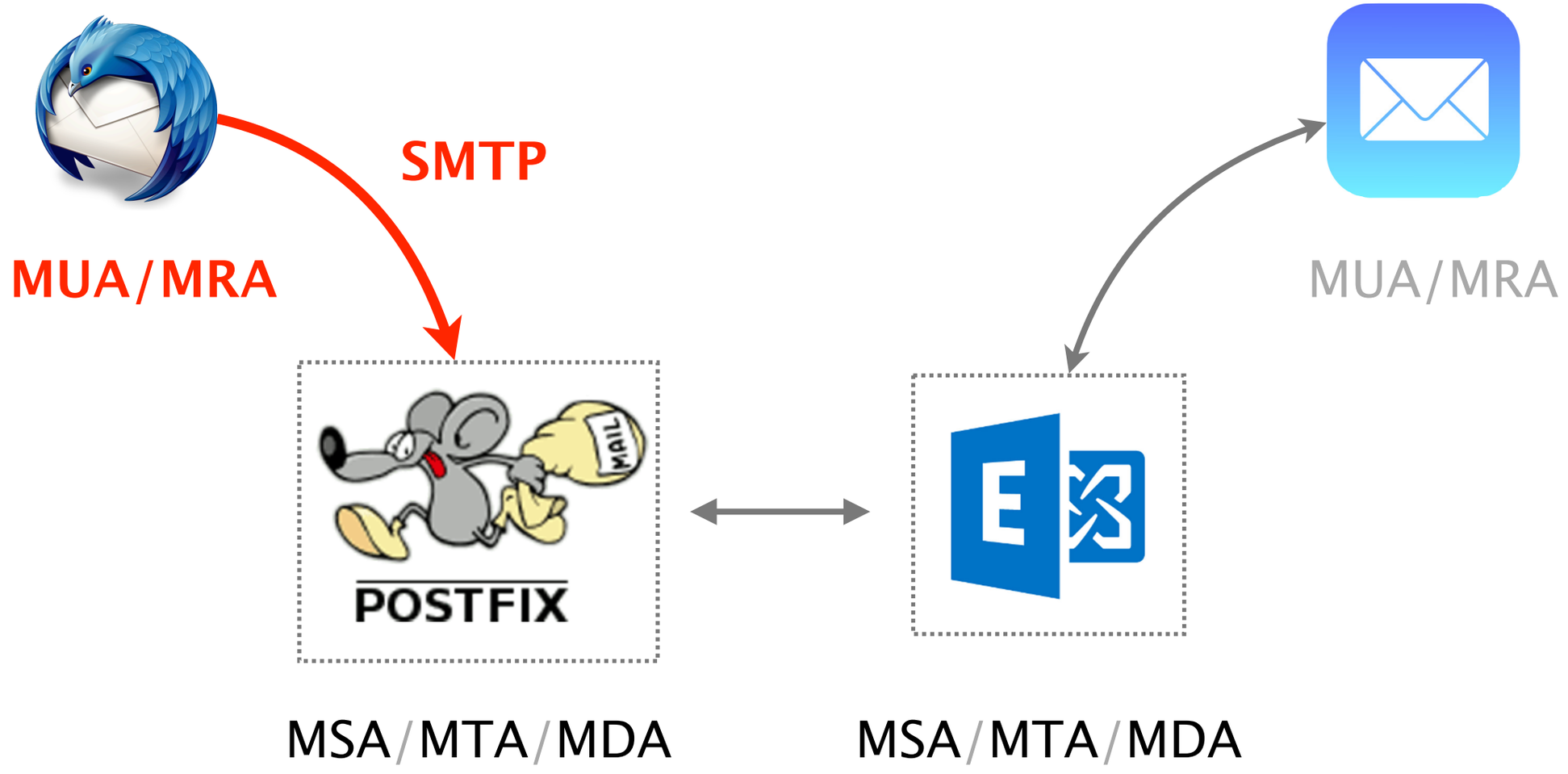
SMTP is a push-based protocol

sender pushes the file to the receiving server

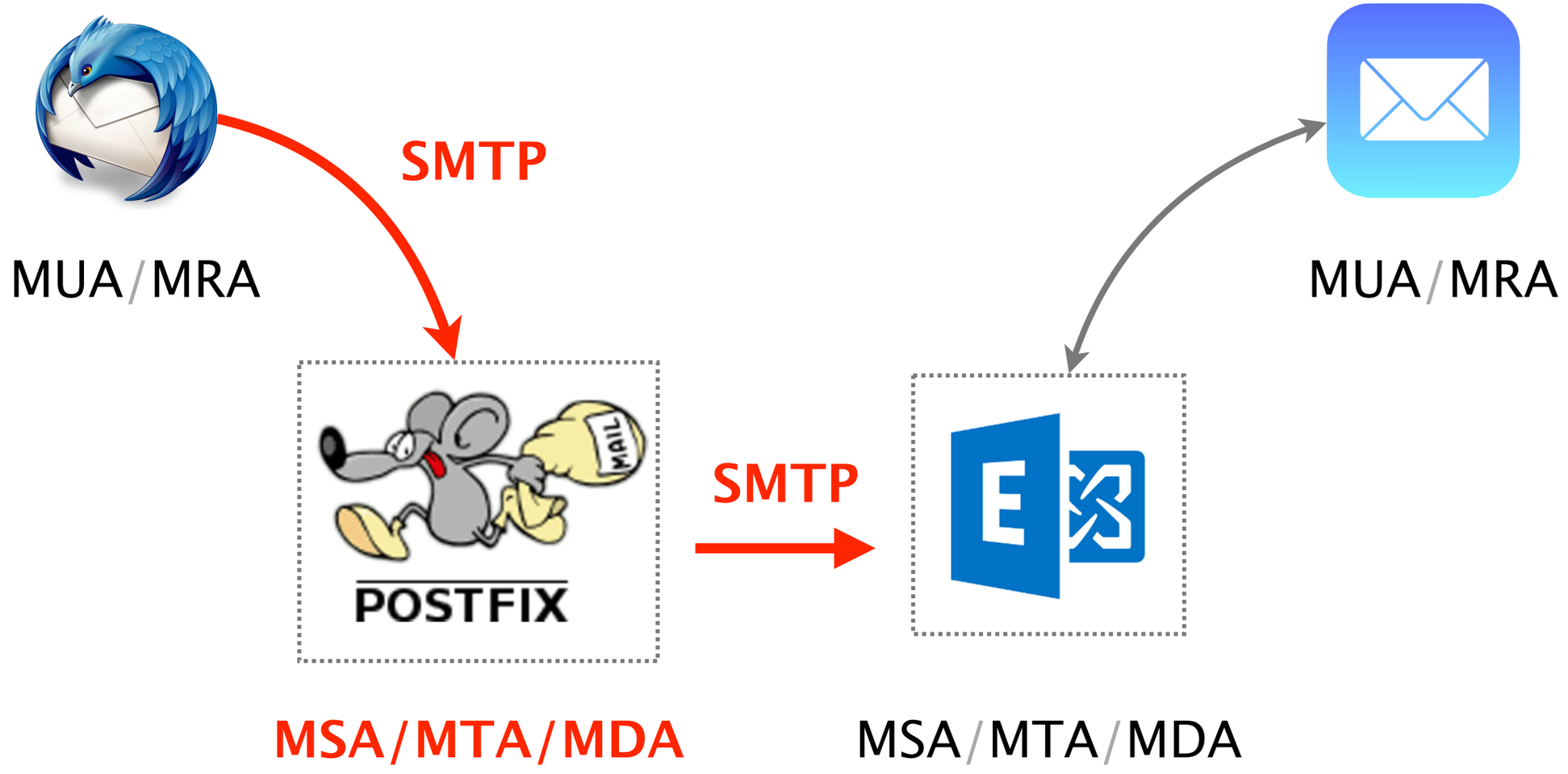
	SMTP 3 digit response code			comment
Status	2XX	success	220	Service ready
			250	Requested mail action completed
	3XX	input needed	354	Start mail input
	4XX	transient error	421	Service not available
			450	Mailbox unavailable
			452	Insufficient space
	5XX	permanent error	500	Syntax error
			502	Unknown command
			503	Bad sequence

```
server ── 220 hamburger.edu
          EHLO crepes.fr
          250  Hello crepes.fr, pleased to meet you
client ── MAIL FROM: <alice@crepes.fr>
          250 alice@crepes.fr... Sender ok
          RCPT TO: <bob@hamburger.edu>
          250 bob@hamburger.edu ... Recipient ok
          DATA
          354 Enter mail, end with "." on a line by
          itself
          Do you like ketchup?
          How about pickles?
          .
          250 Message accepted for delivery
          QUIT
          221 hamburger.edu closing connection
```

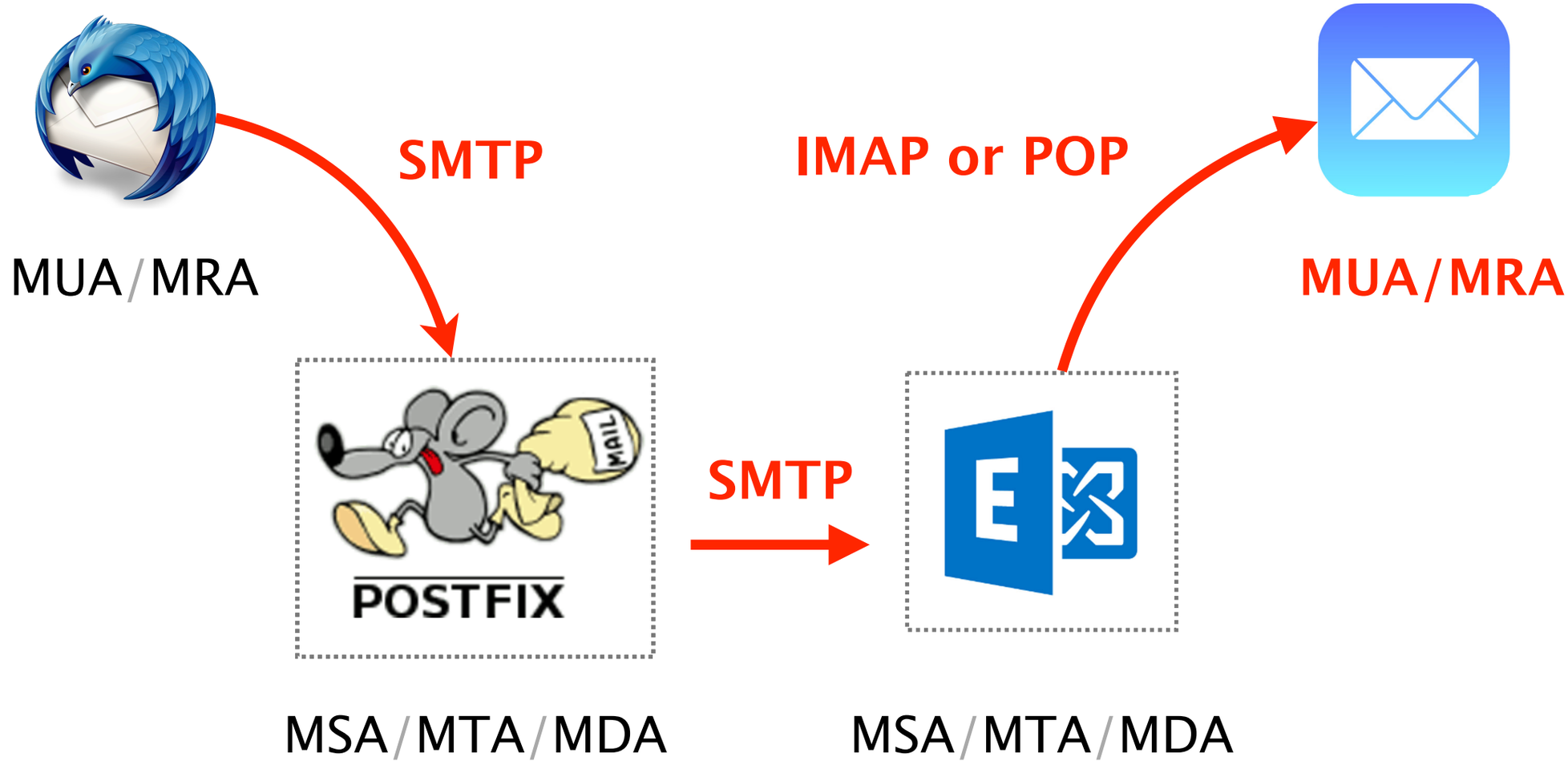
The sender MUA uses SMTP to transmit the e-mail to a local MTA (e.g. mail.ethz.ch, gmail.com, hotmail.com)



The local MTA then looks up the MTA of the recipient domain (DNS MX) and transmits the e-mail further



Once the e-mail is stored at the recipient domain,
IMAP or POP is used to retrieve it by the recipient MUA



E-mails typically go through at least 2 SMTP servers,
but often way more

sending and receiving sides

Each SMTP server/MTA hop adds its identity to the e-mail header by prepending a "Received" entry

8 Received: from edge20.ethz.ch (82.130.99.26) by CAS10.d.ethz.ch (172.31.38.210) with Microsoft SMTP Server (TLS) id 14.3.361.1; Fri, 23 Feb 2018 01:48:56 +0100

7 Received: from phil4.ethz.ch (129.132.183.133) by edge20.ethz.ch (82.130.99.26) with Microsoft SMTP Server id 14.3.361.1; Fri, 23 Feb 2018 01:48:57 +0100

6 Received: from outprodmail02.cc.columbia.edu ([128.59.72.51]) by phil4.ethz.ch with esmtps (TLSv1:AES256-SHA:256) (Exim 4.69) (envelope-from <ethan@ee.columbia.edu>) id 1ep1Xg-0002s3-FH for lvanbever@ethz.ch; Fri, 23 Feb 2018 01:48:55 +0100

5 Received: from hazelnut (hazelnut.cc.columbia.edu [128.59.213.250]) by outprodmail02.cc.columbia.edu (8.14.4/8.14.4) with ESMTP id w1N0iAu4026008 for <lvanbever@ethz.ch>; Thu, 22 Feb 2018 19:48:51 -0500

4 Received: from hazelnut (localhost.localdomain [127.0.0.1]) by hazelnut (Postfix) with ESMTP id 421126D for <lvanbever@ethz.ch>; Thu, 22 Feb 2018 19:48:52 -0500 (EST)

3 Received: from sendprodmail01.cc.columbia.edu (sendprodmail01.cc.columbia.edu [128.59.72.13]) by hazelnut (Postfix) with ESMTP id 211526D for <lvanbever@ethz.ch>; Thu, 22 Feb 2018 19:48:52 -0500 (EST)

2 Received: from mail-pl0-f43.google.com (mail-pl0-f43.google.com [209.85.160.43]) (user=ebk2141 mech=PLAIN bits=0) by sendprodmail01.cc.columbia.edu (8.14.4/8.14.4) with ESMTP id w1N0mnlx052337 (version=TLSv1/SSLv3 cipher=AES128-GCM-SHA256 bits=128 verify=NOT) for <lvanbever@ethz.ch>; Thu, 22 Feb 2018 19:48:50 -0500

1 Received: by mail-pl0-f43.google.com with SMTP id u13so3927207plq.1 for <lvanbever@ethz.ch>; Thu, 22 Feb 2018 16:48:50 -0800 (PST)

E-mails typically go through at least 2 SMTP servers, but often way more

Separate SMTP servers for separate functions

SPAM filtering, virus scanning, data leak prevention, etc.

Separate SMTP servers that redirect messages

e.g. from lvanbever@tik.ee.ethz.ch to lvanbever@ethz.ch

Separate SMTP servers to handle mailing-list

mail is delivered to the list server and then expanded

Try it out yourself!

SMTP-MTA

plaintext (!),
hard to find

```
telnet server_name 25
```

SMTP-MSA

rely on TLS
encryption

```
openssl s_client -starttls smtp
   -connect mail.ethz.ch:587
   -crlf -ign_eof (*)
```

authentication
required

```
perl -MMIME::Base64 -e 'print encode_base64("username");'
perl -MMIME::Base64 -e 'print encode_base64("password");'
```

(*) https://www.ndchost.com/wiki/mail/test-smtp-auth-telnet

As with most of the key Internet protocols, security is an afterthought

SMTP Headers

MAIL FROM: no checks are done to verify that the sending MTA is authorized to send e-mails on behalf of that address

Email content (DATA)

From: no checks are done to verify that the sending system is authorized to send e-mail on behalf of that address

Reply-to: ditto

In short, *none* of the addresses in an email are typically reliable

Let's spoof some e-mails!

(don't try this at home)

And, as usual, multiple countermeasures have been proposed with various level of deployment success

Example*

Sender Policy Framework (SPF)

Enables a domain to explicitly authorize a set of hosts that are allowed to send emails using their domain names in "MAIL FROM".

How? using a DNS TXT resource record

look for "v=spf1" in the results of "dig TXT google.com"

* if you are interested, also check out Sender ID, DKIM, and DMARC

Content

Infrastructure/
Transmission

Retrieval

POP: Post Office Protocol

IMAP: Internet Message
Access Protocol

Content

Infrastructure/
Transmission

Retrieval

POP: Post Office Protocol

IMAP: Internet Message
Access Protocol

POP is a simple protocol which was designed to support users with intermittent network connectivity

POP enables e-mail users to

- retrieve e-mails locally — when connected
- view/manipulate e-mails — when disconnected

and that's pretty much it…

Example

POP server ——— +OK POP3 server ready
user bob
+OK
client ——— pass hungry
+OK user successfully logged on

list
1 498
2 912
.
retr 1
<message 1 contents>
.
dele 1
retr 2
<message 1 contents>
.
dele 2
quit
+OK POP3 server signing off

Authorization phase

Clients declares username

password

Server answers +OK/-ERR

```
+OK POP3 server ready
user bob
+OK
pass hungry
+OK user successfully logged on
```

```
list
1 498
2 912
.
retr 1
<message 1 contents>
.
dele 1
retr 2
<message 1 contents>
.
dele 2
quit
+OK POP3 server signing off
```

Transaction phase

`list`	get message numbers
`retr`	retrieve message X
`dele`	delete message X
`quit`	exit session

```
+OK POP3 server ready
user bob
+OK
pass hungry
+OK user successfully logged on

list
1 498
2 912
.
retr 1
<message 1 contents>
.
dele 1
retr 2
<message 1 contents>
.
dele 2
quit
+OK POP3 server signing off
```

POP is heavily limited. Among others, it does not go well with multiple clients or always-on connectivity

Cannot deal with multiple mailboxes

designed to put incoming emails in one folder

Not designed to keep messages on the server

designed to download messages to the client

Poor handling of multiple-client access

while many (most?) users have now multiple devices

Content

Infrastructure/
Transmission

Retrieval

POP: Post Office Protocol

IMAP: Internet Message
Access Protocol

Unlike POP, Internet Message Access Protocol (IMAP) was designed with multiple clients in mind

Support multiple mailboxes and searches on the server

client can create, rename, move mailboxes & search on server

Access to individual MIME parts and partial fetch

client can download only the text content of an e-mail

Support multiple clients connected to one mailbox

server keep state about each message (e.g. read, replied to)

Today on

Communication Networks

E-mail

MX, SMTP, POP, IMAP

IPv6

128-bits IPv4 addresses?

(the beginning)

The long way from…

World population: 7.8 billion

~0.6 IPv4 addresses per person

...to...

Average # of atoms in a human: 6.10^{27}

~7.5 IPv6 addresses per "human" atom

Let's look at some history first

late 1980s	Exponential growth of the Internet
1992	Most class B networks have been assigned experts warn that IPv4 addresses might run out
1993	Introduction of classless IPv4 addresses
1994	"Address Allocation for Private Internets" 3 reserved IPv4 blocks for private networks Hosts in private IP space are unreachable from Internet

IPv6 originally appeared in 1998,

more than 20 years ago

1994 (cont'd)	"IP Network Address Translator (NAT)" A public address is mapped to an entire private IP space
1998	IETF standardization of the IPv6 draft
2005	Estimated timeframe for massive adaption of IPv6 Did not happen…
2008	It is possible to resolve domain names using IPv6 only

IPv6 and is *finally* picking up steam

2011 — Last unassigned top-level IPv4 block is distributed

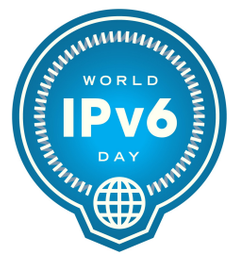
All major operating systems have stable IPv6 support

Support for mobile devices varies

2012 — World IPv6 Launch day

A large number of content and ISPs

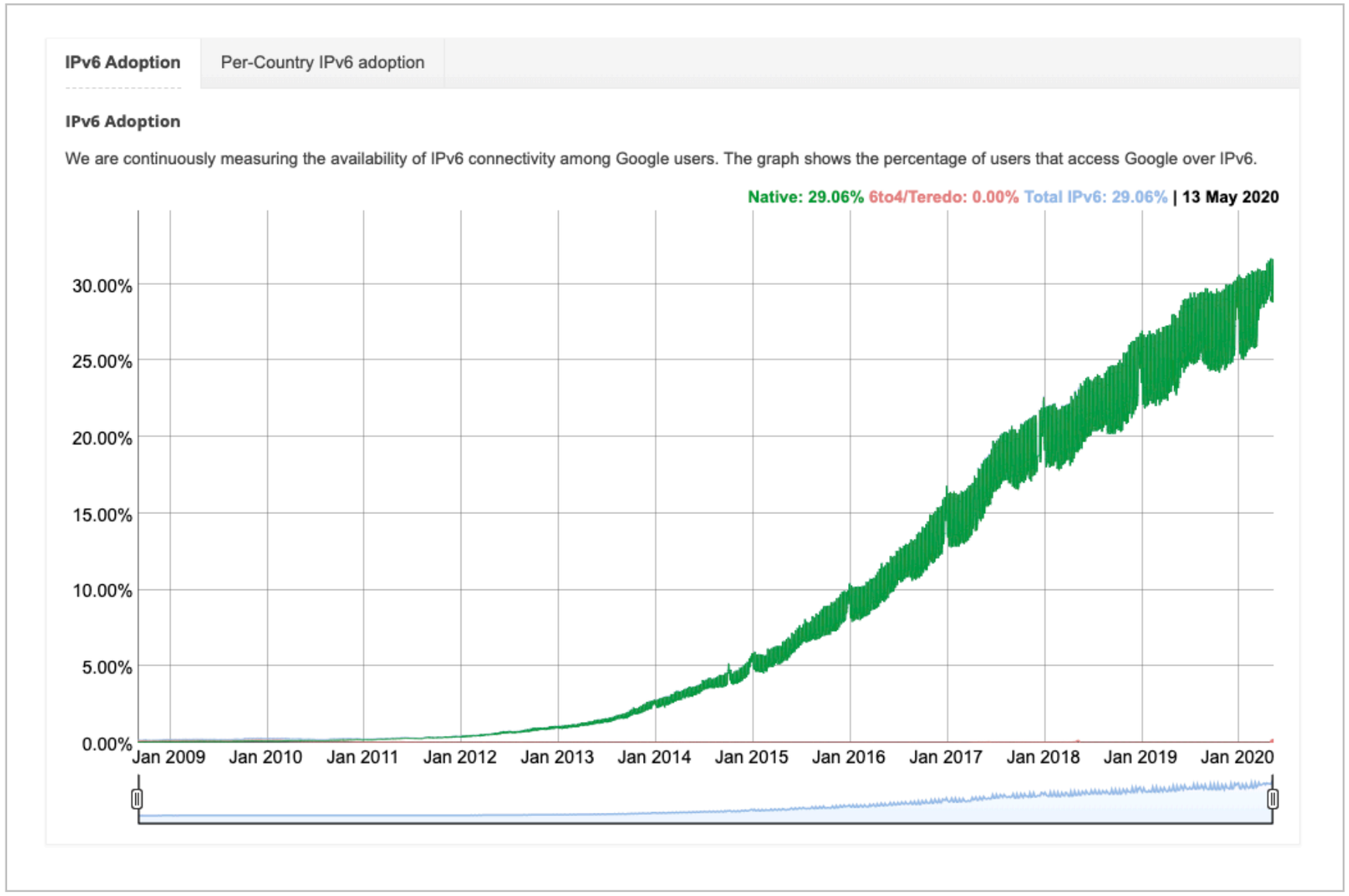
permanently enable IPv6



2020 — ~30% of Google traffic is on IPv6

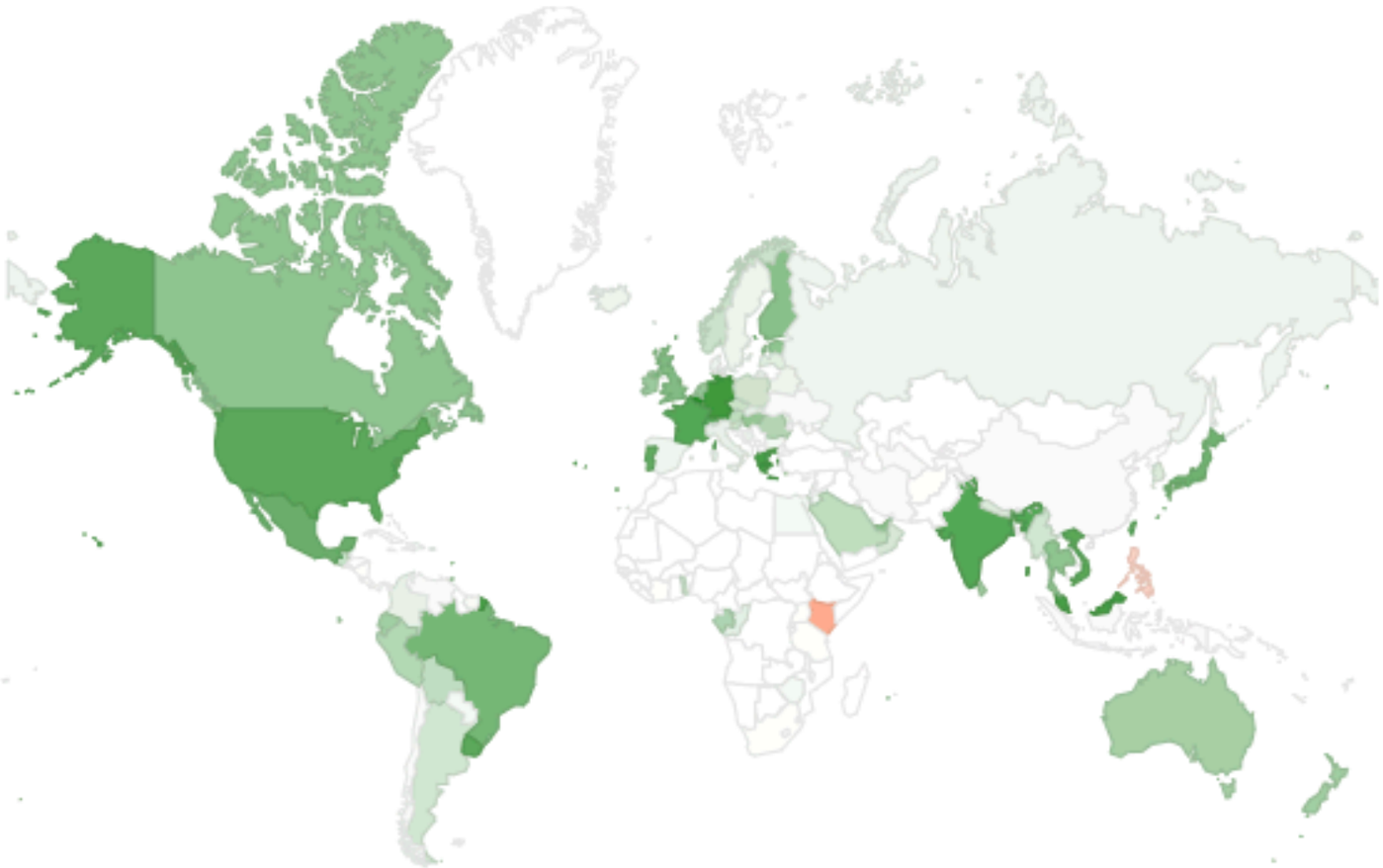
with wide differences across countries

Today, ~30% of the Google users access it using IPv6



https://www.google.com/intl/en/ipv6/statistics.html#tab=ipv6-adoption

Yet, there still exists wide discrepancy across countries



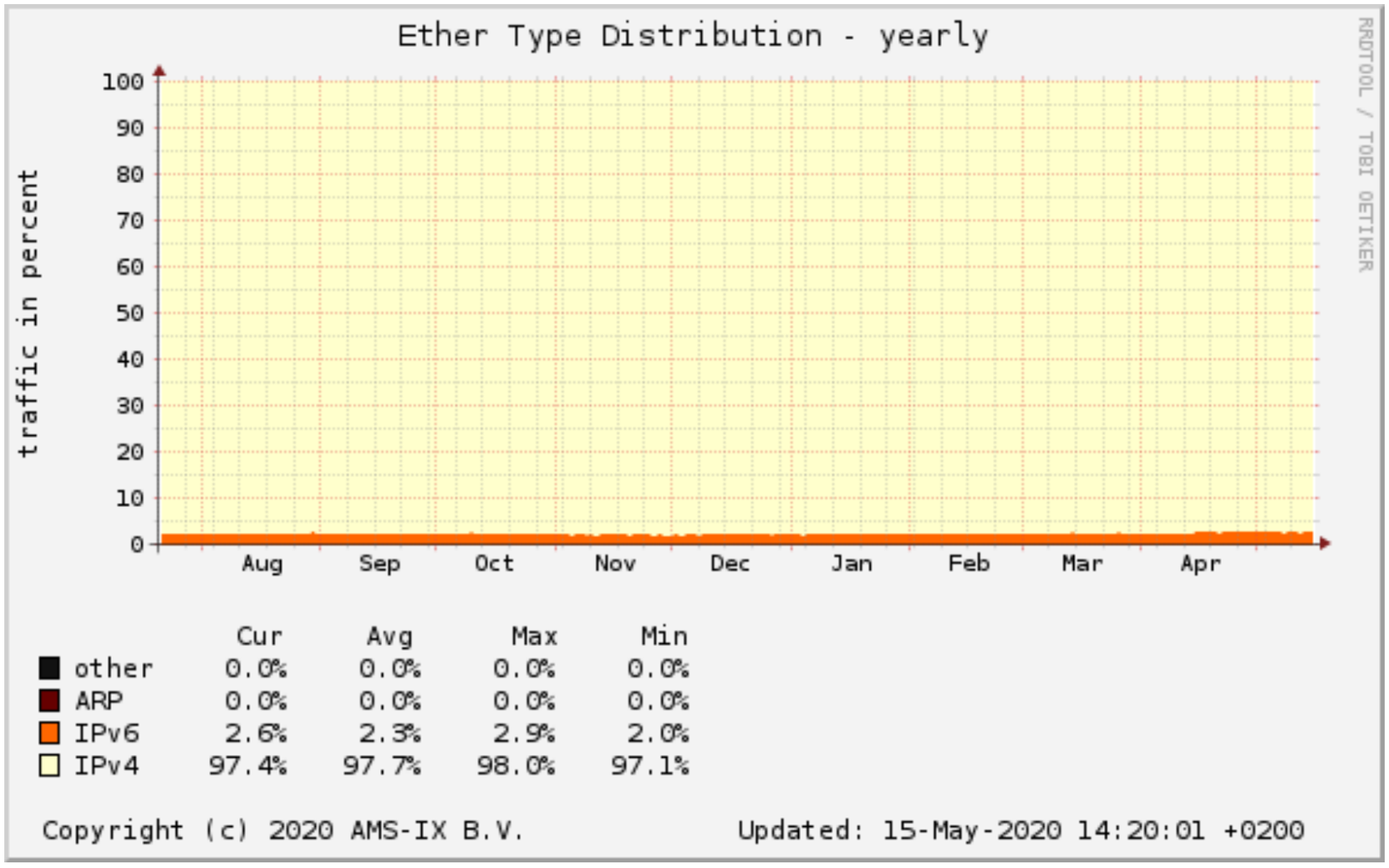
The darker the green,
the larger the deployment

Belgium 🇧🇪
(56.59% adoption, leader)

Switzerland 🇨🇭
(40.74% adoption)

https://www.google.com/intl/en/ipv6/statistics.html#tab=per-country-ipv6-adoption

Looking at AMS-IX traffic statistics,
we see that less than 3% of it is v6



https://stats.ams-ix.net/sflow/ether_type.html

IPv4 has been very persistent,
and for good reasons

Deploying IPv6 require every device to support it

All routers, middleboxes, end hosts, applications, …

Most of IPv6 new features were back-ported to IPv4

No obvious advantage in using IPv6

Network Address Translation is working well

The pain of address depletion is not obvious

Network Address Translation (NAT)

Sharing a single (public) address between hosts

Port numbers (transport layer) used to multiplex

One of the main reasons why we can still use IPv4

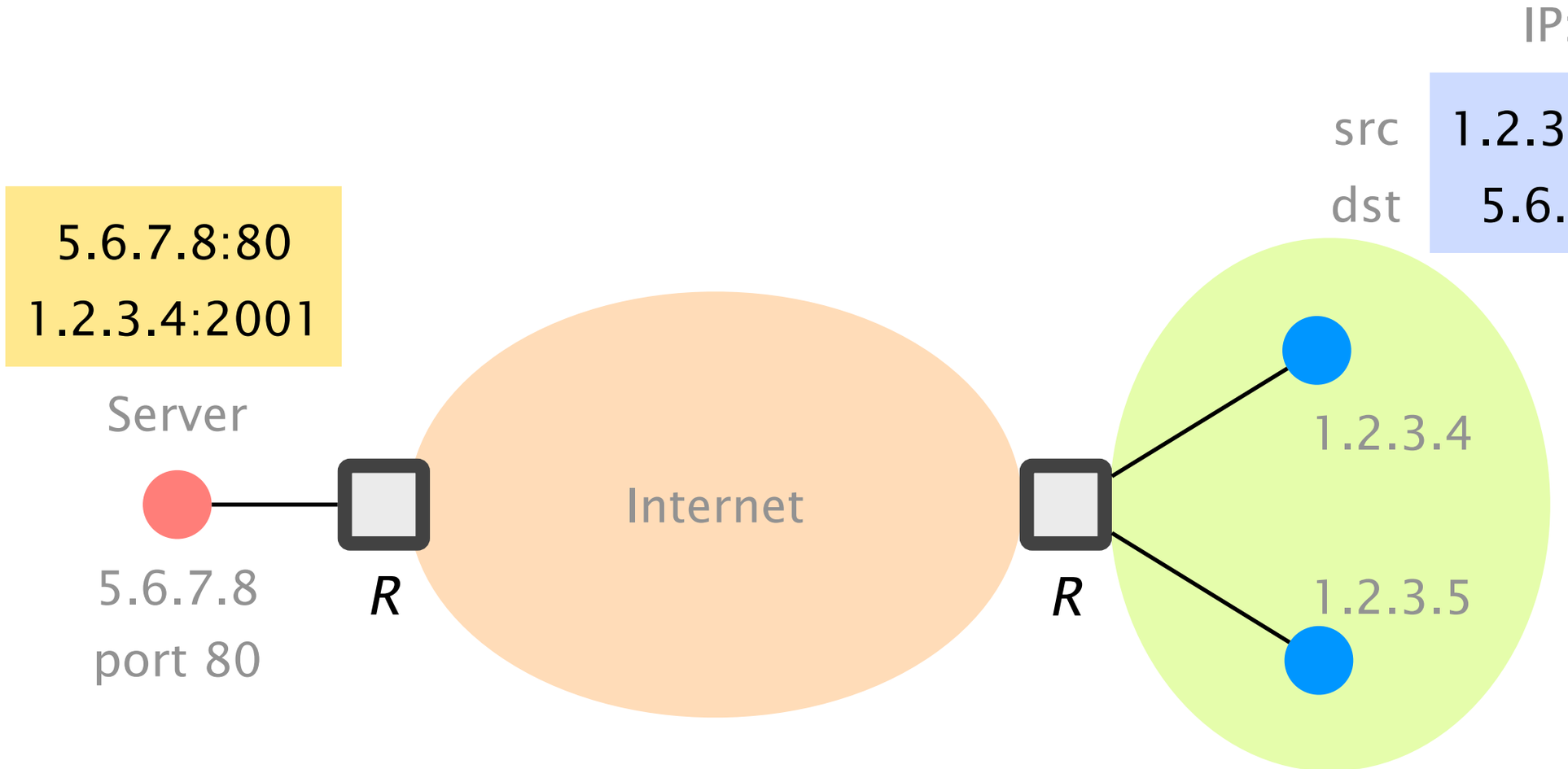
Saved us from address depletion

Violates the general end-to-end principle of the Internet

A NAT box adds a layer of indirection

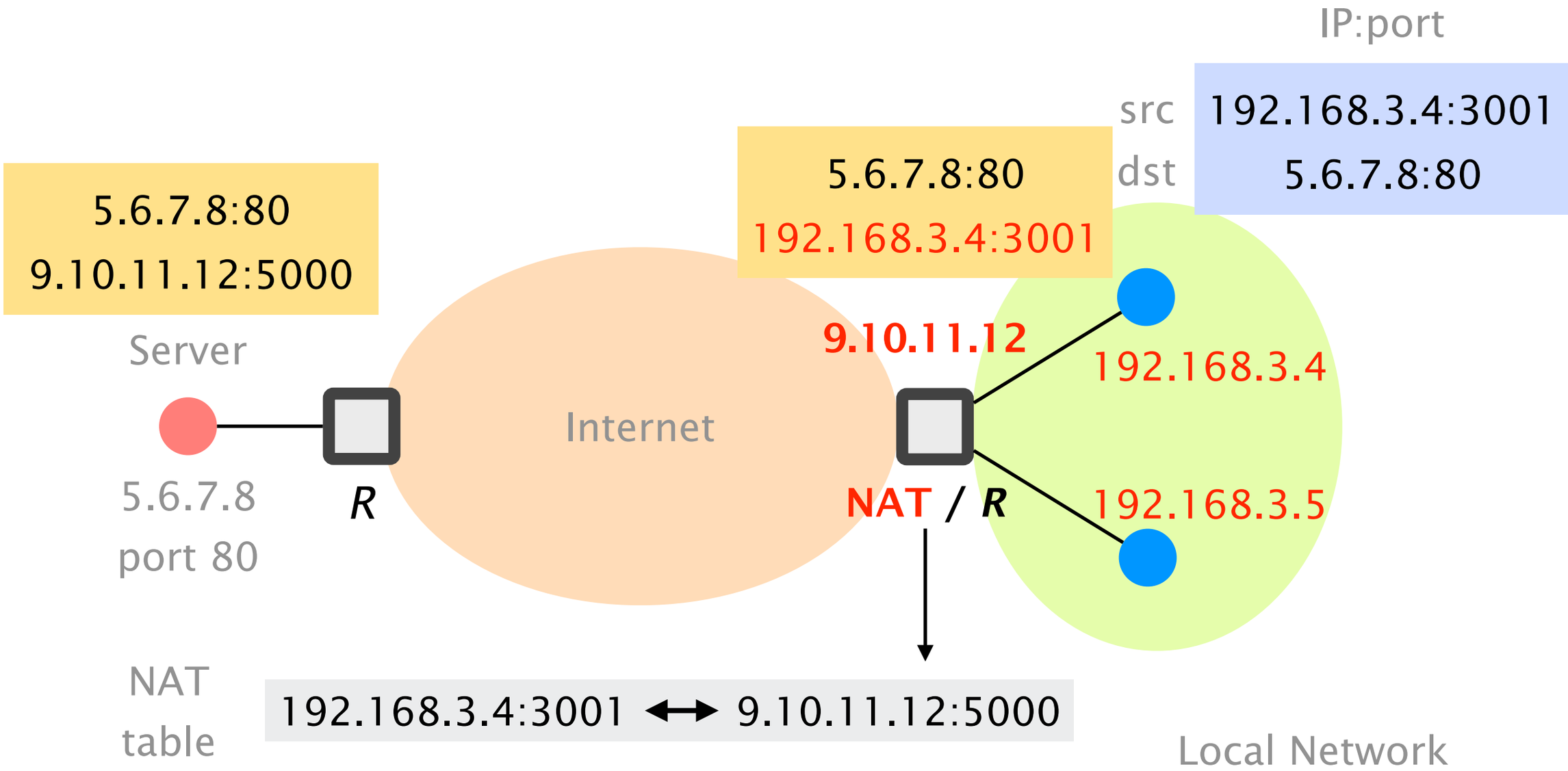
The Internet before NAT

Every machine connected to the Internet had a unique IP



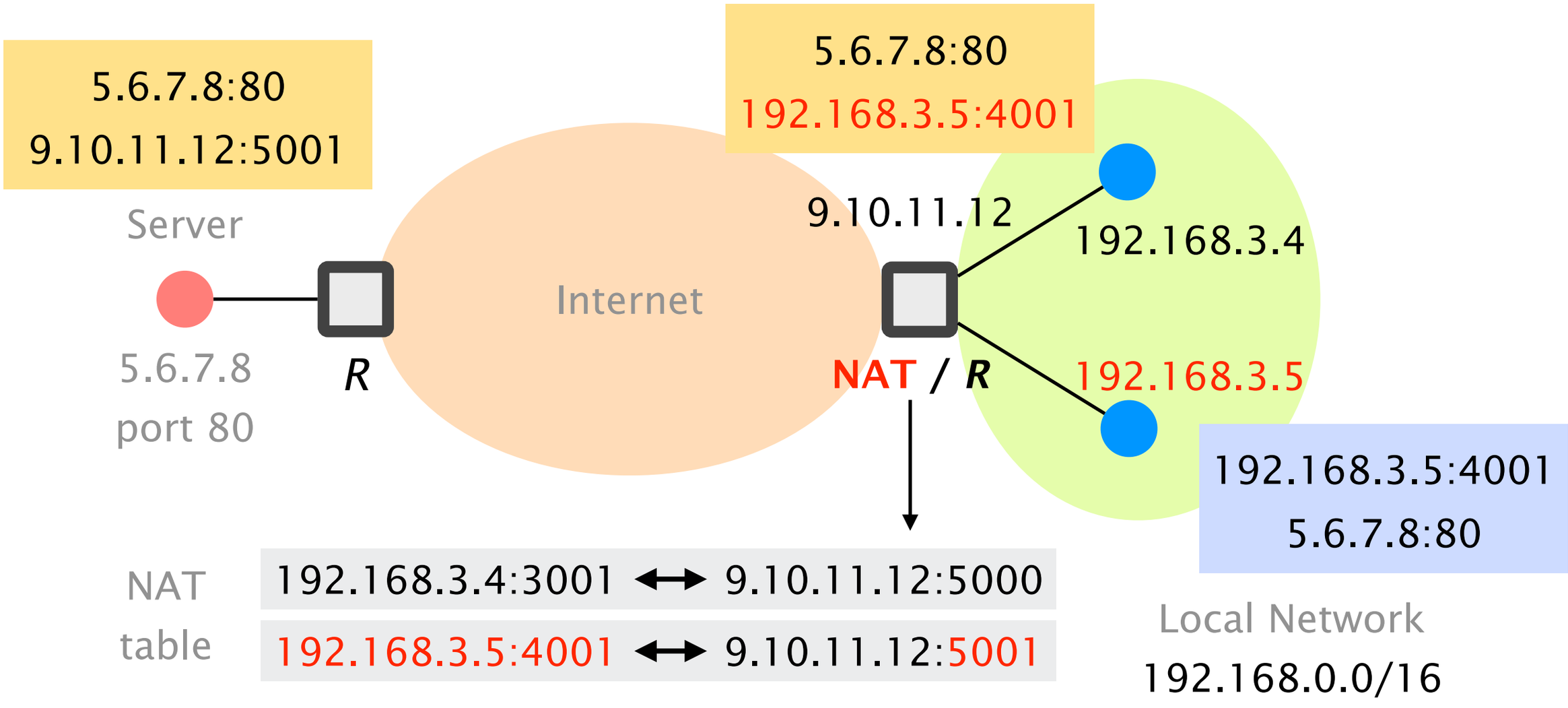
The Internet with NAT

Hosts behind NAT get a private address



The Internet with NAT

The port numbers are used to multiplex single addresses



NAT also provides other (dis-)advantages

Better privacy

All hosts in one network get the same public IP

But, cookies, browser version, … still identify hosts

Better security

From the outside you cannot directly reach the hosts

Problematic e.g., for online gaming

Reduced scalability (size of the mapping table)

Example: Wi-Fi access problems in public places

(e.g., lecture hall) often due to a full NAT table

Enters IPv6

The easy way to think of IPv6 is to consider it as equivalent to IPv4 but with 128 bits addresses

Notation	8 groups of 16 bits each separated by colons (:)
	Each group is written as four hexadecimal digits
Simplification	Leading zeros in any group are removed
	Max 1 section of zeros can be replaced by a double colon (::)
	Normally, the longest section
Examples	1080:0:0:0:8:800:200C:417A → 1080::8:800:200C:417A
	FF01:0:0:0:0:0:0:0101 → FF01::101
	0:0:0:0:0:0:0:1 → ::1

There are three types of IPv6 addresses: unicast, anycast, and multicast

Unicast	Identifies a single interface Packets are delivered to this specific interface
Anycast	Identifies a set of interfaces Packets are delivered to the *nearest* interface
Multicast	Identifies a set of interfaces Packets are delivered to *all* interfaces

Unicast

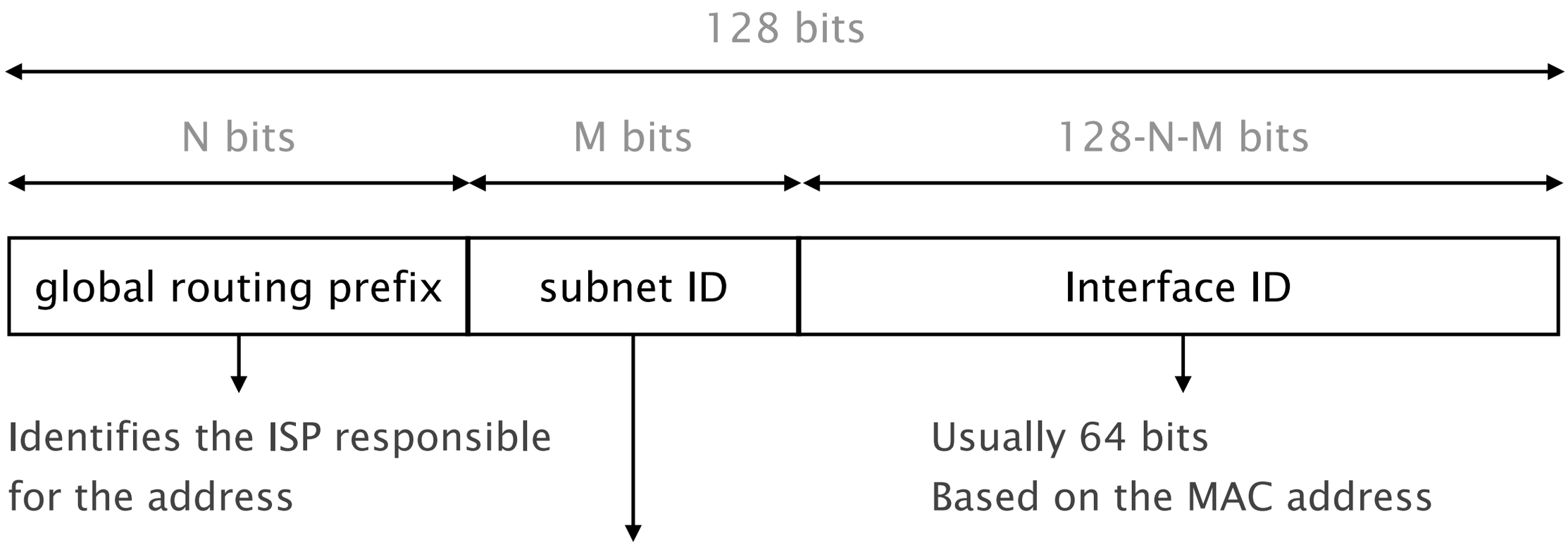
Identifies a single interface

Packets are delivered to this specific interface

Global unicast addresses

are hierarchically allocated

similar to global IPv4 addresses



Allocation of IPv6 (global unicast) addresses

Internet Assigned Numbers Authority

The Internet Assigned Numbers Authority (IANA)
assigns blocks to Regional IP address Registries (RIR)

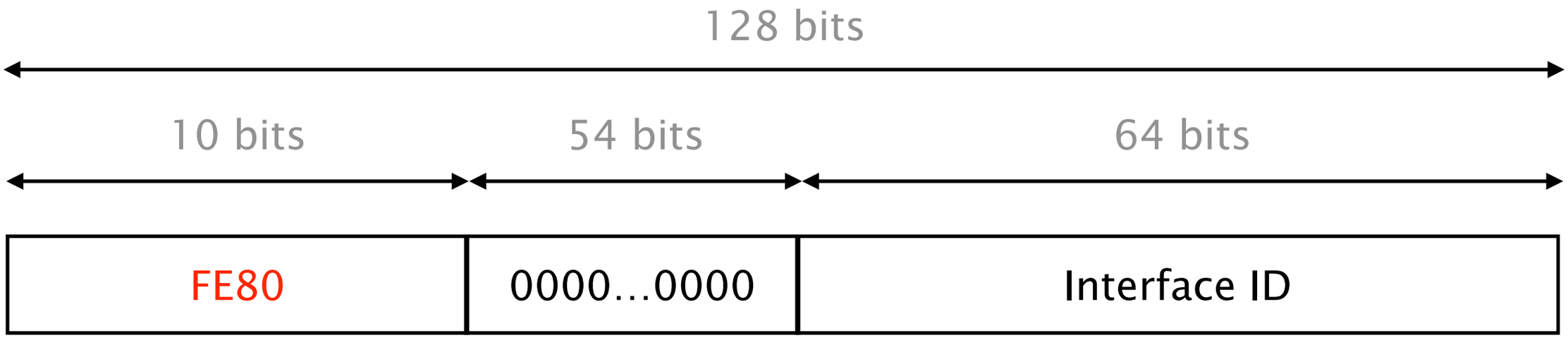
For example RIPE, ARIN, APNIC, …

Currently, only 2000::/3 is used for global unicast

All addresses are in the range of 2000 to 3FFF

Link-local addresses are unique to a single link (subnet)

same as private IPv4 addresses



Each host/router **must** generate a link-local address for **each** of its interfaces

An interface therefore can have multiple IPv6 addresses

ETH's IPv6 prefix **2001:67c:10ec::/48**

```
inet6num:       2001:67c:10ec::/48
netname:        ETHZ-NET-IPv6
descr:          ETHZ
descr:          Zurich, Switzerland
country:        CH
org:            ORG-ETHZ1-RIPE
admin-c:        AW1297-RIPE
tech-c:         HE688-RIPE
status:         ASSIGNED PI
mnt-by:         RIPE-NCC-END-MNT
mnt-by:         SWITCH-MNT
mnt-routes:     SWITCH-MNT
mnt-domains:    SWITCH-MNT
created:        2012-09-18T11:49:33Z
last-modified:  2016-04-14T08:45:10Z
source:         RIPE # Filtered
sponsoring-org: ORG-SG2-RIPE
```

In addition to global and link-local addresses, some IPv6 unicast addresses have a special meaning

Unspecified address	0:0:0:0:0:0:0:0 Used as src address if no IPv6 address available
Loopback address	0:0:0:0:0:0:0:1 → ::1 127.0.0.1 for IPv4 addresses
IPv4 embedded	The lowest 32 bits contains an IPv4 address useful when deploying IPv6
Important	There are no IPv6 broadcast addresses

Anycast

Identifies a set of interfaces

Packets are delivered to the „nearest“ interface

IPv6 anycast addresses

Multiple interfaces with the same address

Packets are sent to the nearest interface

Anycast use the global unicast address range

E.g. for DNS or HTTP services

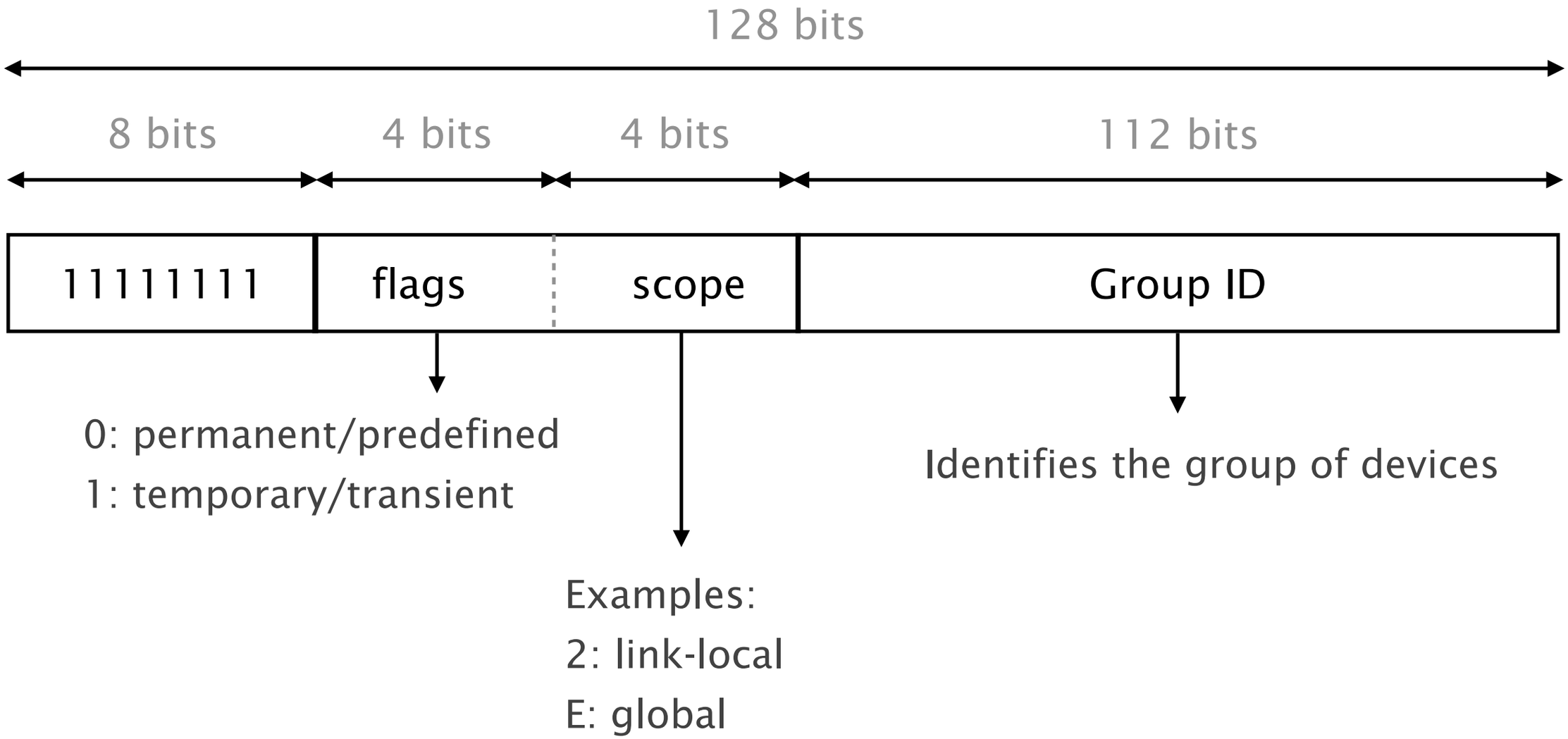
IPv6 anycast is rarely used (as of now)

Multicast

Identifies a set of interfaces

Packets are delivered to **all** interfaces

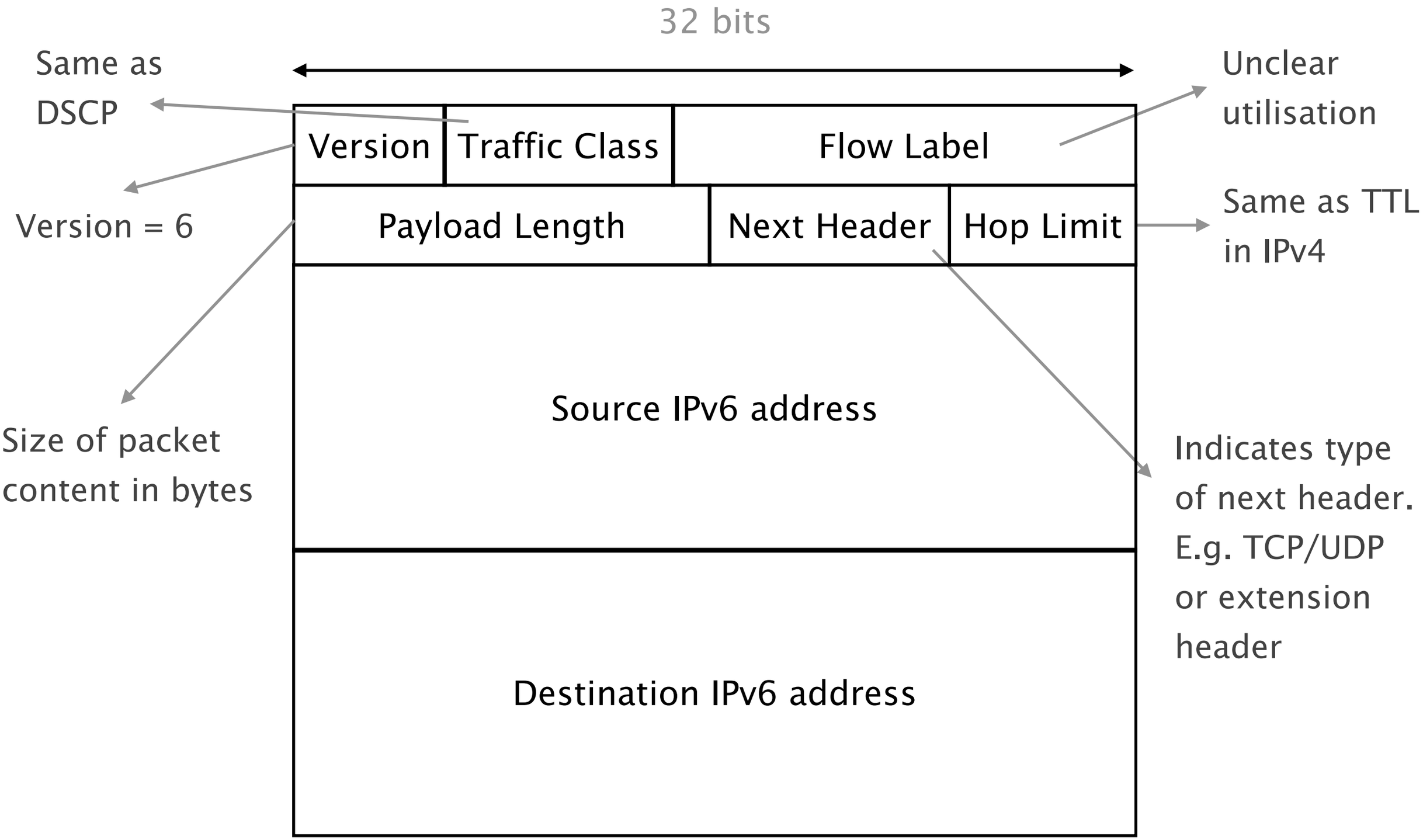
Multicast addresses identify
a group of receivers/interfaces



Some multicast addresses are well-known and used for auto-discovery, bootstrapping, etc.

FF02::1	All IPv6 end-systems E.g. hosts, servers, routers, mobile devices, …
FF02::2	All IPv6 routers All routers automatically belong to this group

The IPv6 packet header format



Compared to IPv4,
IPv6 does…

not include checksums in the packet header

link, transport or application layer provide checksums

not support fragmentation

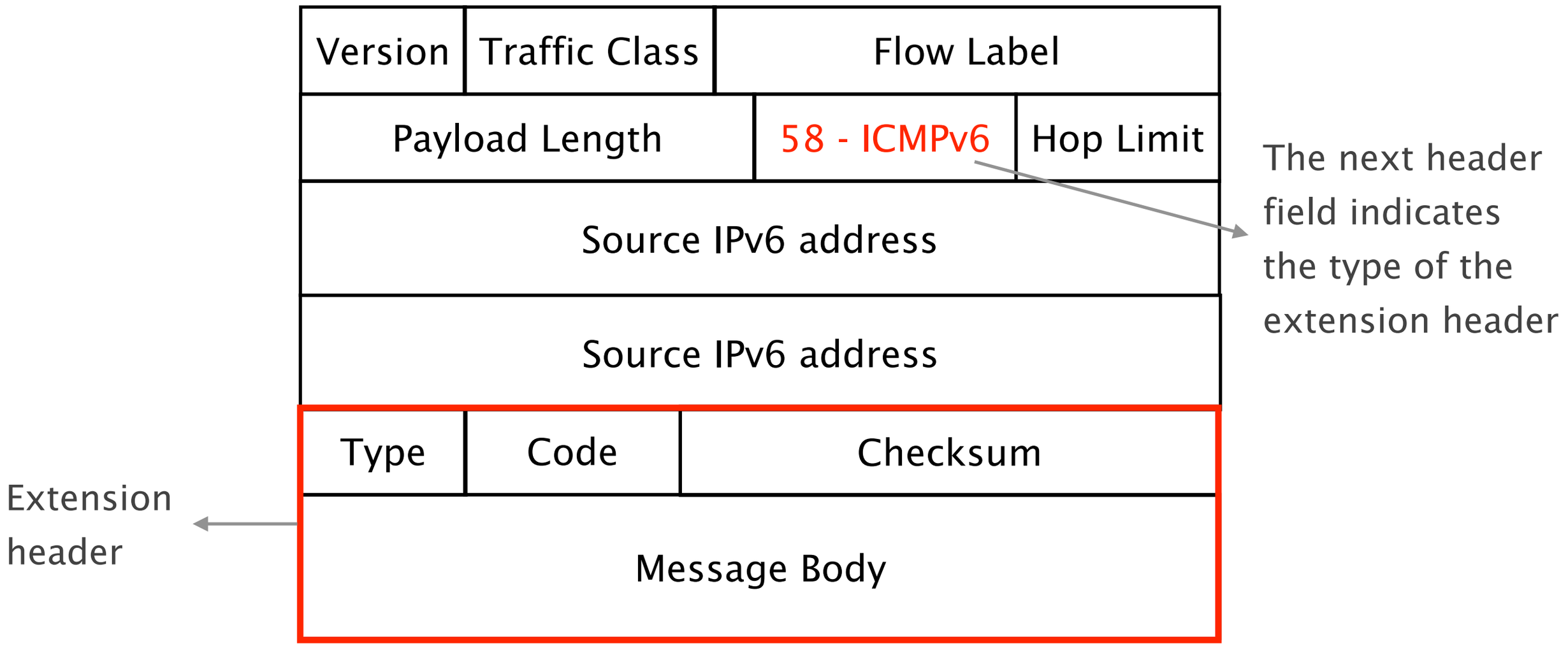
End host is required to send small enough packets

provide more flexibility

flow labels and extension headers

Extension header example: ICMPv6

Similar functions than IPv4 ICMP

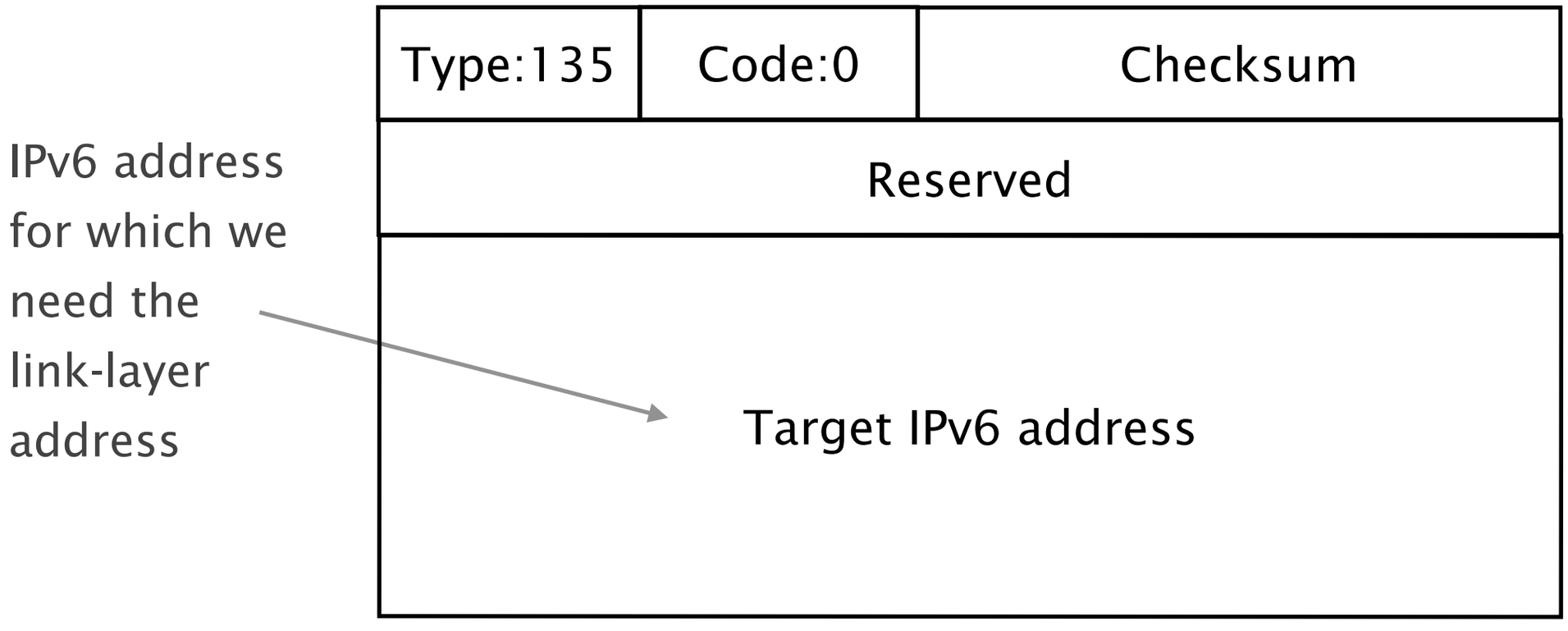


ICMPv6 can be used for

neighbor discovery

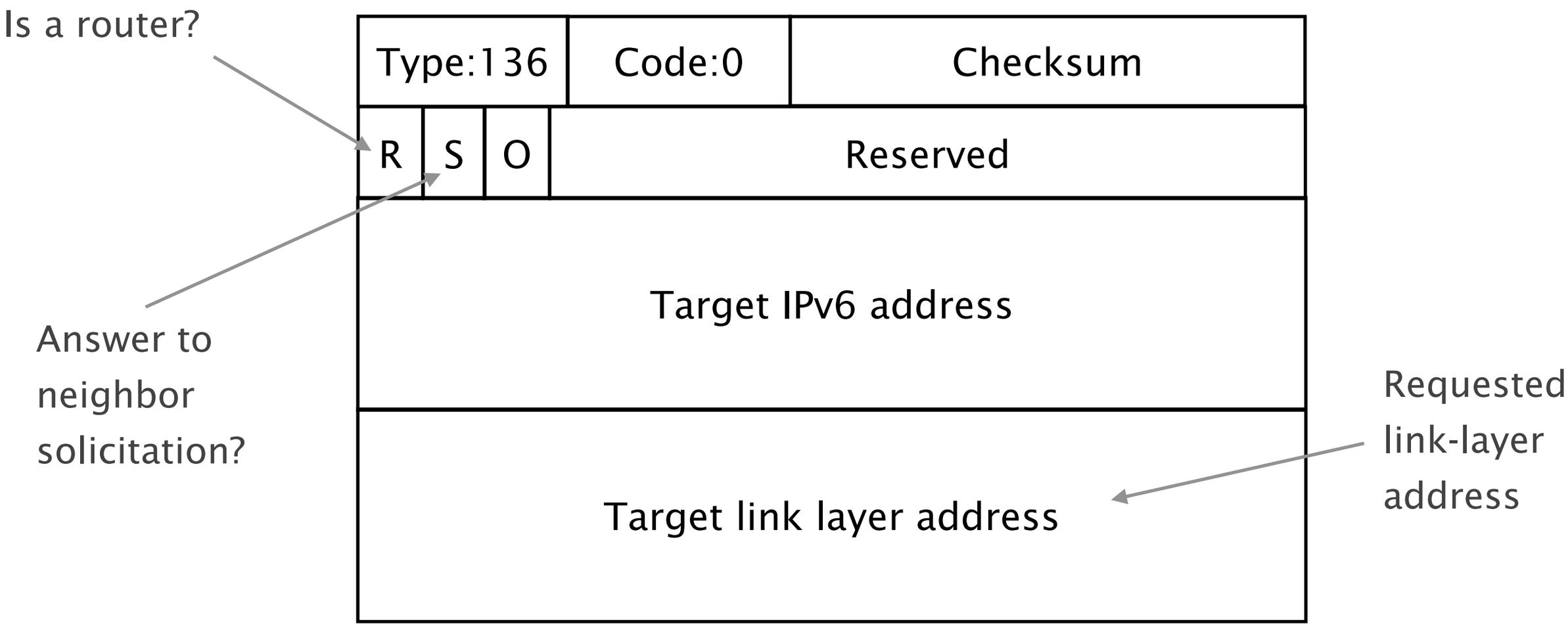
replacement for IPv4's ARP

First step: neighbor solicitation



ICMPv6 can be used for neighbor discovery

Second step: neighbor advertisement



How can a node obtain its IPv6 address(es)?

Manual configuration

As in the project, e.g. with ifconfig

From a server by using DHCPv6

Similar to the IPv4 version

Automatically

Using its link-local address and neighbor discovery

IPv6 autoconfiguration
to find link-local address

Consider an end-system which has just started,

it needs an IPv6 address to send ICMPv6 messages

Ethernet (MAC): 0800:200C:417A

Link-local: FE80::M_{64}(800:200C:417A)

M_{64}: 64-bit representation of the MAC address

Neighbor solicitation for FE80::M_{64}(800:200C:417A)

If **no** answer, the created link-local address is valid

IPv6 autoconfiguration
to obtain the IPv6 prefix of subnet

Routers periodically advertise the prefix

Sent to all end-systems: FF02::1

The advertisements can contain:

IPv6 prefix and length

Network MTU to use

Maximum hop limit to use

Lifetime of the default router

How long generated addresses are preferred

IPv6 autoconfiguration
to build global unicast address

Ethernet (MAC): 0800:200C:417A

Prefix: 2001:6a8:3080:1::/64

Global unicast:

2001:6a8:3080:1:M_{64}(800:200C:417A)

contains MAC address of host

To port your IPv4-based application to IPv6, you need to…

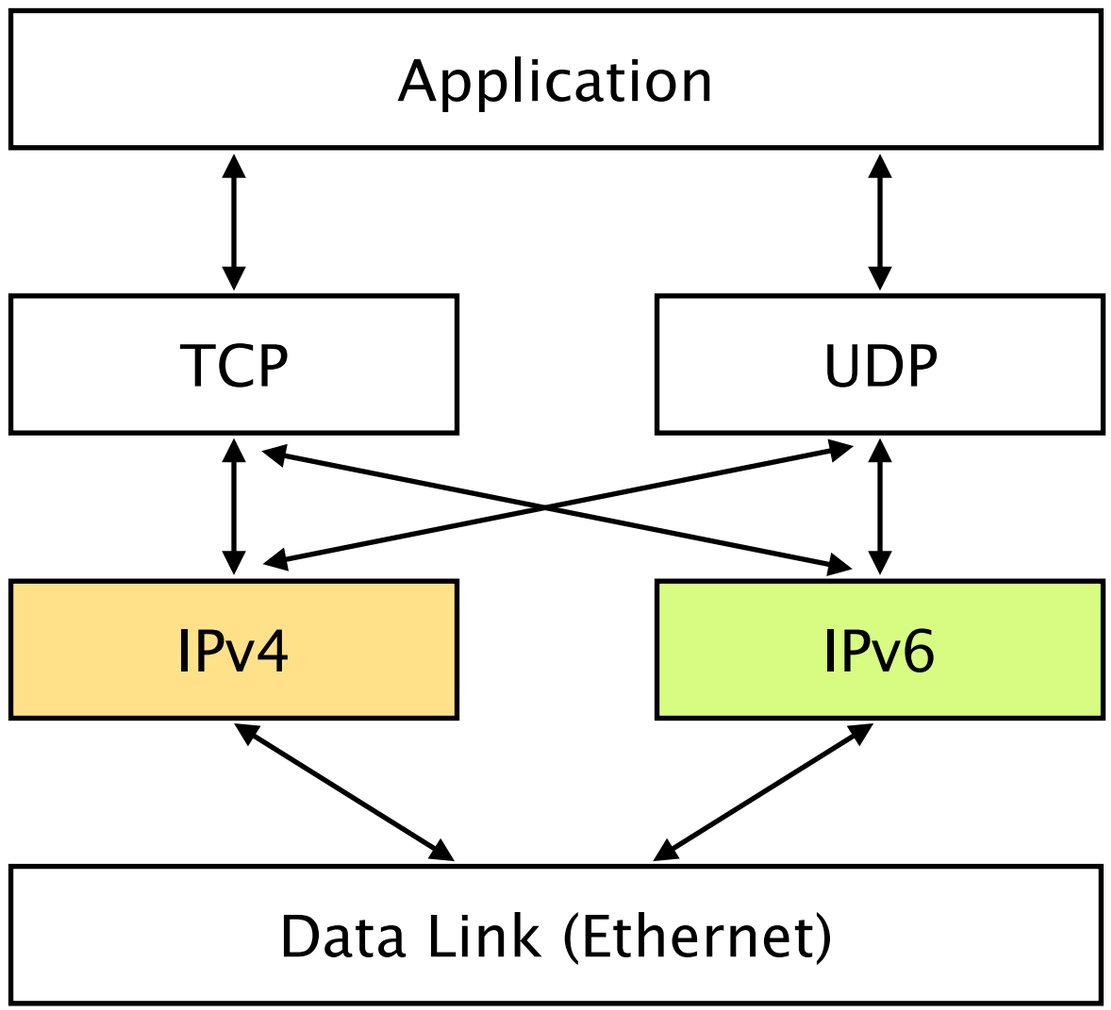
change the used socket functions

adjust all logging functions

adapt all data structures to support IPv6 addresses

adjust user interface elements to display IPv6

Today, a lot of applications and OSes use a dual stack approach



Over the years, a lot of

transition mechanisms were developed

6in4

6to4

Teredo

SIIT

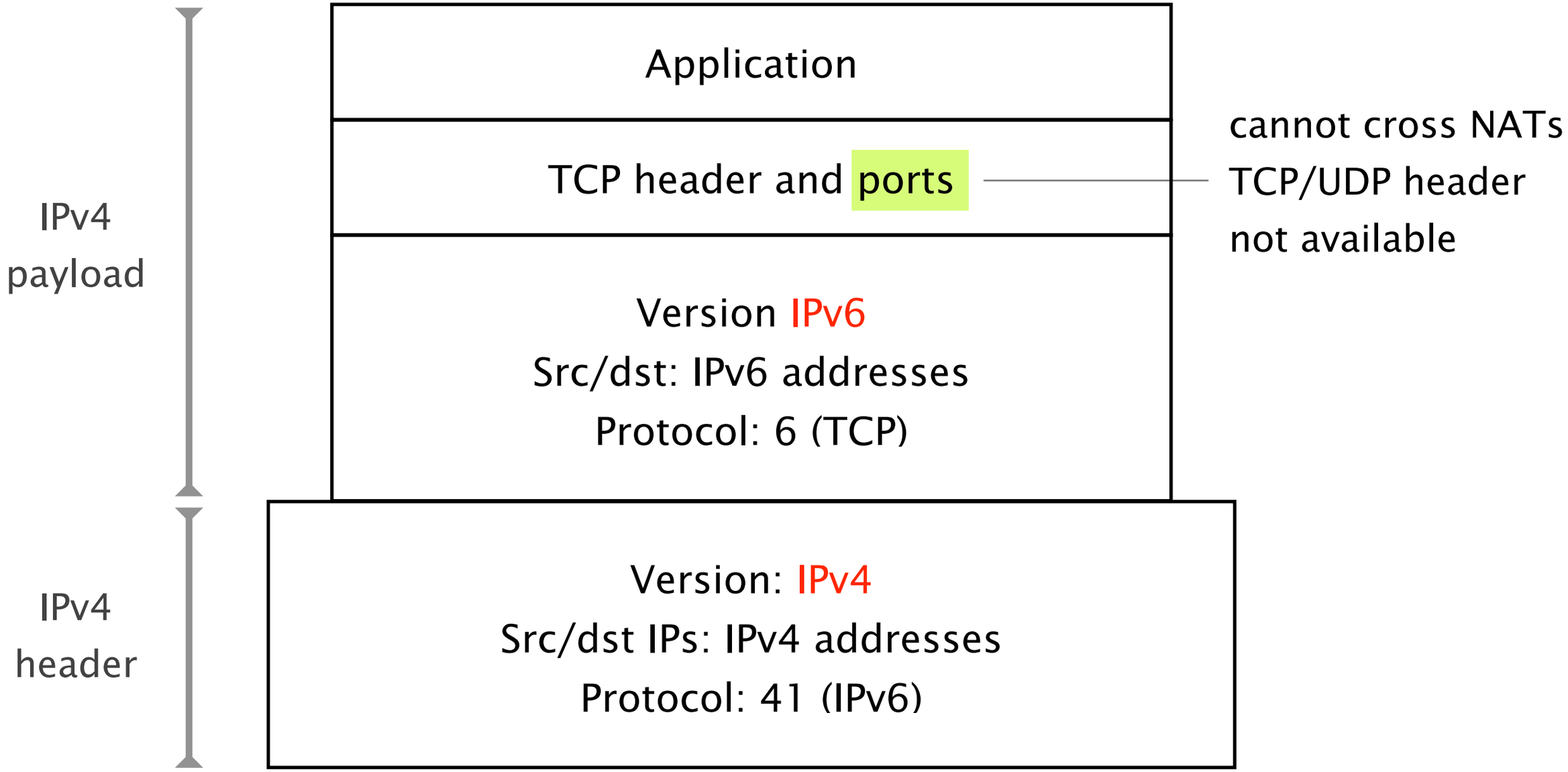
6rd

GRE

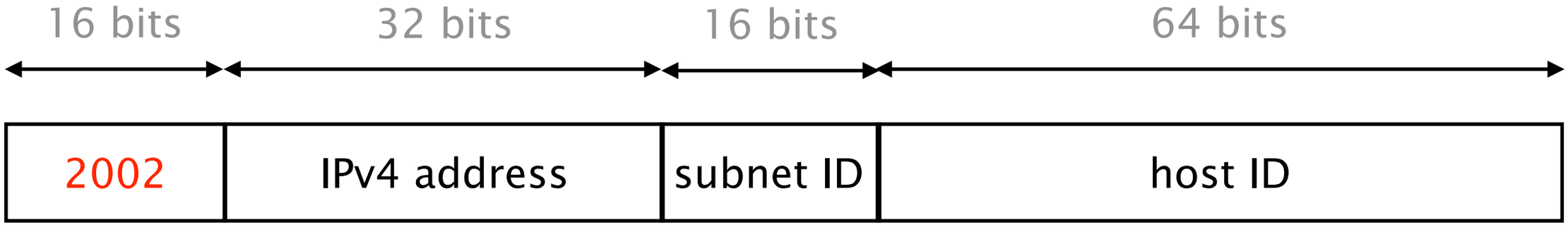
AYiYA

…

Tunnel IPv6 packets over static IPv4 links (6in4)



6to4 uses special IPv6 addresses

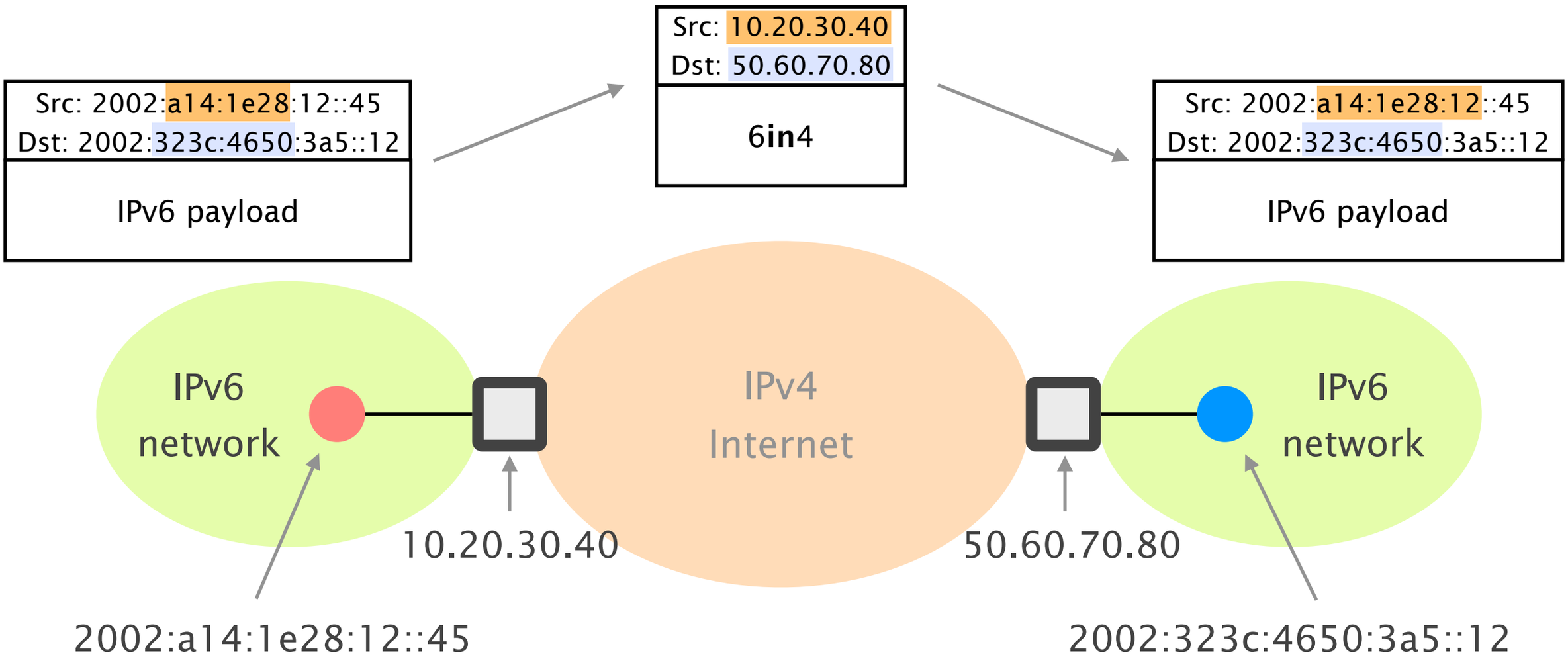


IPv4: 192.15.3.73

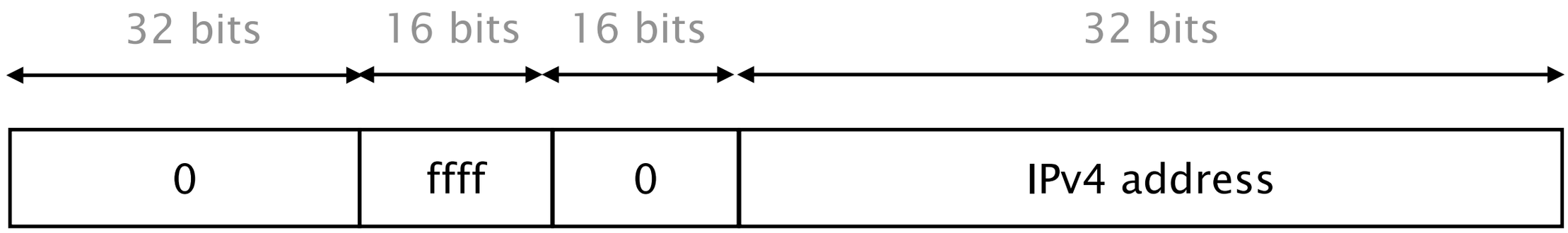
c0.0f.03.49

6to4: 2002:c00f:0349::/48

6to4 transmits IPv6 packets over IPv4 networks without explicit tunnels



Stateless IP/ICMP Translation (SIIT)
uses IPv4-embedded IPv6 addresses



Example: ::ffff:0:c00f:0349

Other notation: ::ffff:0:192.15.3.73

Similar to 6to4, a router translates addresses

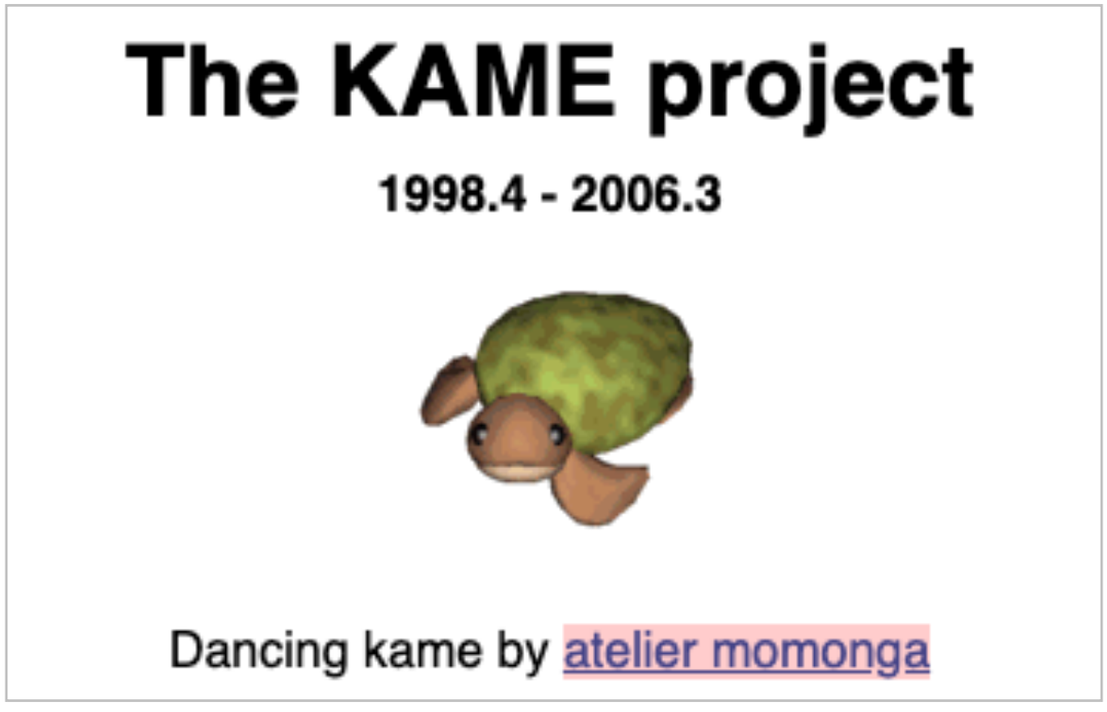
/96 prefix is such that checksums stay the same

when going from IPv6 to IPv4

If you don't have IPv6 @home already,
look at your set-top box configuration to activate it

(Swisscom set-top box's Configuration)

Head to www.kame.net to check
if you see the dancing turtle 😃



Communication Networks

Spring 2020

Laurent Vanbever

nsg.ee.ethz.ch

ETH Zürich (D-ITET)

May 18 2020