neat

QoS Challenges for Real Time Traffic

[tj]
tj@enoti.me

NEAT is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 644334.

Horizon 2020
European Union funding
for Research & Innovation

Internet Protocol Datagram

RFC791

Source

Destination

Version ☐ *If other than version 4, attach form RFC 2460.*

Type of Service
- ☐ high reliability
- ☐ high throughput
- ☐ low delay

Precedence
- ☐ Routine
- ☐ Priority
- ☐ Immediate
- ☐ Flash
- ☐ Flash Override
- ☐ CRITIC/ECP
- ☐ Internetwork Control
- ☐ Network Control

Fragmentation *Transport layer use only*
- ☐ more to follow
- ☐ do not fragment
- ☐ this bit intentionally left blank

Offset ☐☐☐☐

Identifier _____

Protocol
- ☐ TCP
- ☐ UDP
- ☐ Other _____

Length ☐☐☐☐☐

Header Length ☐☐

Data
Print legibly and press hard. You are making up to 255 copies.

Time to Live ☐☐☐

Options *Do not write in this space.*

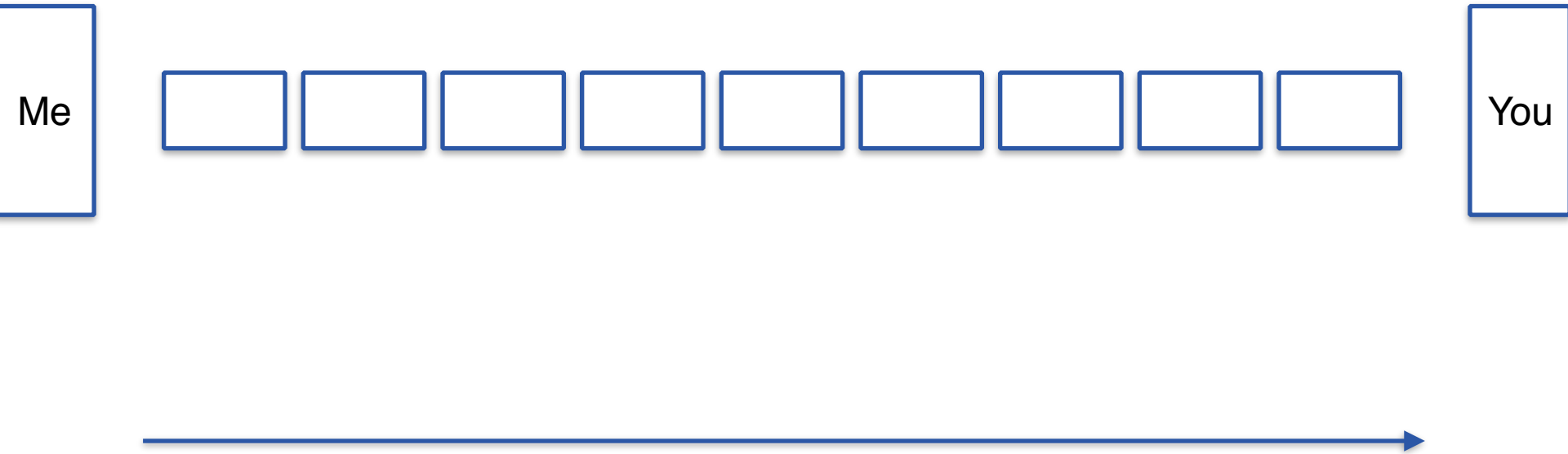
Header Checksum ☐☐☐☐

for more info, check IPv4 specifications at http://www.ietf.org/rfc/rfc0791.txt

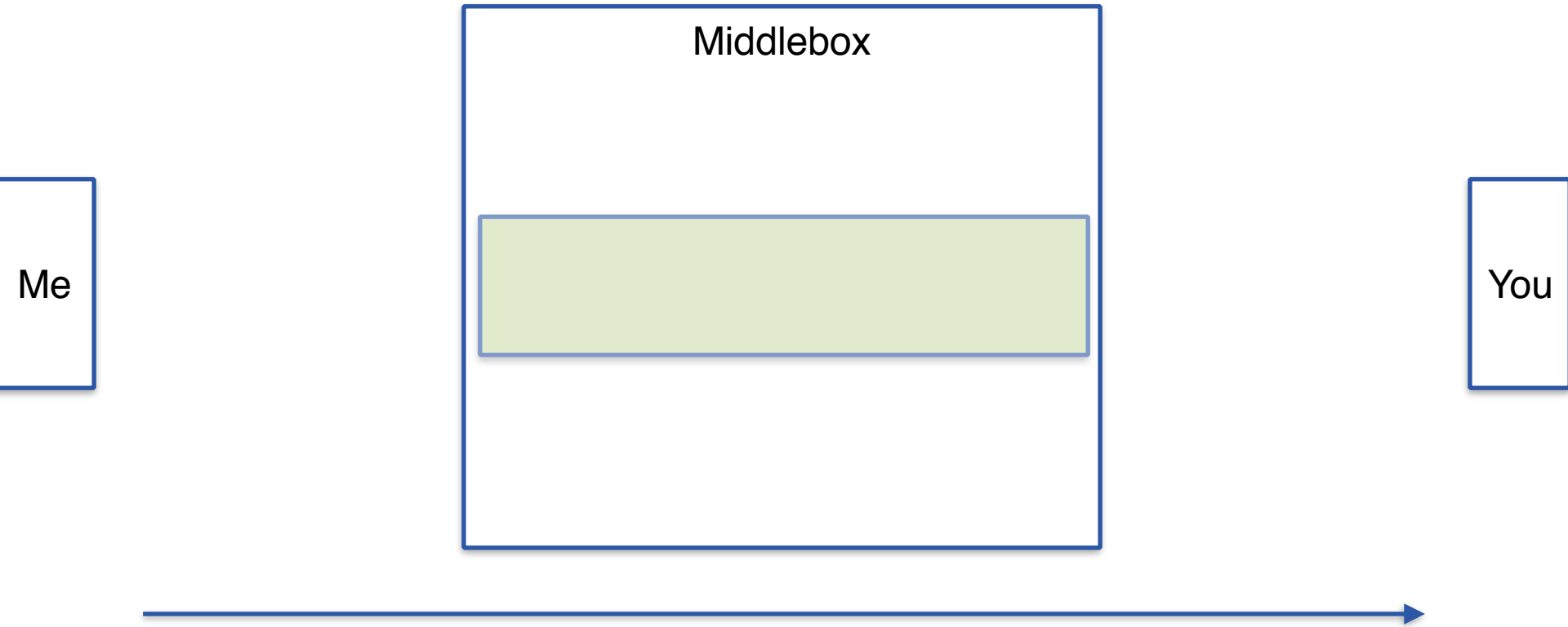
Packets

Packets

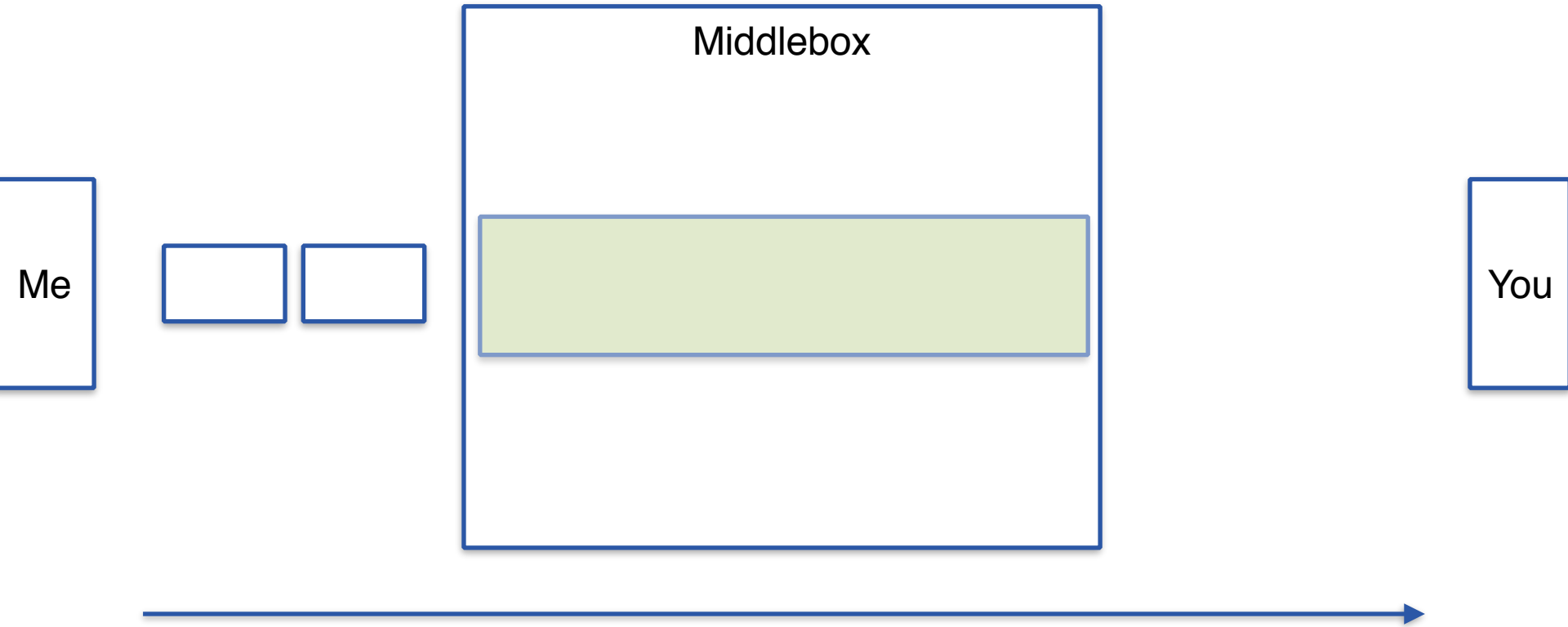
Packets



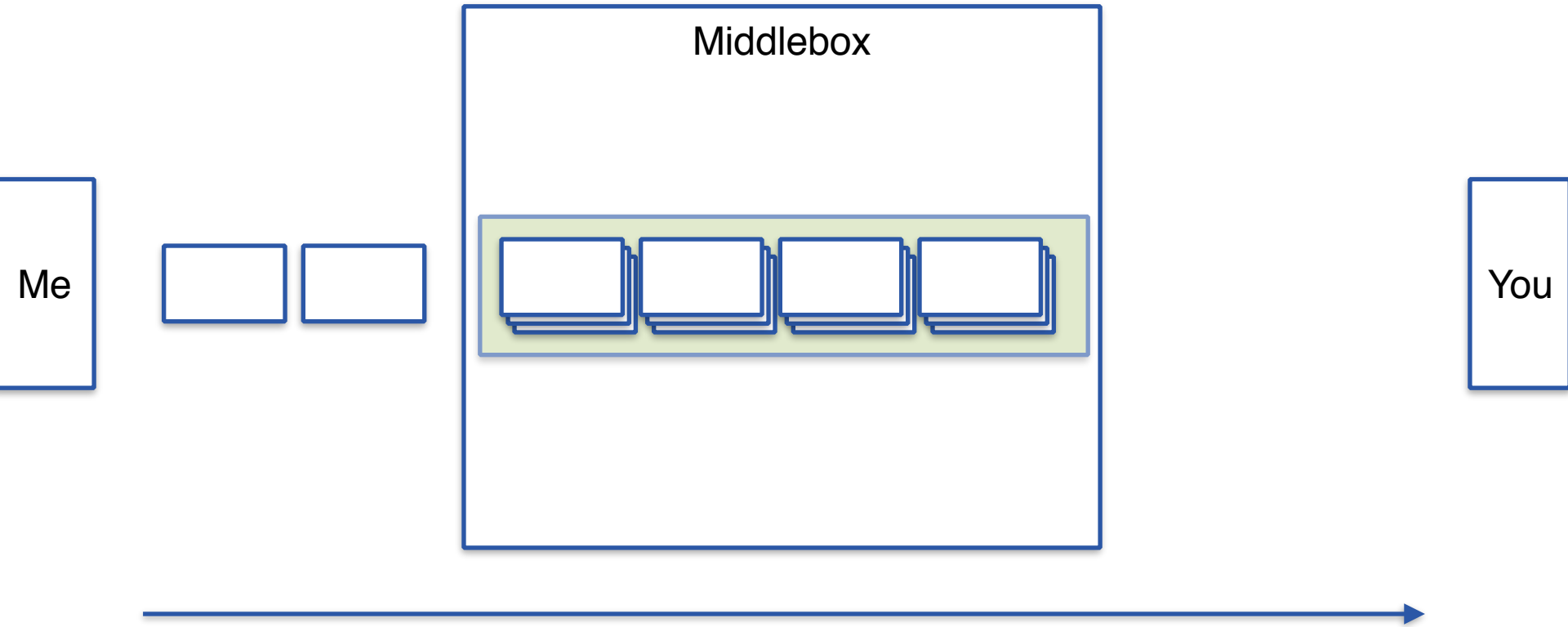
Middleboxes



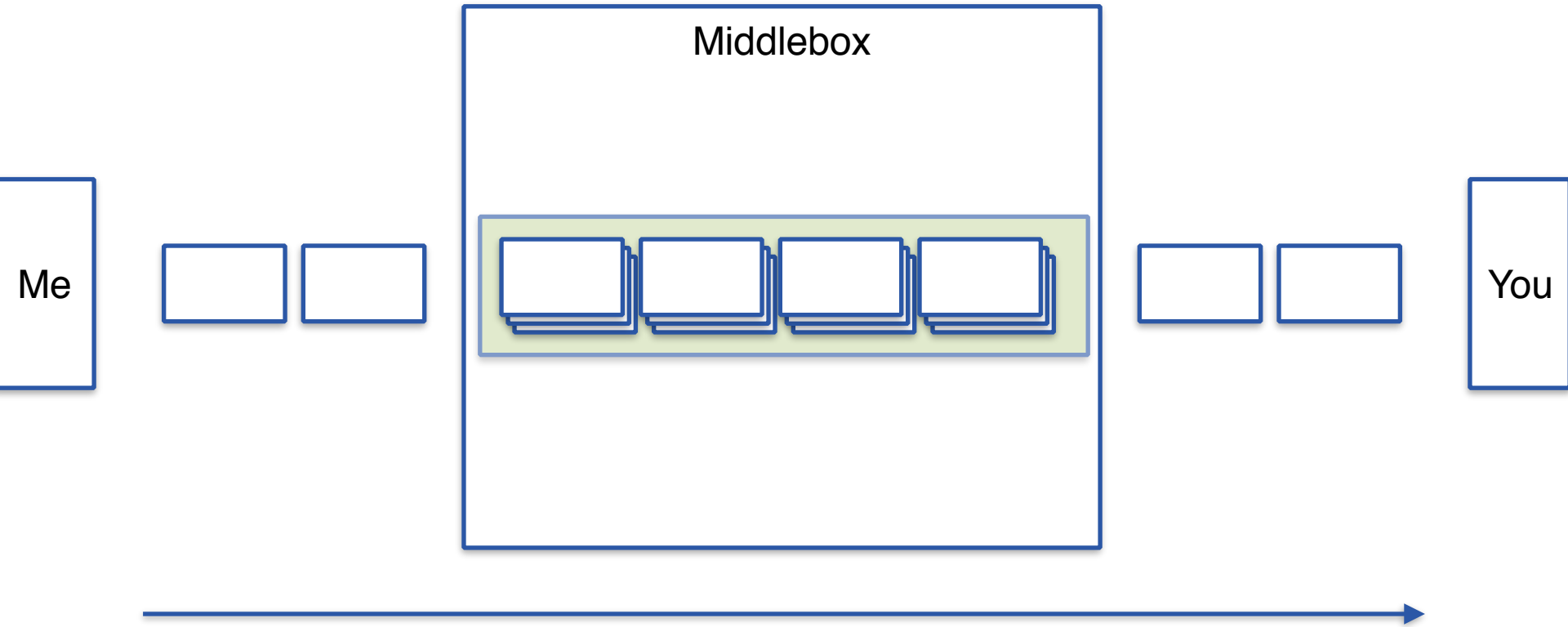
Middleboxes



Middleboxes



Middleboxes



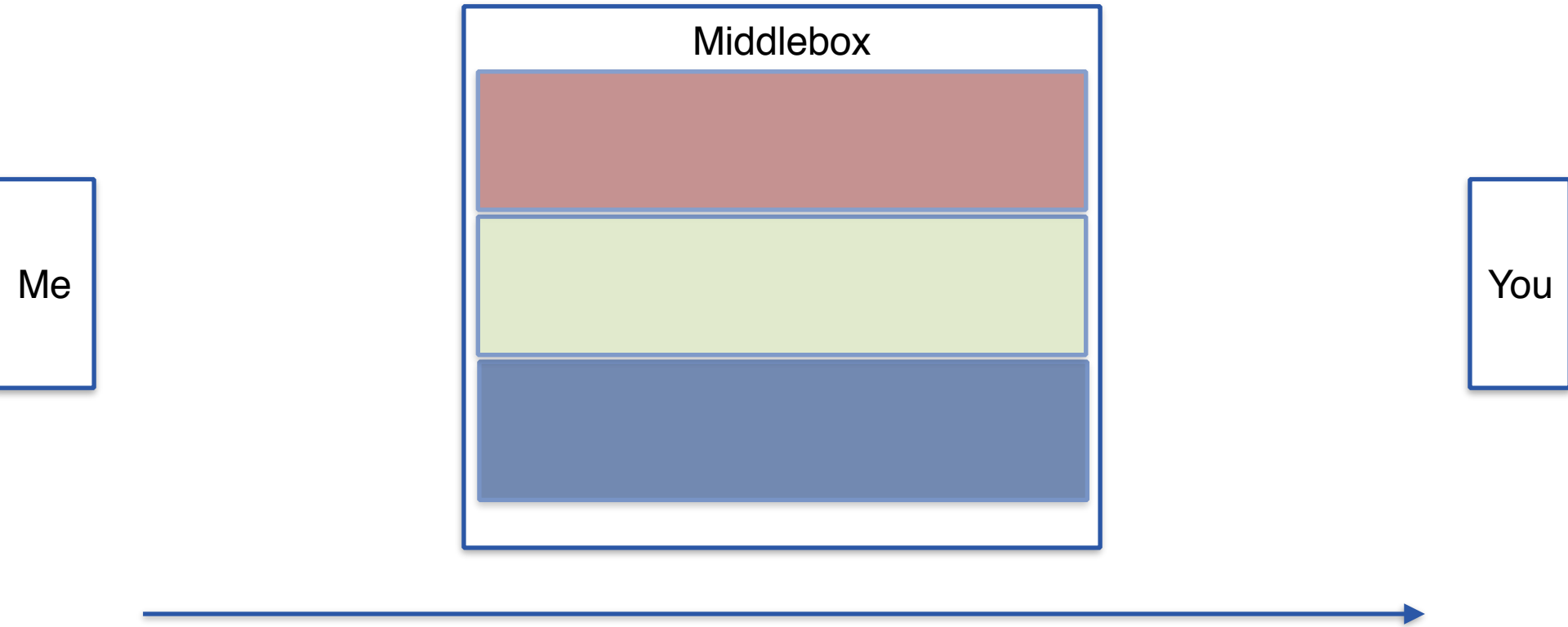
Per Hop Behaviours

- Default Forwarding

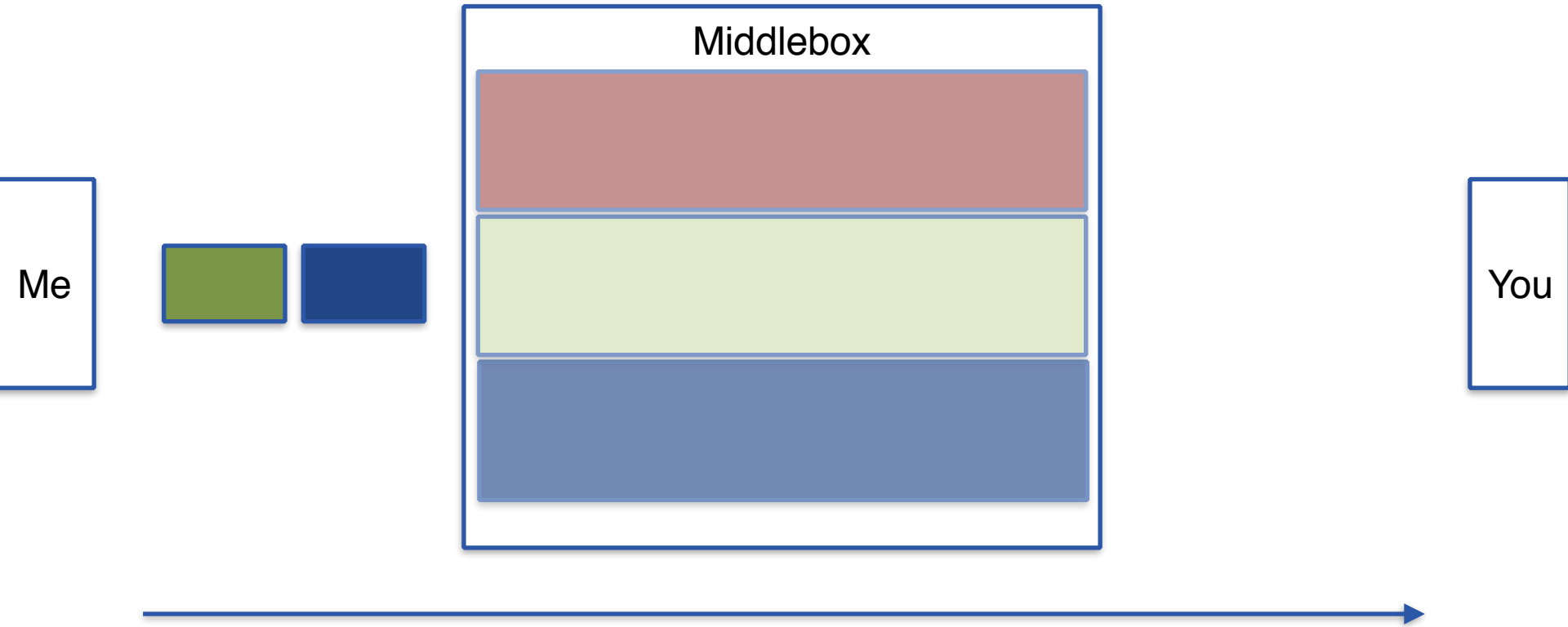
Differentiated Services

Differentiated Services

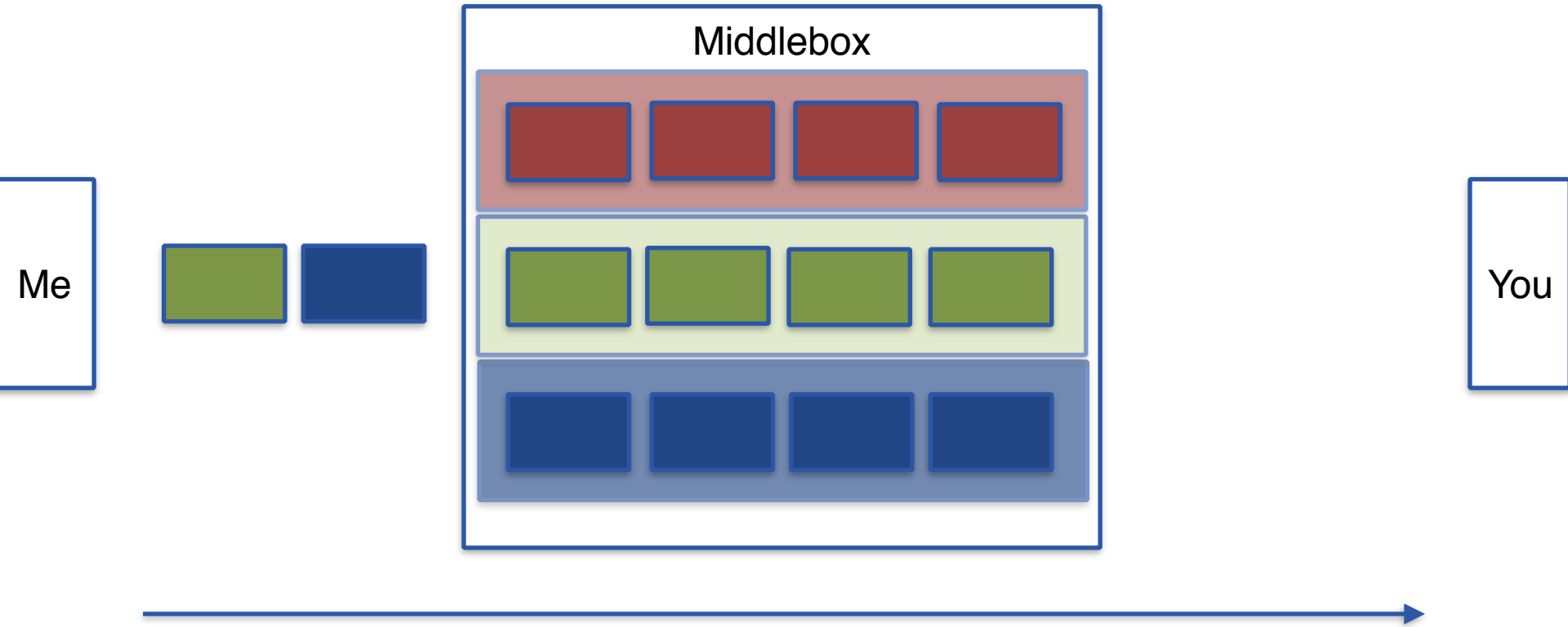
Smart Queueing



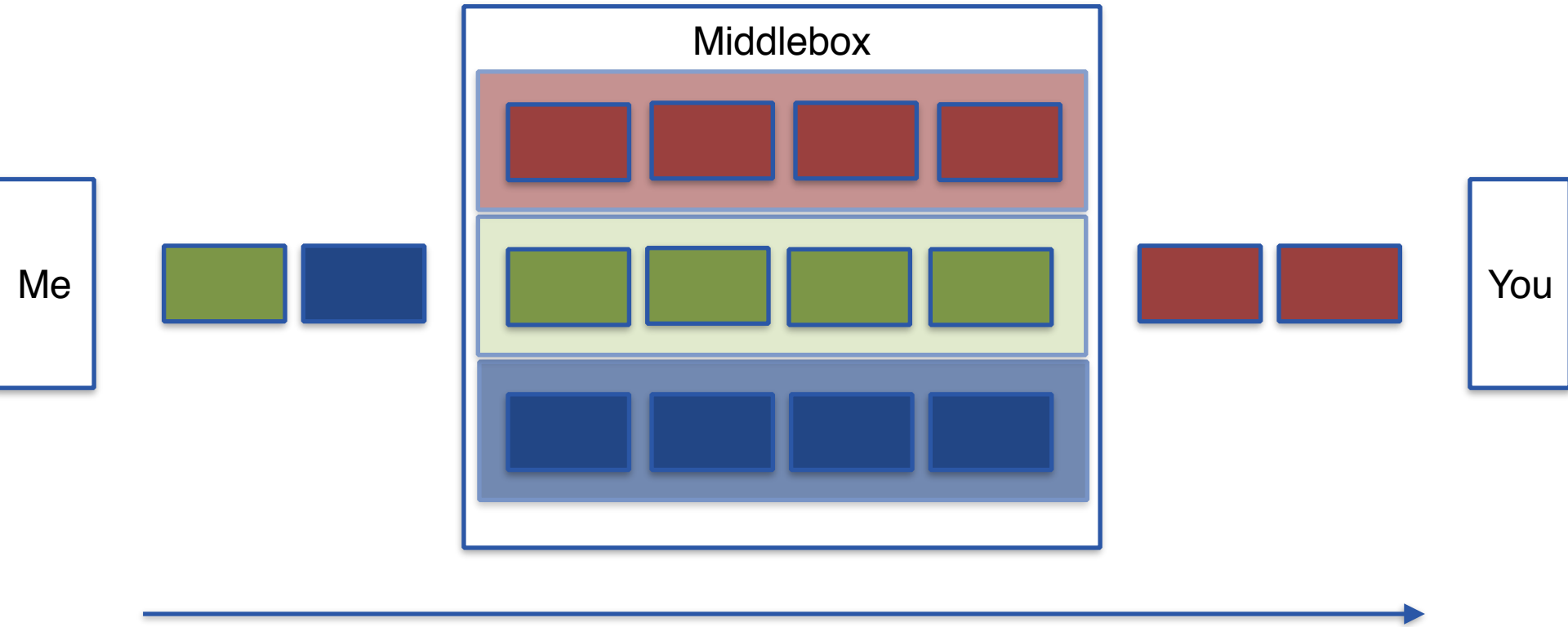
Smart Queueing



Smart Queueing



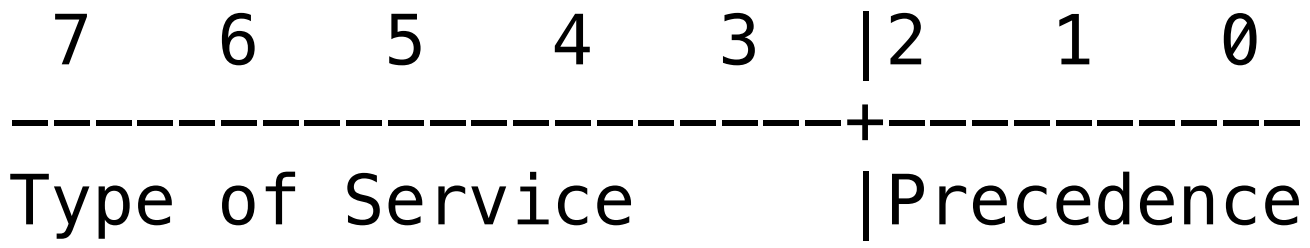
Smart Queueing



Per Hop Behaviours

- Default Forwarding
  - The default PHB has best-effort (BE) forwarding characteristics
- Expedited Forwarding
  - The EF PHB has the characteristics of low delay, low loss and low jitter.
- Assured Forwarding
  - Assured forwarding allows the operator to provide assurance of delivery as long as the traffic does not exceed some subscribed rate.

The IP Type of Service (TOS) Field



The IP Type of Service (TOS) Field

```
 7   6   5   4   3  |2   1   0
--------------------+----------
Type of Service     |Precedence
```

```
 7   6   5   4   3   2  |1   0
-----------------------+------
Diff Serv Code Point   |ECN
```

What Happens to Code Points?

- Generate a set of points to evaluate
  - RFC recommendations
- Measurement survey
  - Pass/Fail Test on Code Points
  - Verify Code Group Treatment

DSCP Treatment

- Pass
- Drop
- Bit Bleaching
- Remark

Code Points recommended by recent RFC's

- Which Code Points should we use?
- 80211
  - draft-ietf-tsvwg-ieee-802-11
- MPLS
  - RFC5127
  - draft-ietf-tsvwg-diffserv-intercon-14
- WebRTC
  - draft-ietf-tsvwg-rtcweb-qos

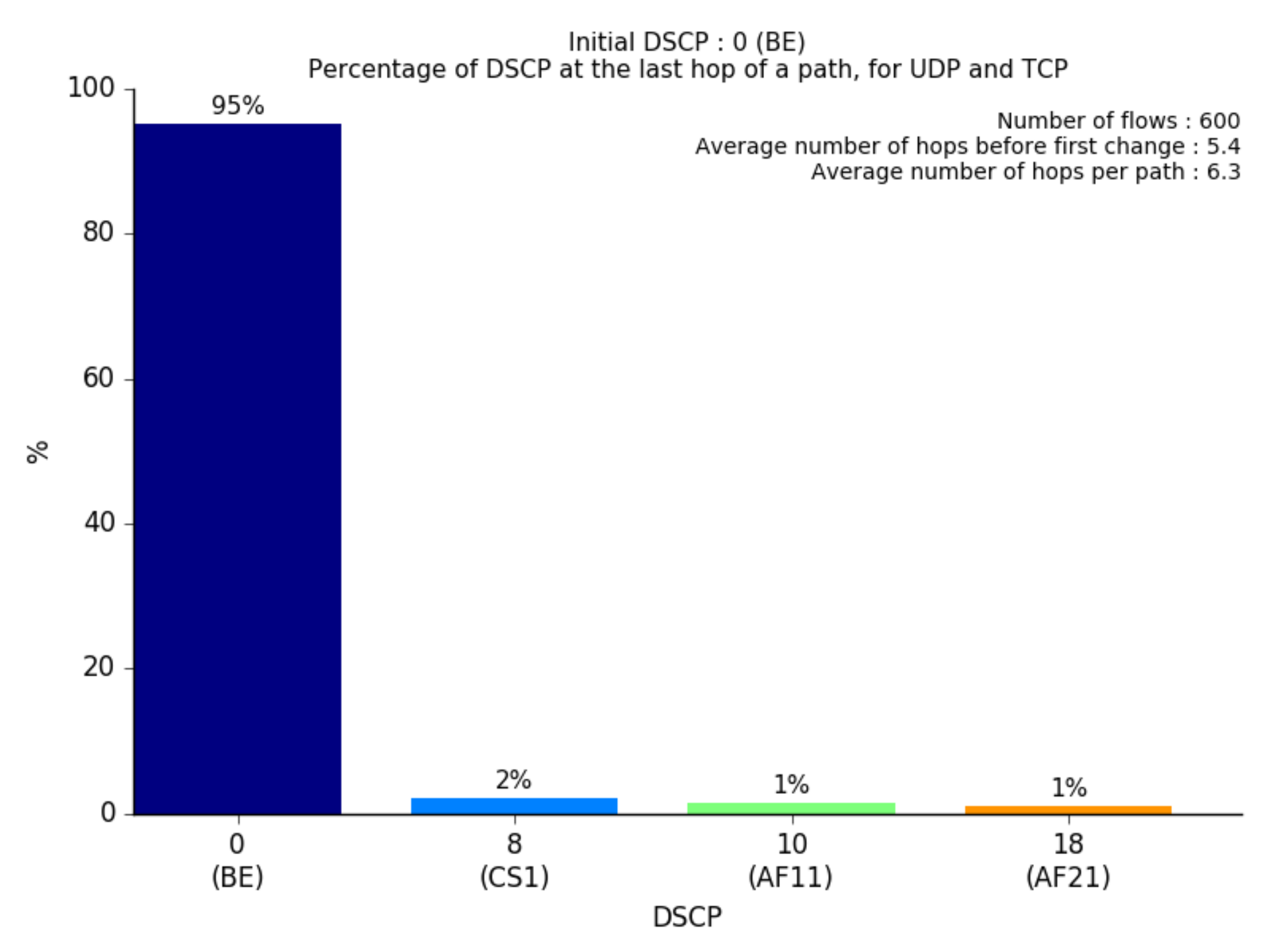
Code Points recommended by recent RFC's

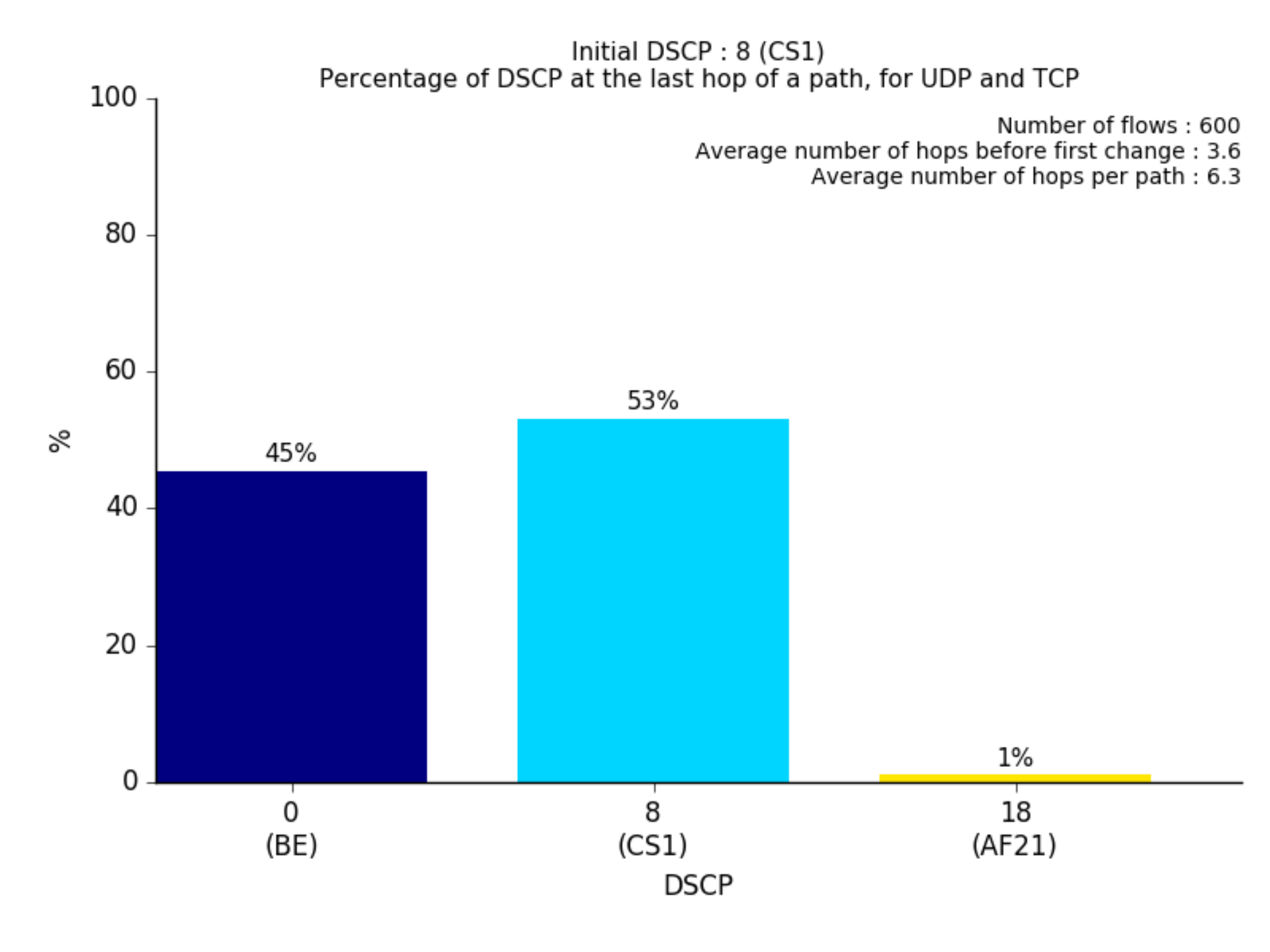
	CS0	AF11	AF31
	CS1	AF12	AF32
	CS2	AF13	AF33
DF	CS3		
EF	CS4	AF21	AF41
(LBE)	CS5	AF22	AF42
	CS6	AF23	AF43

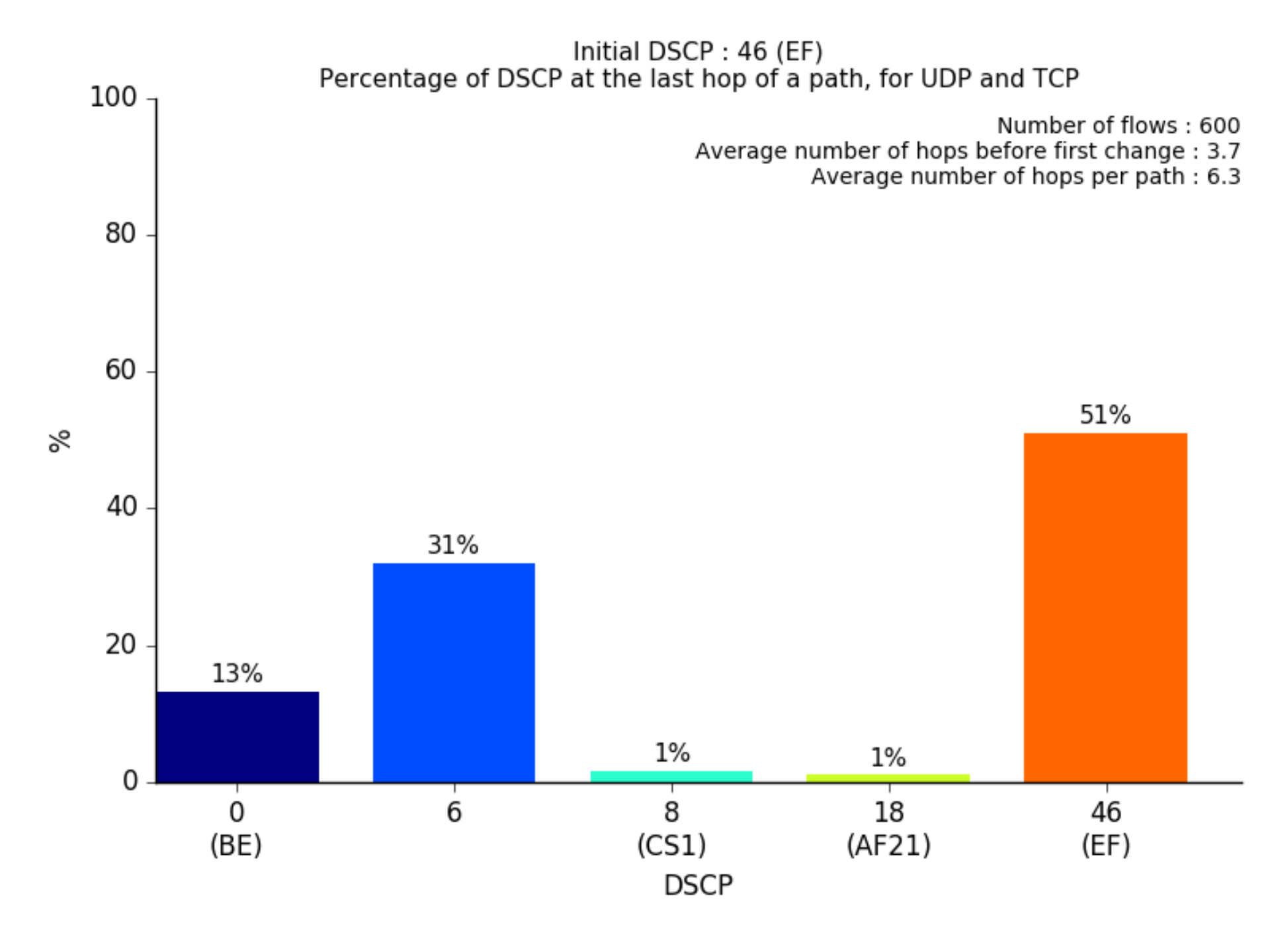
21 code points recommended

Graphs

Horizon 2020
European Union funding
for Research & Innovation


Initial DSCP : 0 (BE)
Percentage of DSCP at the last hop of a path, for UDP and TCP
Number of flows : 600
Average number of hops before first change : 5.4
Average number of hops per path : 6.3
100
80
60
40
20
0
%
95%
2%
1%
1%
0
(BE)
8
(CS1)
10
(AF11)
18
(AF21)
DSCP


Initial DSCP : 8 (CS1)
Percentage of DSCP at the last hop of a path, for UDP and TCP
Number of flows : 600
Average number of hops before first change : 3.6
Average number of hops per path : 6.3
100
80
60
40
20
0
%
45%
53%
1%
0
(BE)
8
(CS1)
18
(AF21)
DSCP


Initial DSCP : 46 (EF)
Percentage of DSCP at the last hop of a path, for UDP and TCP
Number of flows : 600
Average number of hops before first change : 3.7
Average number of hops per path : 6.3
100
80
60
40
20
0
%
13%
31%
1%
1%
51%
0
(BE)
6
8
(CS1)
18
(AF21)
46
(EF)
DSCP

Awesome! How do I use this?

Awesome! How do I use this?

```
uint8_t dscp = 0x2e             //EF
```


Awesome! How do I use this?

```
uint8_t dscp = 0x2e             //EF
uint8_t ecn = 0;
```

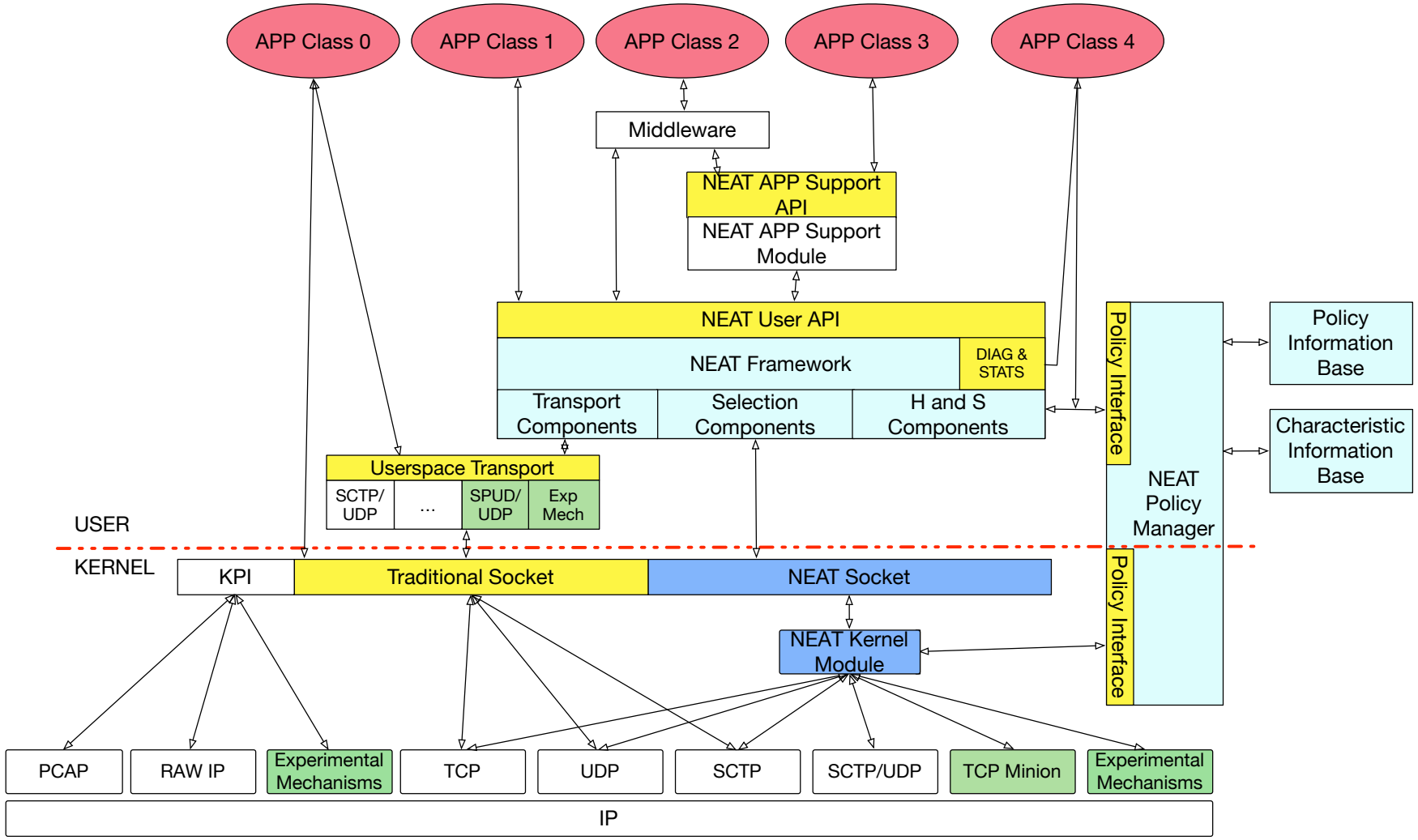
Awesome! How do I use this?

```
uint8_t dscp = 0x2e           //EF
uint8_t ecn = 0;
uint8_t tos = dscp << 2 | ecn;
```

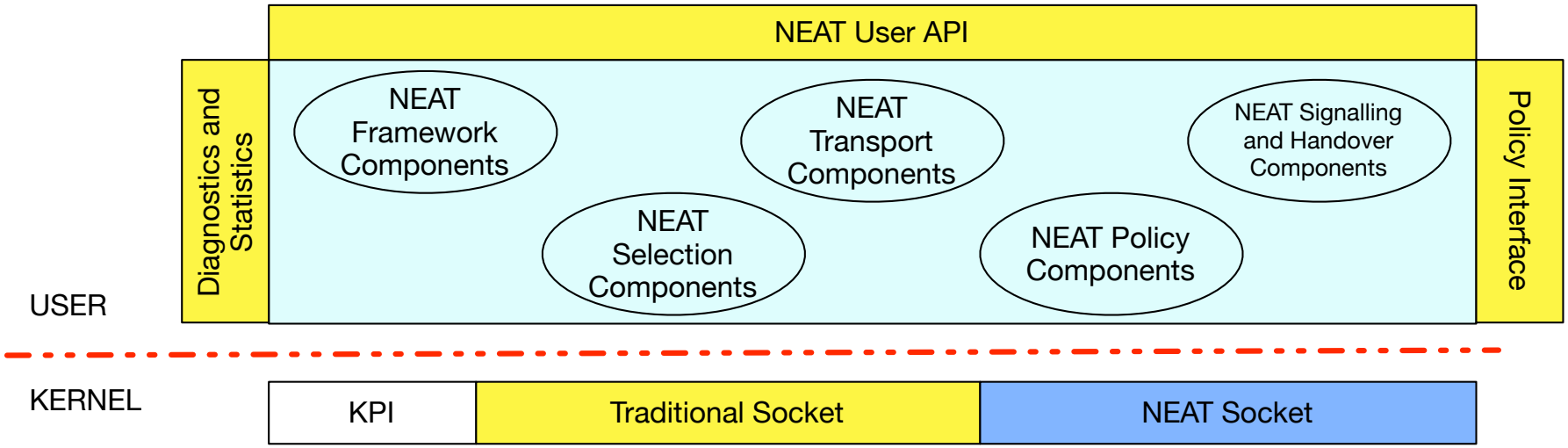
Awesome! How do I use this?

```
uint8_t dscp = 0x2e           //EF
uint8_t ecn = 0;
uint8_t tos = dscp << 2 | ecn;
if(setsockopt(flow->socket->fd,
    IPPROTO_IP, IP_TOS, &tos, sizeof(tos)) == -1) {
    return ERROR;
}
return OK;
```

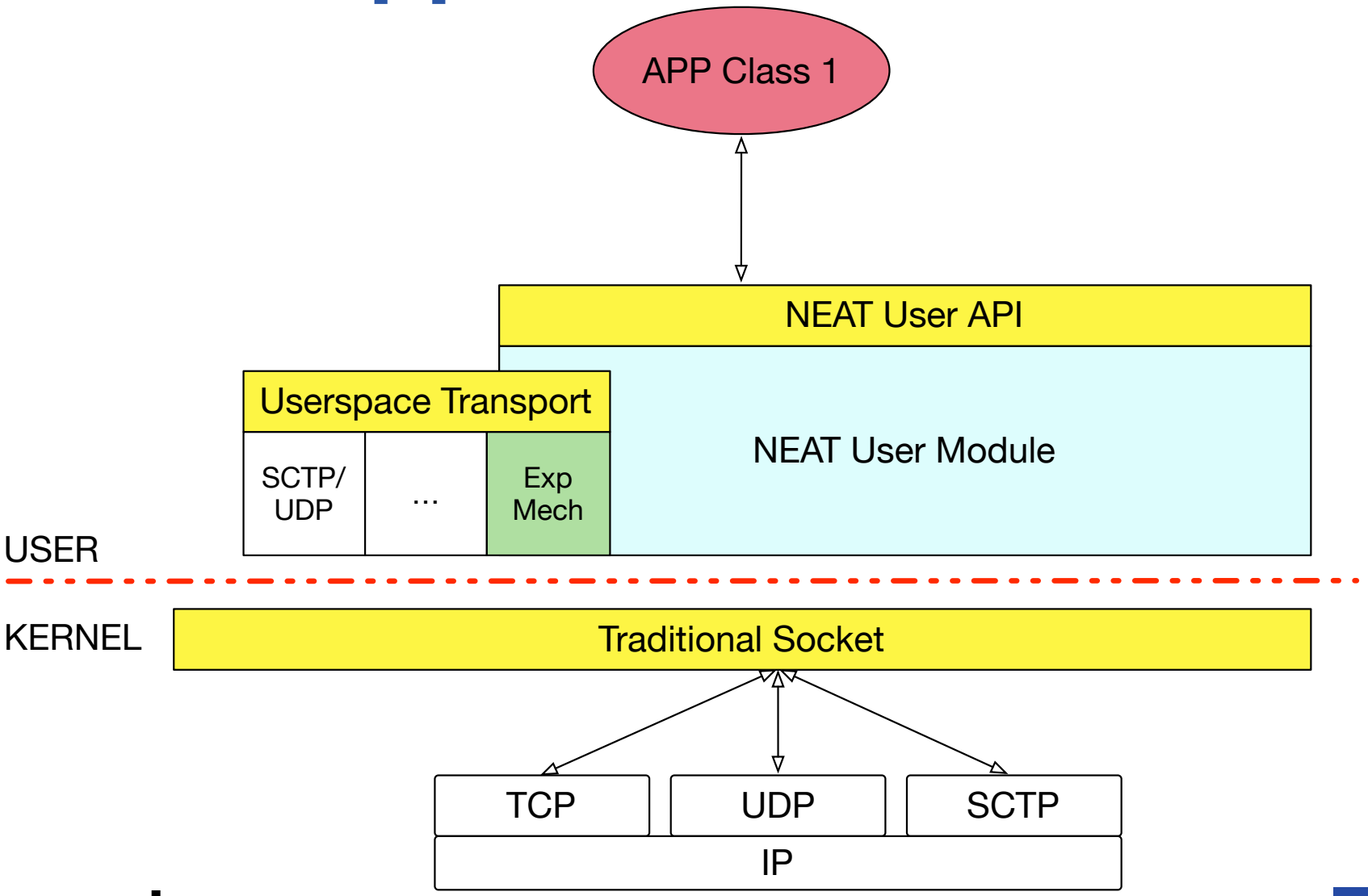

The NEAT System



The NEAT User Module



NEAT Application



NEAT Application

```
static struct neat_flow_operations ops;
static struct neat_ctx *ctx = NULL;
static struct neat_flow *flow = NULL;

ctx = neat_init_ctx()
flow = neat_new_flow(ctx)

prop = NEAT_PROPERTY_UDP_REQUIRED | NEAT_PROPERTY_IPV6_REQUIRED;
neat_set_property(ctx, flow, &prop)

ops.on_writable = on_writable;
ops.on_readable = on_readable;
ops.on_error = on_error;

neat_set_operations(ctx, flow, &ops)
neat_open(ctx, flow, argv[argc - 2], argv[argc - 1])

neat_start_event_loop(ctx, NEAT_RUN_DEFAULT);
```

NEAT Application

```
static neat_error_code
on_writable(struct neat_flow_operations *opCB)
{
    neat_write(opCB->ctx, opCB->flow, buf)
    return NEAT_OK;
}

static neat_error_code
on_readable(struct neat_flow_operations *opCB)
{
    neat_read(opCB->ctx, opCB->flow, buf)
    return NEAT_OK;
}
```

https://github.com/NEAT-project/neat/blob/master/examples/client.c

NEAT QoS Setting

NEAT QoS Setting

```
neat_set_qos(flow->ctx, flow, 0x2e);
```

NEAT QoS Setting

```
neat_set_qos(flow->ctx, flow, 0x2e);
neat_set_qos(flow->ctx, flow, NEAT_DSCP_EF);
```


NEAT QoS Setting

```
neat_set_qos(flow->ctx, flow, 0x2e);
neat_set_qos(flow->ctx, flow, NEAT_DSCP_EF);
neat_set_qos(flow->ctx, flow,
    NEAT_QOS_REALTIME_INTERACTIVE_DATA);
```


neat

https://www.neat-project.org

https://github.com/neat-project/neat

NEAT is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 644334.

Horizon 2020
European Union funding
for Research & Innovation