SRv6Pipes: enabling in-network bytestream functions

Fabien Duchêne <fabien.duchene@uclouvain.be>
David Lebrun <david.lebrun@uclouvain.be>
Olivier Bonaventure <olivier.bonaventure@uclouvain.be>

IFIP Networking 2018 - Zurich

SRv6Pipes?

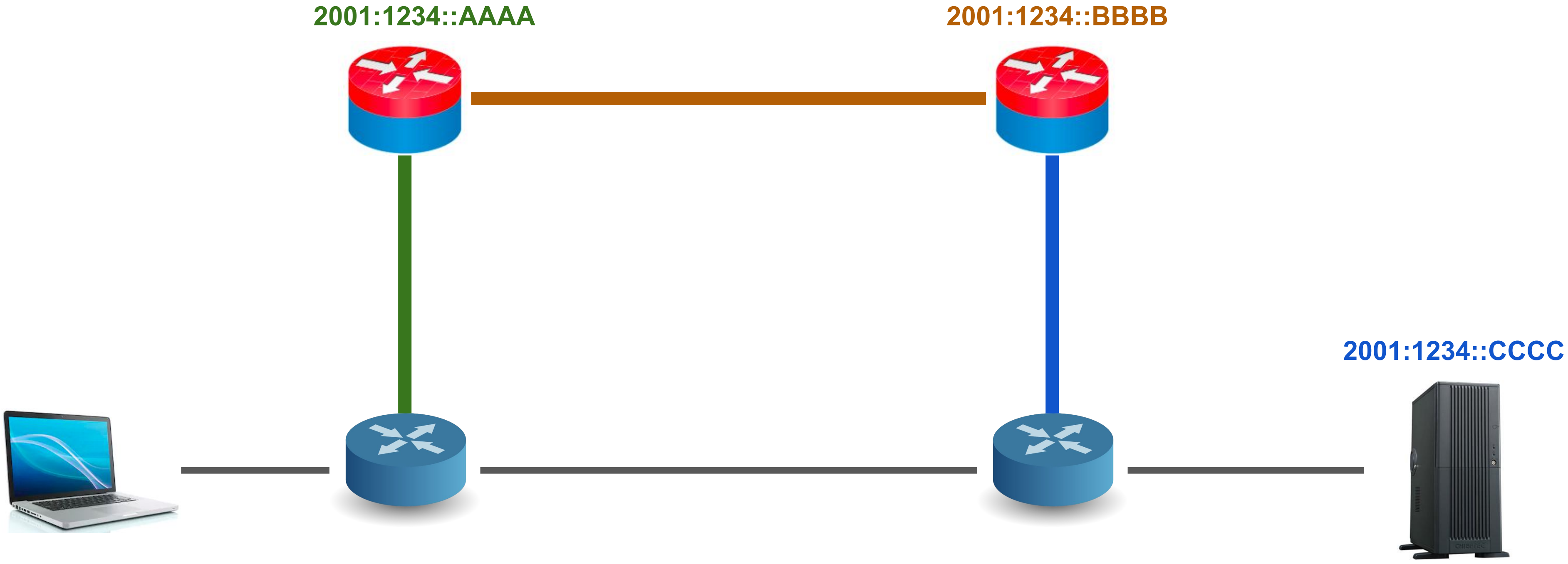
SRv6: IP**v6 S**egment **R**outing

Pipes : Unix **pipe**line

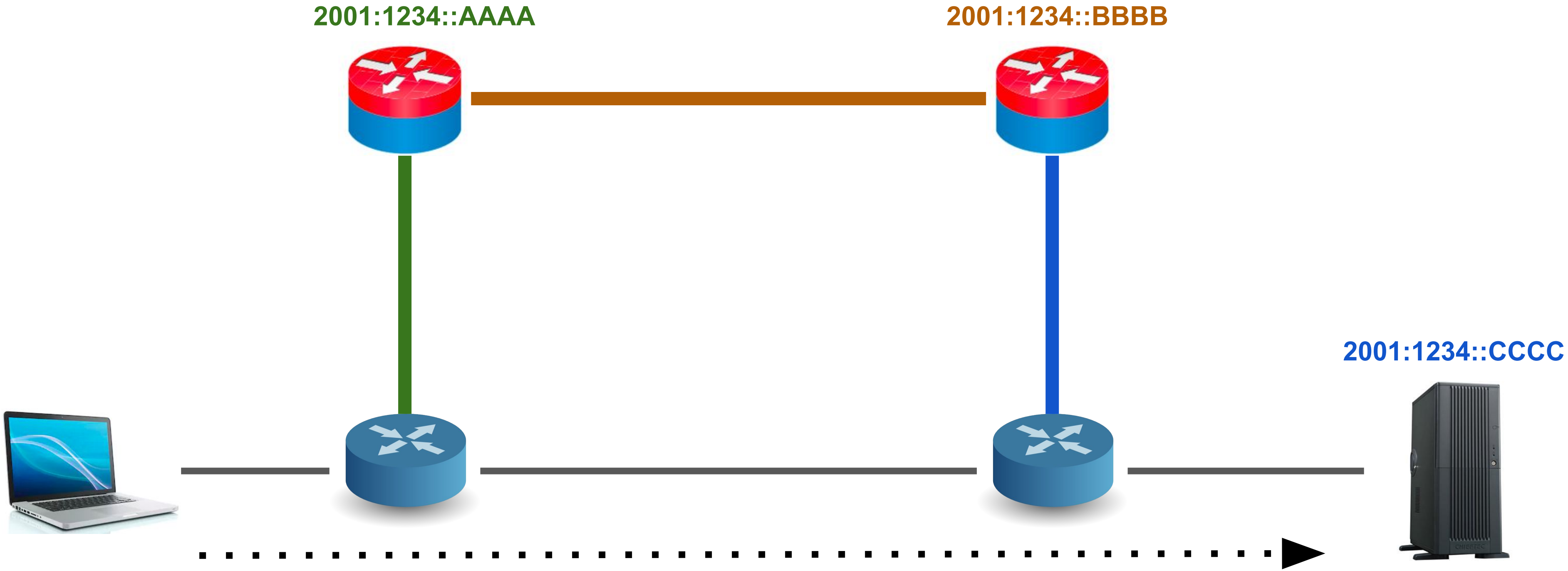
IPv6 **S**egment **R**outing

- IPv6 flavor of a modern variant of the **source routing** paradigm
- Extension header, named **S**egment **R**outing **H**eader (SRH)
- Each **segment** is an IPv6 address representing a **node or link to traverse**

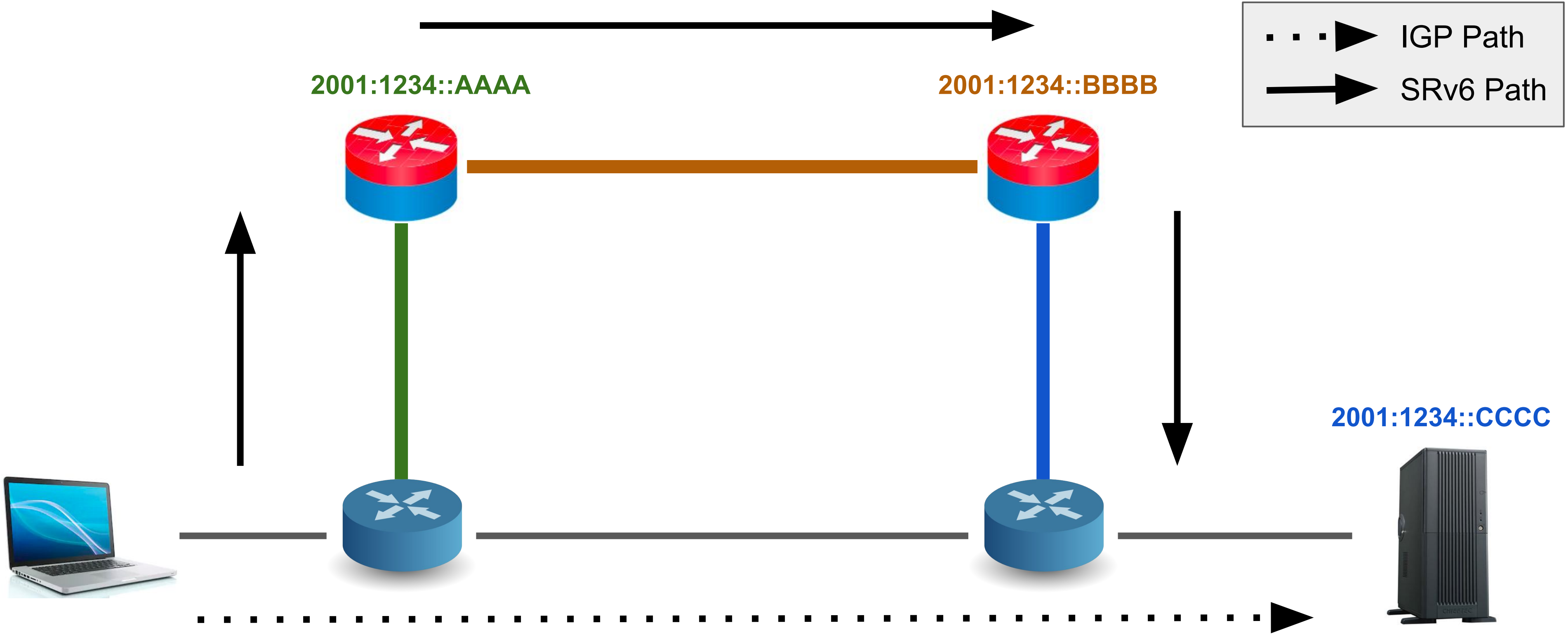
IPv6 Segment Routing



IPv6 Segment Routing



IPv6 Segment Routing



IPv6 **S**egment **R**outing **H**eader (**SRH**)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Header   |  Hdr Ext Len  | Routing Type  | Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Last Entry   |     Flags     |              Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[0] (128 bits IPv6 address)            |
|                      2001:1234::CCCC                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
                              ...
|                                                               |
|                      2001:1234::BBBB                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[n] (128 bits IPv6 address)            |
|                      2001:1234::AAAA                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                             //
//         Optional Type Length Value objects (variable)       //
//                                                             //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

IPv6 **S**egment **R**outing **H**eader (**SRH**)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Header   |  Hdr Ext Len  | Routing Type  | Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Last Entry   |     Flags     |              Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[0] (128 bits IPv6 address)            |
|                      2001:1234::CCCC                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
                              ...
|                      2001:1234::BBBB                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[n] (128 bits IPv6 address)            |
|                      2001:1234::AAAA                          |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                             //
//         Optional Type Length Value objects (variable)       //
//                                                             //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Traffic Class |           Flow Label                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Payload Length        |  Next Header  |   Hop Limit   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                         Source Address                        +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                      Destination Address                      +
|                                                               |
+                       2001:1234::AAAA                         +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

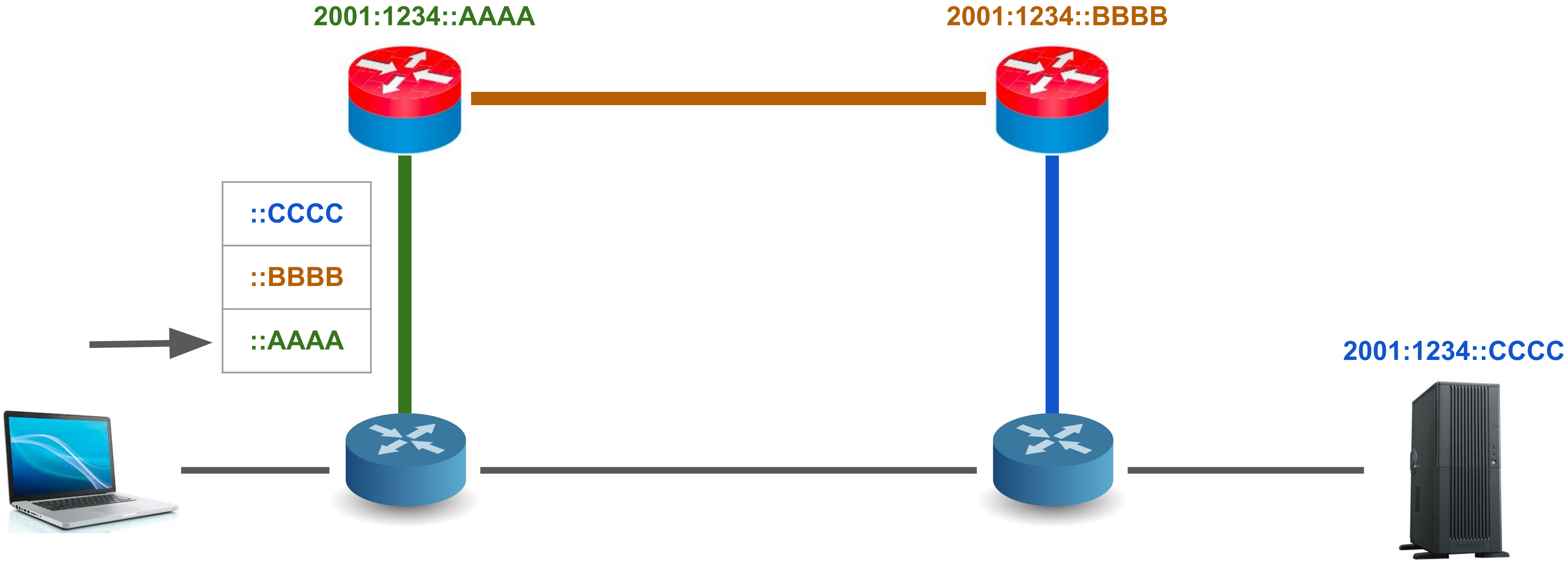
IPv6 **S**egment **R**outing **H**eader (**SRH**)

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Header   |  Hdr Ext Len  | Routing Type  | Segments Left |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Last Entry   |     Flags     |              Tag              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[0] (128 bits IPv6 address)            |
|                     2001:1234::CCCC                           |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                                                               |
                              ...
|                                                               |
|                     2001:1234::BBBB                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|            Segment List[n] (128 bits IPv6 address)            |
|                     2001:1234::AAAA                           |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
//                                                             //
//         Optional Type Length Value objects (variable)       //
//                                                             //
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

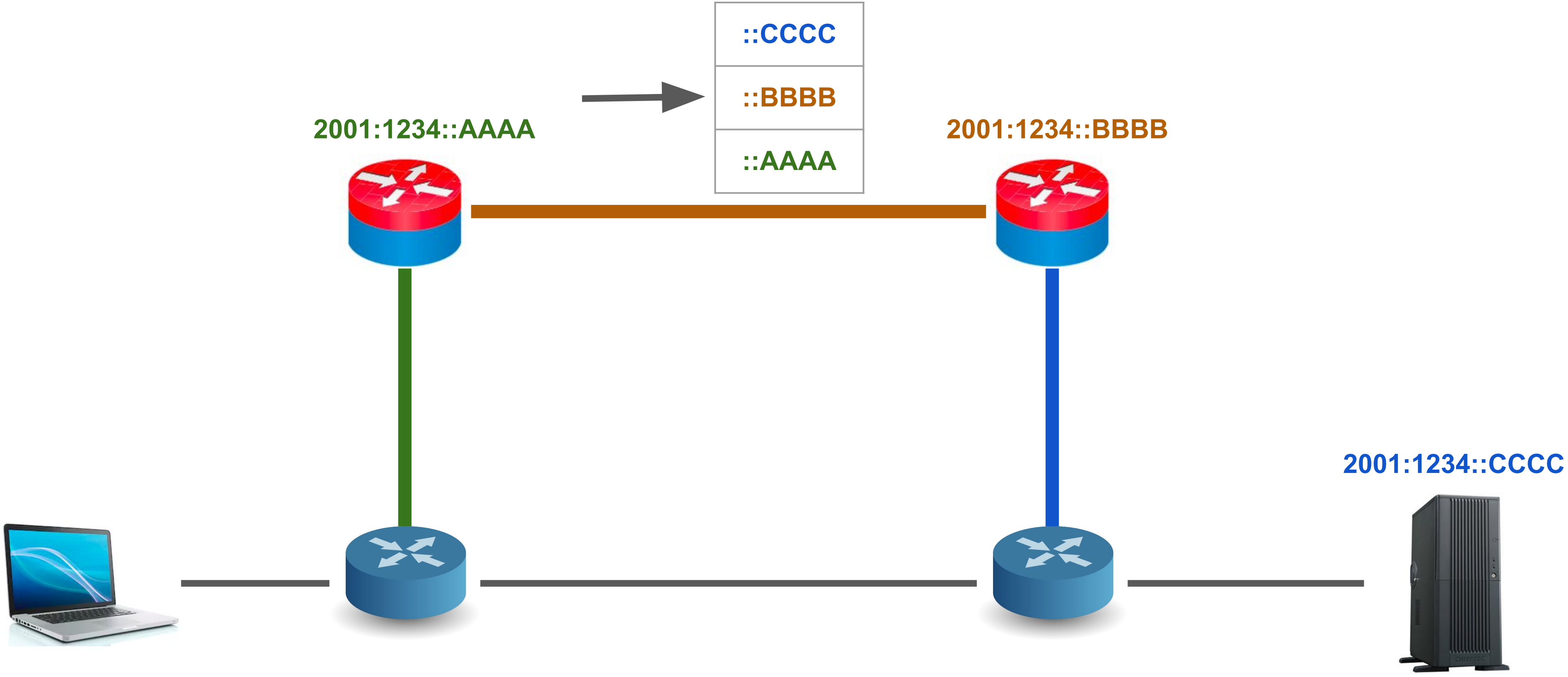
→

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version| Traffic Class |           Flow Label                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Payload Length        |  Next Header  |   Hop Limit   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                         Source Address                        +
|                                                               |
+                                                               +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                                                               +
|                                                               |
+                      Destination Address                      +
|                                                               |
+                      2001:1234::BBBB                          +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

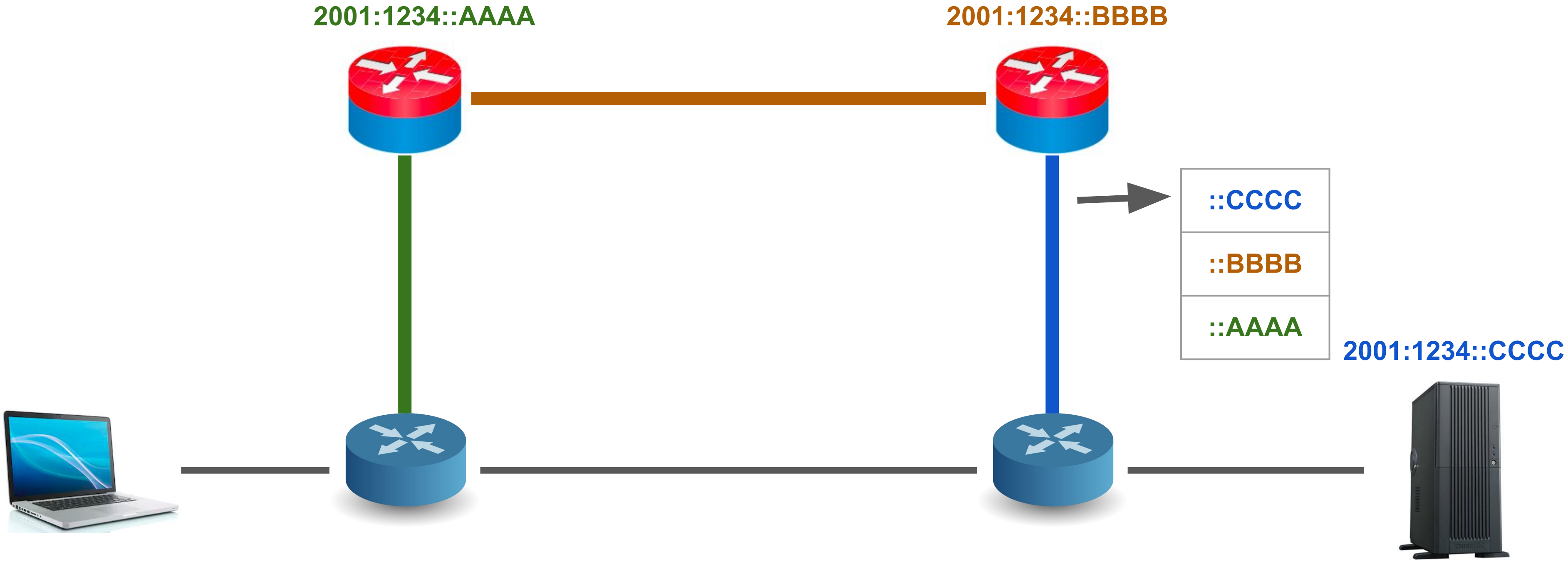
IPv6 Segment Routing



IPv6 Segment Routing



IPv6 Segment Routing



Unix **pipe**line

fab@x240-fab: ~

```
fab@x240-fab:~$ echo "I love dogs!"
I love dogs!
fab@x240-fab:~$ 
```

Unix **pipe**line

fab@x240-fab: ~

```
fab@x240-fab:~$ echo "I love dogs!"
I love dogs!
fab@x240-fab:~$ echo "I love dogs!" | base64
SSBsb3ZlIGRvZ3MhCg==
fab@x240-fab:~$
```

Unix **pipe**line

```
fab@x240-fab:~$ echo "I love dogs!"
I love dogs!
fab@x240-fab:~$ echo "I love dogs!" | base64
SSBsb3ZlIGRvZ3MhCg==
fab@x240-fab:~$ echo "I love dogs!" | base64 | sed 's/RvZ3/NhdH/g'
SSBsb3ZlIGNhdHMhCg==
fab@x240-fab:~$
```

Unix **pipe**line

```
fab@x240-fab: ~
fab@x240-fab:~$ echo "I love dogs!"
I love dogs!
fab@x240-fab:~$ echo "I love dogs!" | base64
SSBsb3ZlIGRvZ3MhCg==
fab@x240-fab:~$ echo "I love dogs!" | base64 | sed 's/RvZ3/NhdH/g'
SSBsb3ZlIGNhdHMhCg==
fab@x240-fab:~$ echo "I love dogs!" | base64 | sed 's/RvZ3/NhdH/g' | base64 -d
I love cats!
fab@x240-fab:~$
```

Unix **pipe**line

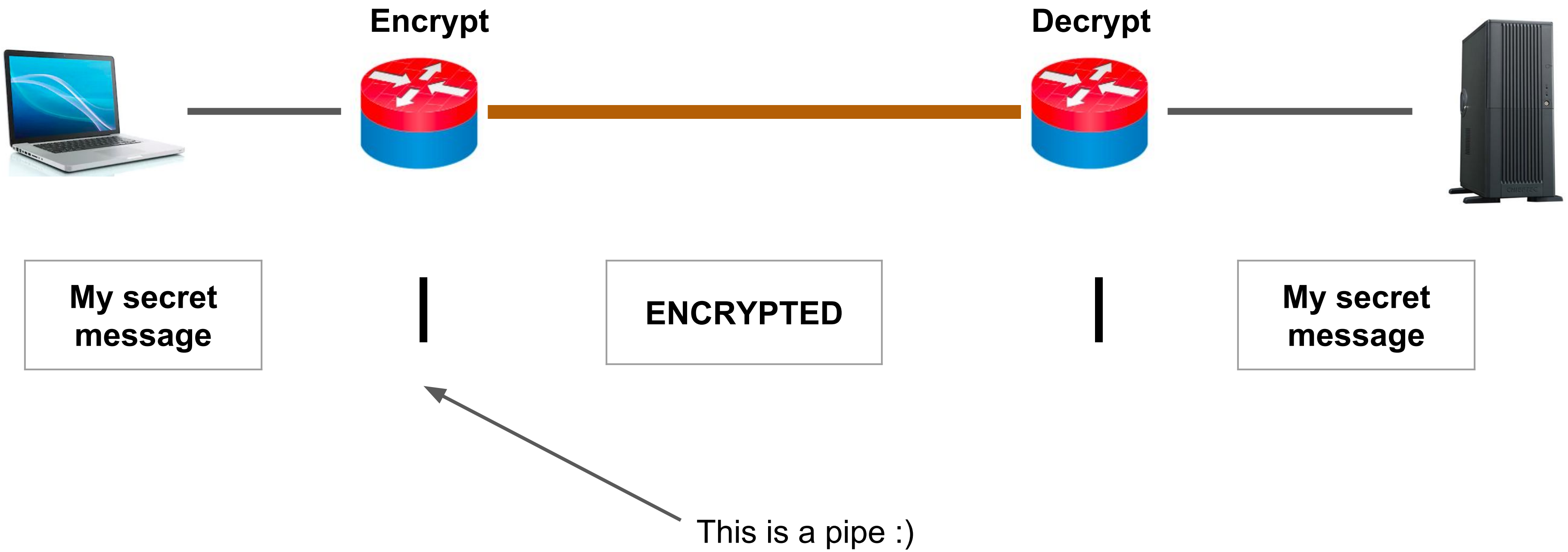
fab@x240-fab: ~

```
fab@x240-fab:~$ echo "I love dogs!"
I love dogs!
fab@x240-fab:~$ echo "I love dogs!" | base64
SSBsb3ZlIGRvZ3MhCg==
fab@x240-fab:~$ echo "I love dogs!" | base64 | sed 's/RvZ3/NhdH/g'
SSBsb3ZlIGNhdHMhCg==
fab@x240-fab:~$ echo "I love dogs!" | base64 | sed 's/RvZ3/NhdH/g' | base64 -d
I love cats!
fab@x240-fab:~$
```

SRv6 + Pipes = SRv6Pipes !

Leverage IPv6 Segment Routing to allow the user to **choose and apply** a **chain of functions** to the **payload**.

SRv6 + pipes = SRv6Pipes !

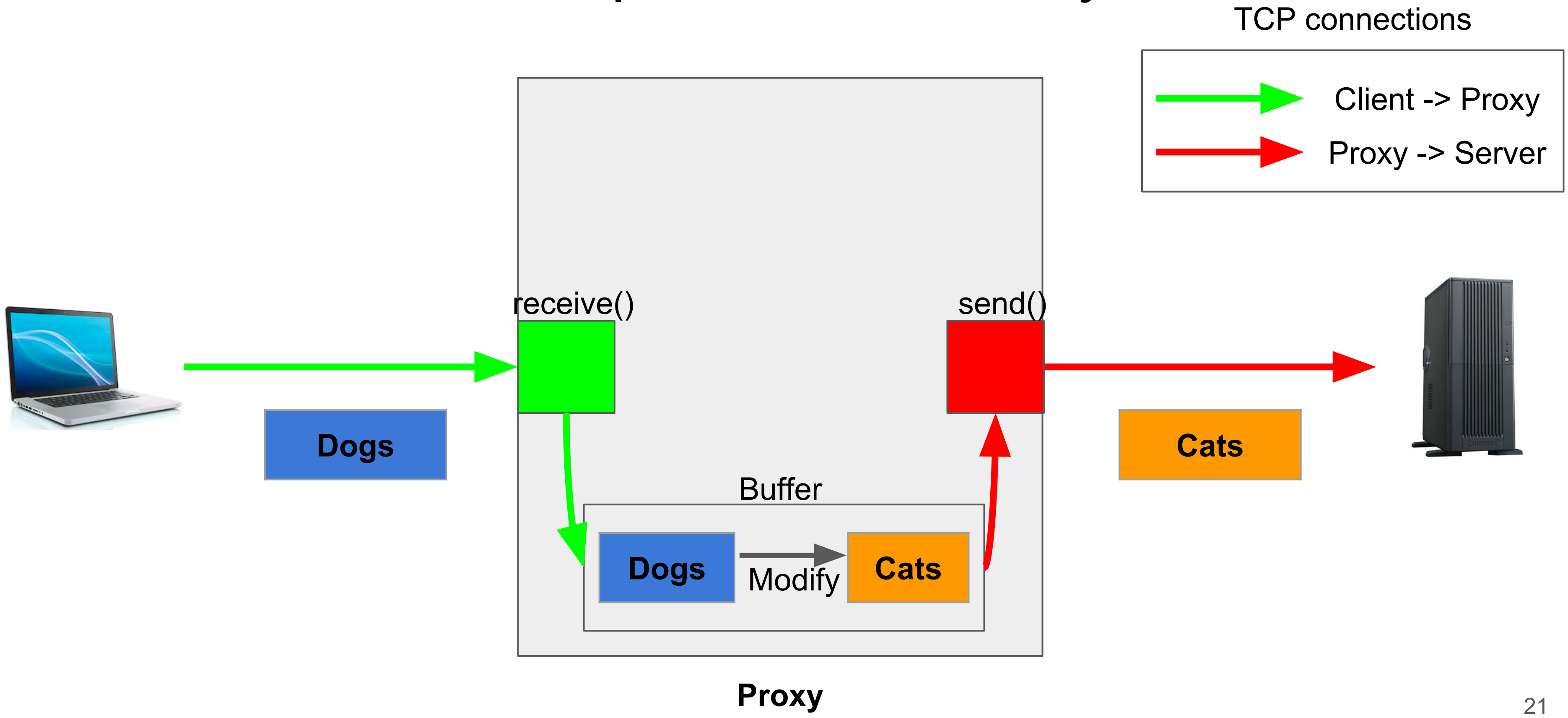


Enabling in-network **bytestream** functions

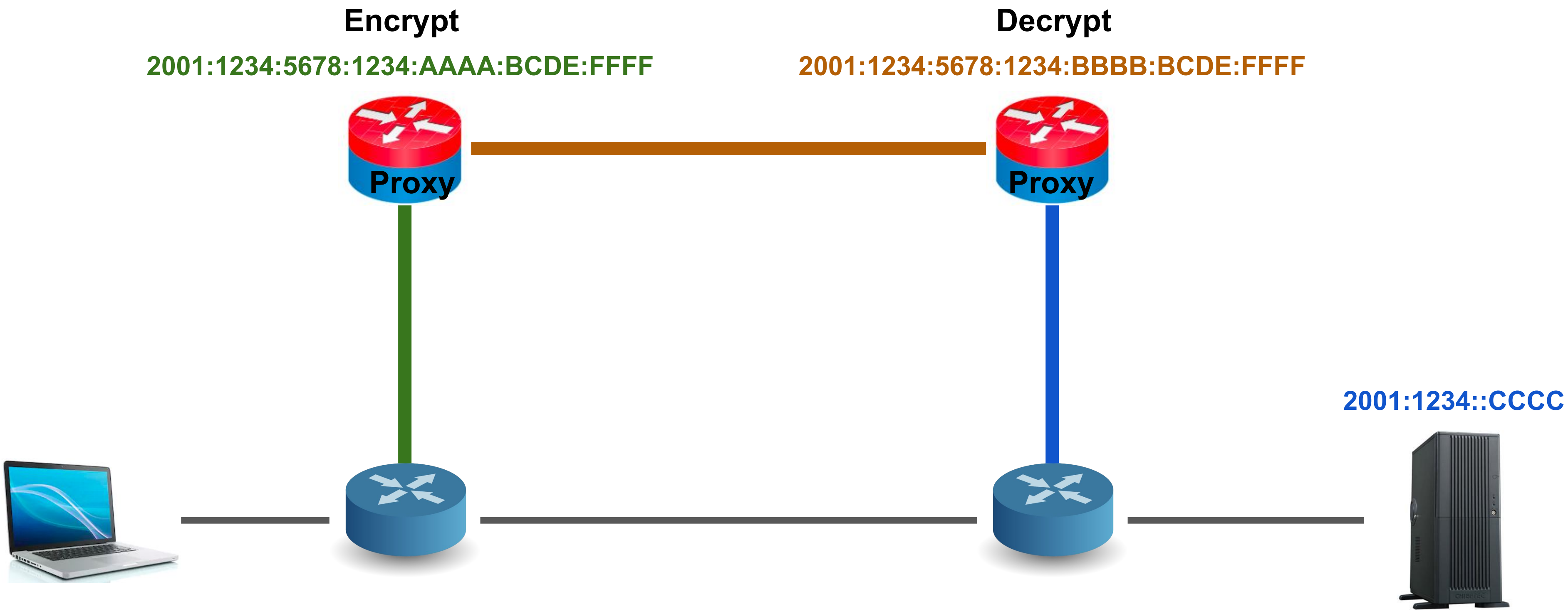
Middleboxes can perform two different types of network functions:

- **per-packet** functions: operate on a per-packet basis
  - **N**etwork **A**ddress **T**ranslation (**NAT**), stateless firewall….
  - operate on the network and sometimes transport header
  - **simple**
- **per-bytestream** functions: operate on the payload of the TCP packets
  - compression, encryption, transcoding...
  - **reorder** the received TCP packets and often **modify the payload** of TCP packets
  - include an almost complete **TCP implementation**
  - **complex**

Transparent TCP Proxy



SRv6Pipes: almost ready!



SRv6Pipes : modular transformation

The client should be able to create **any chain of functions**.

How to represent a function ?

- One proxy per function is too expensive
- One proxy should be able to perform several functions

⇒ The client must be able to **inform the proxy** about the **function**

Encoding Functions and Parameters

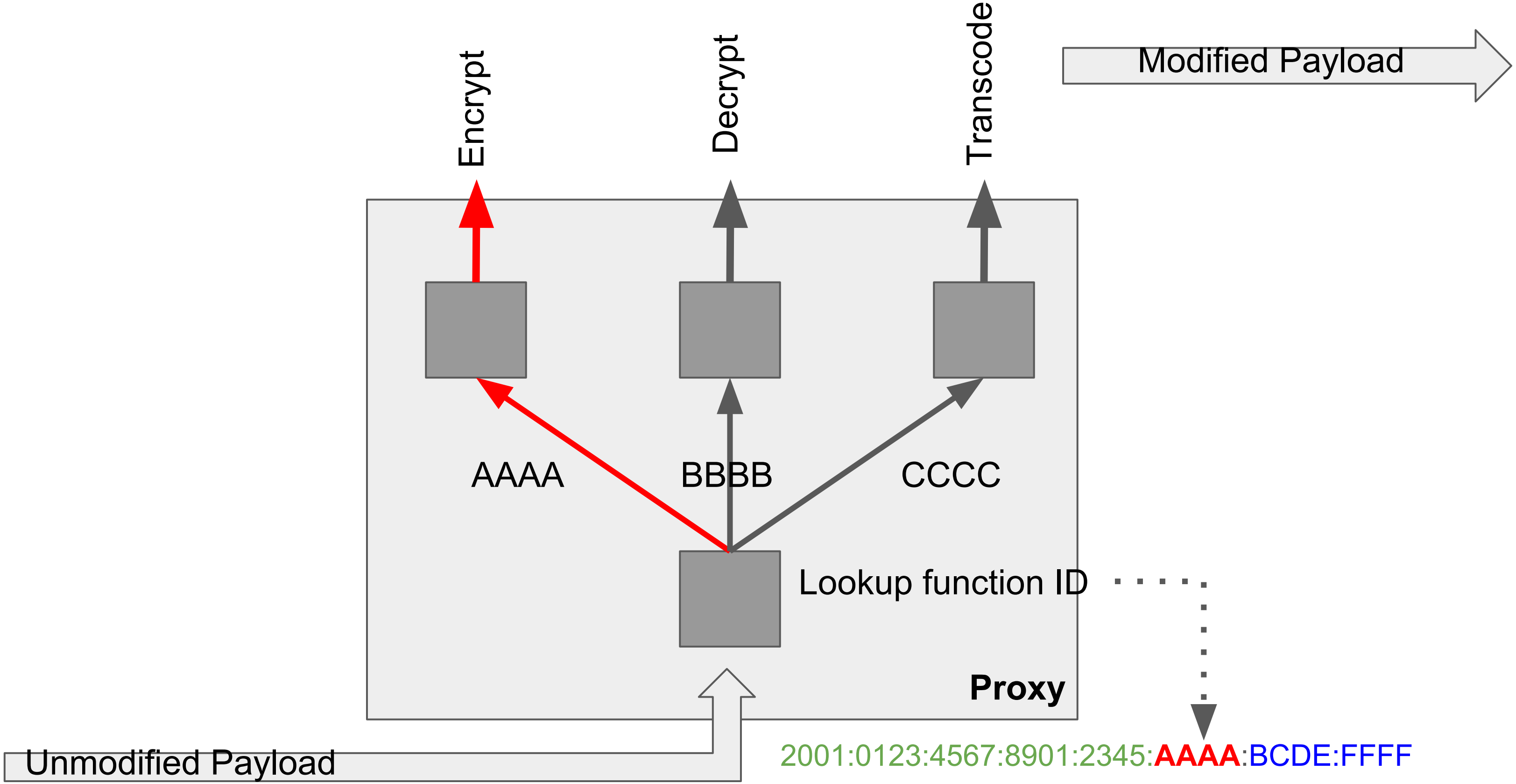
- We leverage the large addressing space available in IPv6.
- Each **proxy announces** one or more **IPv6 prefixes**
- Allocate a given amount of bits to encode the **identifier of the function**
- The remaining **low order bits** are used to specify **parameters** of the function

2001:0123:4567:8901:2345:AAAA:BCDE:FFFF

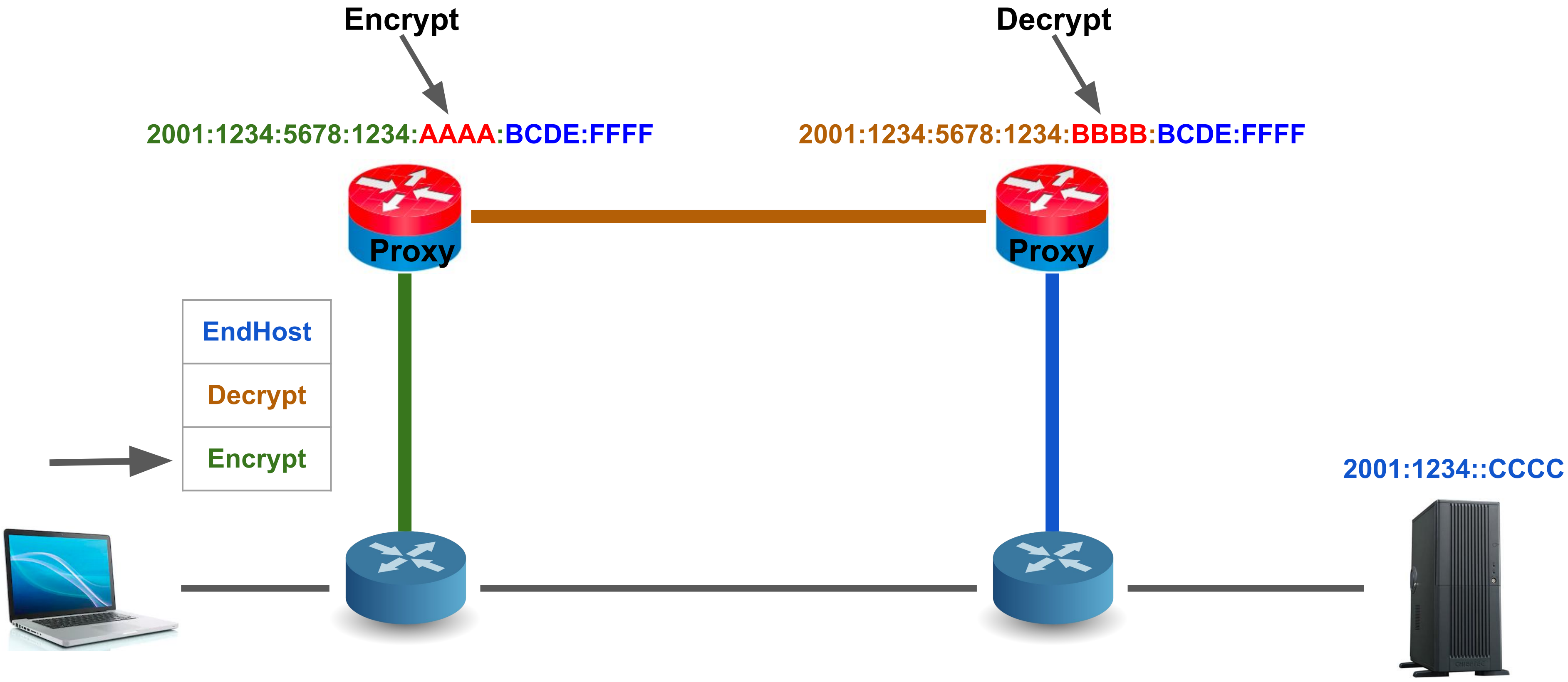
- **2001:0123:4567:8901:2345/80 : Proxy range**
- **AAAA : Identifier of the function**
- **BCDE:FFFF : Parameters of the function**

[1] C. filsfils et al. SRv6 Network Programming. IETF

Encoding Functions and Parameters



SRv6Pipes: the big picture



SRv6Pipes : other design points

How is the **return traffic** handled?

⇨ We insert a **T**ype **L**ength **V**alue (TLV) object containing the return path in the original SRH.

How does the client get informations about the **addresses of the proxies**?

⇨ We modify the **DNS** resolver. This is detailed in [1].

[1] D. Lebrun et al. Software Resolved Networks: Rethinking Enterprise Networks with IPv6 Segment Routing. SOSR 18

SRv6Pipes : implementation

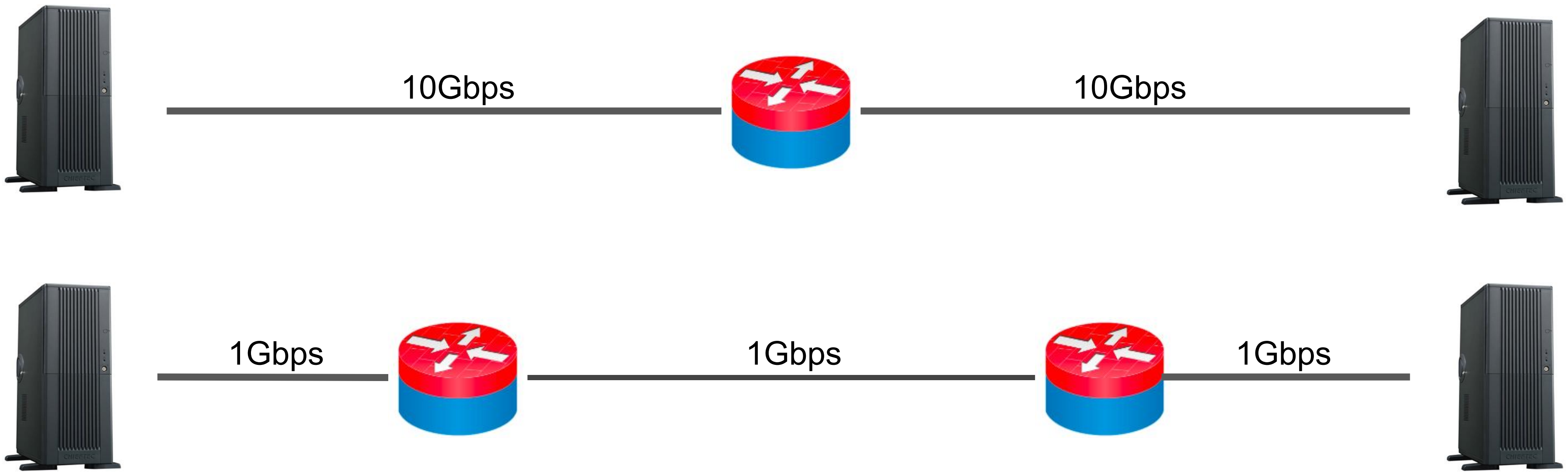
Implementation details :

- Modification of the Linux **kernel** version 4.16.0
- Modification of **iproute2**
- Implementation of the **proxy** (~1000 C lines)
    - Uses **NFQUEUE** to intercept the SYN and extract the SRH
    - Leverages **TPROXY** to establish and **transparent** connection
    - Uses **ip6tables**
    - Allows to load **dynamic modules** to support **new functions**

Runs on commodity hardware.

Code is available on: http://segment-routing.org/index.php/SRv6Pipes

Implementation : performance evaluation



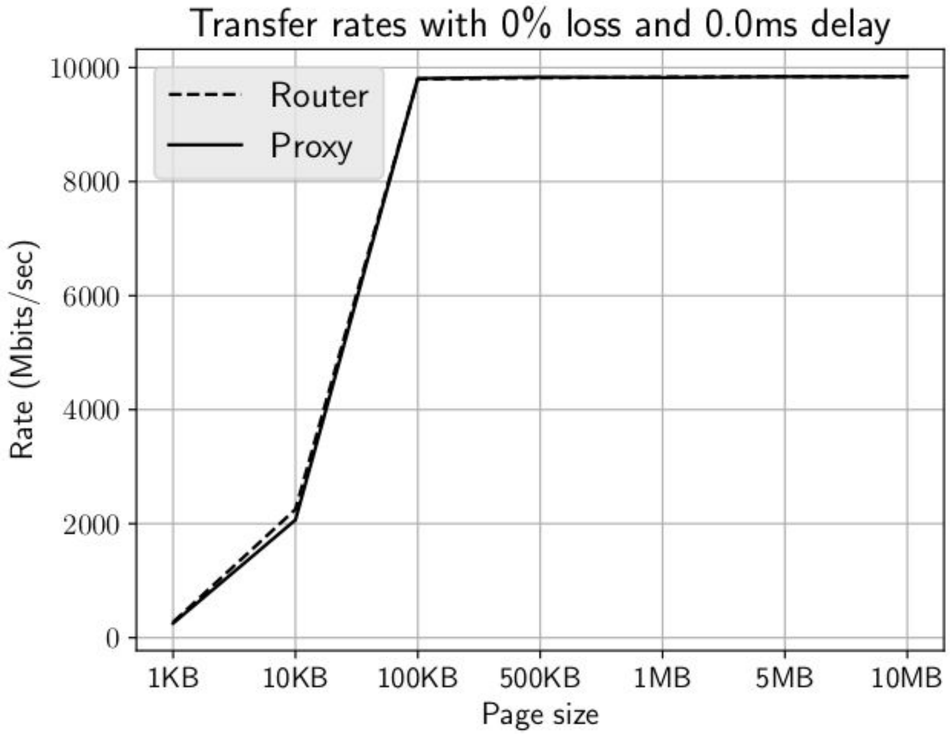
Client : 2,53Ghz Intel Xeon 16GB RAM - Debian Stretch - Kernel 4.16 - wrk 4.0.2-5
Middleboxes/Server: 2,53Ghz Intel Xeon 8GB RAM - Debian Stretch - Kernel 4.16 - lighttpd 1.4.35

Measurements Results : no loss/delay

- 1 middlebox acting as proxy/router
- 200 clients downloading web pages of a given size.

- For 10MB files :
  - Proxy : ~9840Mb/s
  - Router : ~9840Mb/s
- For 1KB files:
  - Proxy : 253Mb/s
  - Router : 272Mb/s

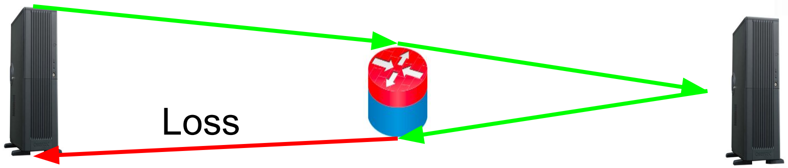
⇨ Cost of establishment.

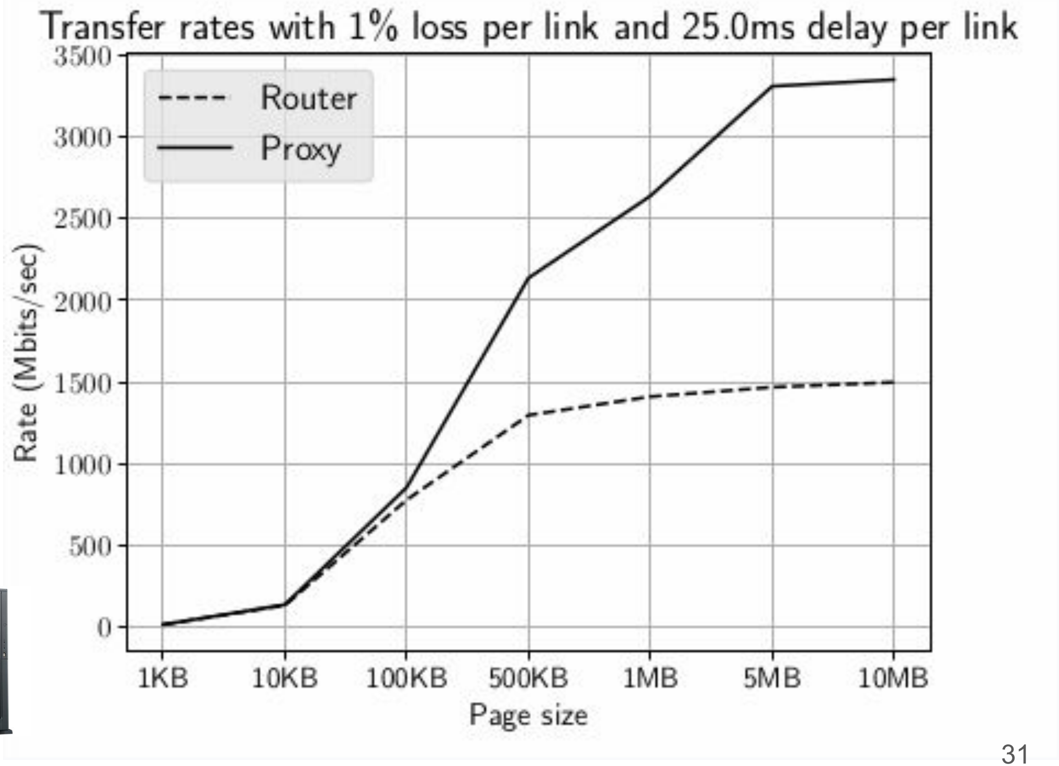


Measurements Results : with loss and delay

- Same setup
- Loss: 1% per link
- Delay 1% per link

- Our proxy acts as a **P**erformance **E**nchancing **P**roxy (**PEP**)

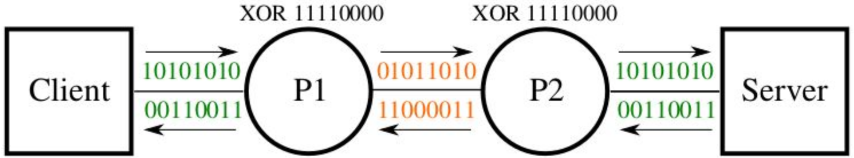


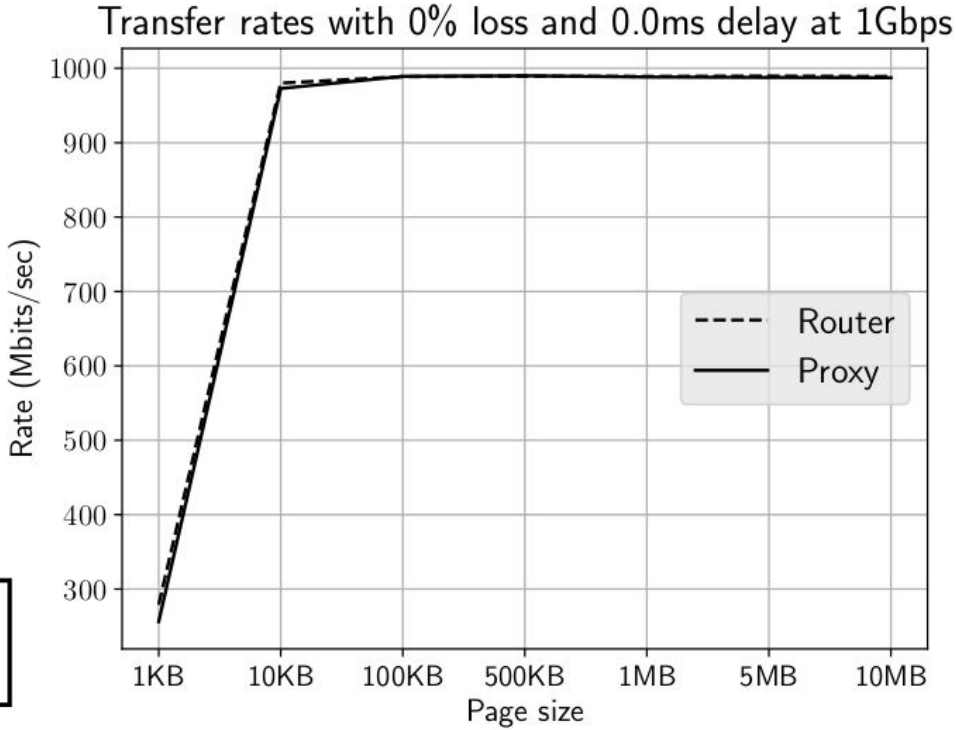


Measurements Results : no loss/delay - 2 proxies

- 2 proxies/routers
- Applying a XOR function

- No significant overhead





Conclusion

SRv6 pipes :

- Middleboxes are **explicitly** exposed
- **Flexibility** for the Network Operators
- In-network **per-bytestream** and **per-packet** functions
- **New use cases** for IPv6 Segment Routing
- **Implemented** in the Linux kernel and available today!

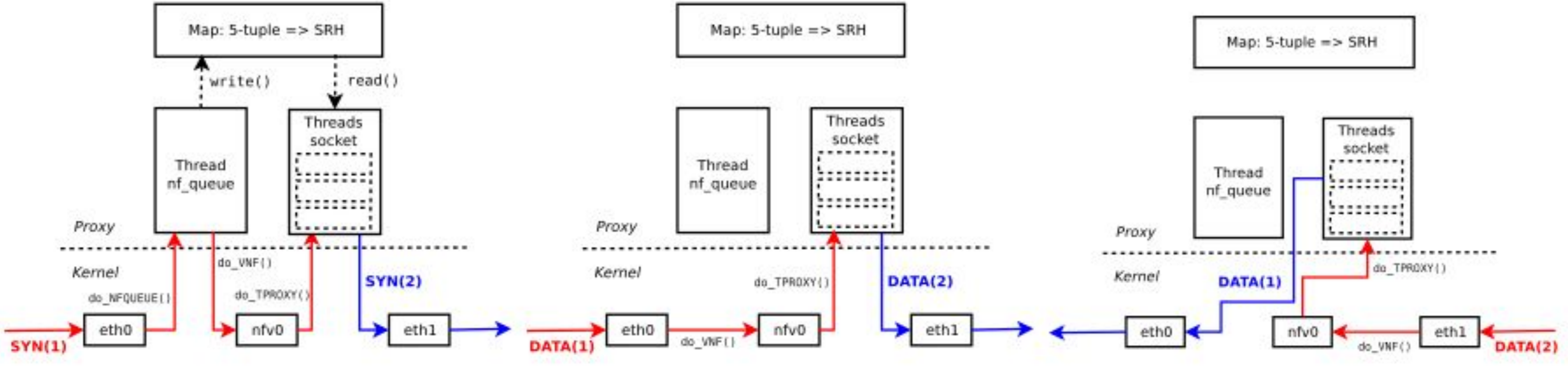
Thank you!

Try it now! http://segment-routing.org/index.php/SRv6Pipes

Fabien Duchêne <fabien.duchene@uclouvain.be>

Measurements Results: CPU intensive functions

Implementation details



(a) Traversal of a SYN packet through the proxy. The SRH is recorded for the 5-tuple.

(b) Traversal of data packets.

(c) Traversal of return packets.