Offloading SCION Packet Forwarding to XDP BPF

Lars-Christian Schulz, David Hausheer
Networks and Distributed Systems Lab
Otto-von-Guericke University Magdeburg

3. KuVS Fachgespräch "Network Softwarization"
April 7, 2022

What is SCION?

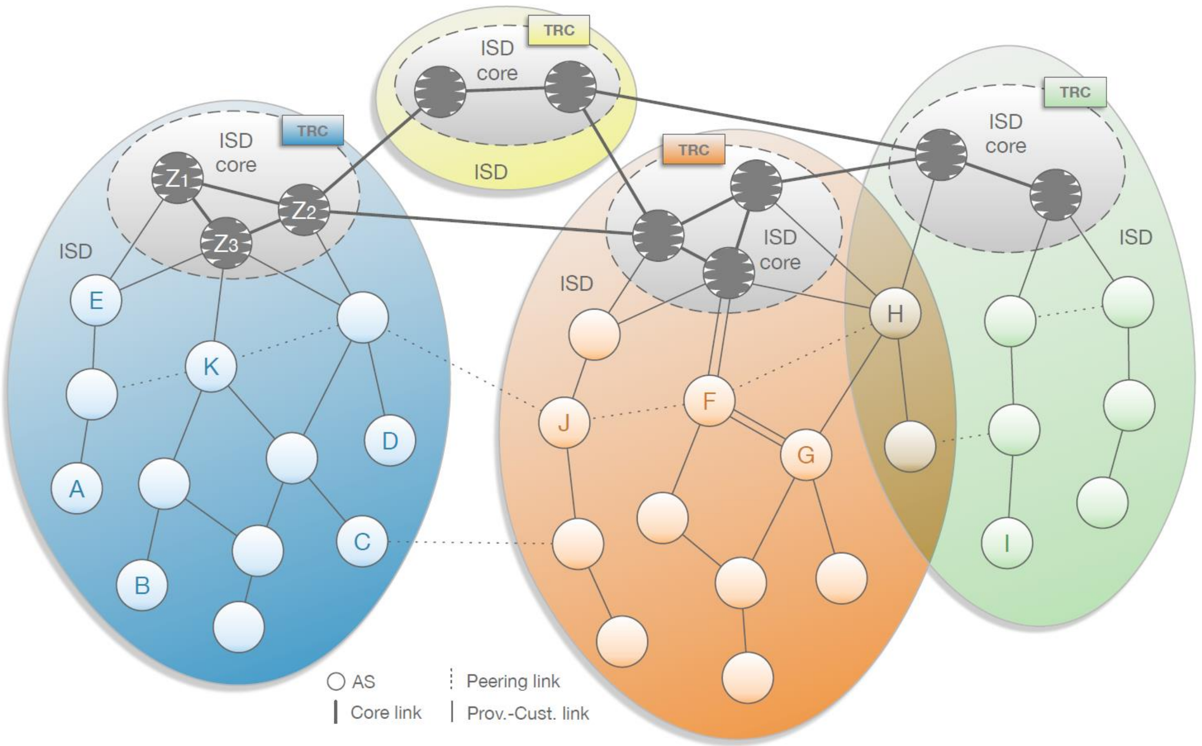
- SCION = **S**calability, **C**ontrol, and **I**solation **O**n **N**ext-generation networks
- New Internet Architecture intended to replace BGP
- Clean-slate approach with strong focus on reliability, security, and transparency

https://scion-architecture.net/

- Developed at ETH Zurich
- Real-world deployments exist
- SCIONLab global research network https://www.scionlab.org/

SCION Basics

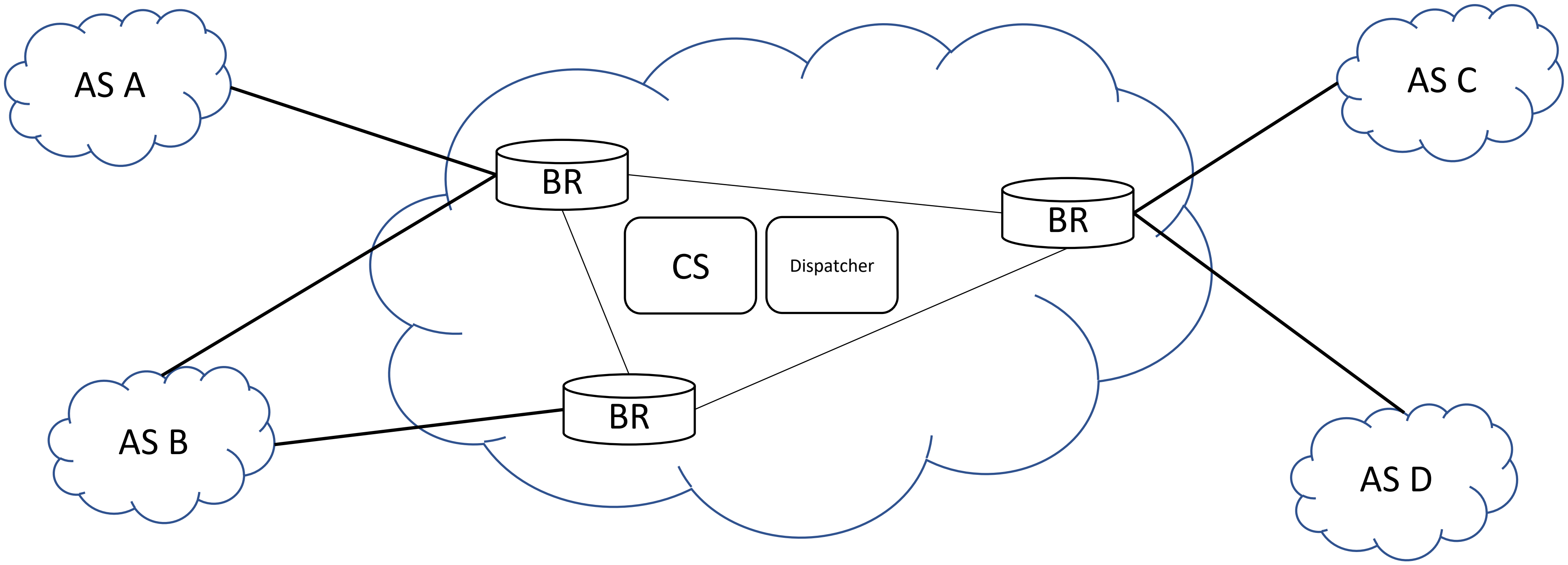
- SCION ASes (Autonomous Systems) are grouped by Isolation Domain (ISD)
  - **ISDs** operate independently from one another (different trust roots, etc.)
  - **Core Ases** manage the ISD and provide links to other ISDs
- SCION end hosts are aware of the AS-level forwarding path
  - Accomplished through the use of Packet-Carried Forwarding State
  - Path choice is limited to predefined **path segments**
- Path segment construction: **Beaconing**
  - Inter-ISD: Core-Segments
  - Intra-ISD: Up-/Down-Segments from core ASes to leaf ASes



From A. Perrig, P. Szalachowski, R.M. Reischuk, L. Chuat
SCION: A Secure Internet Architecture, 2017
("The SCION Book")

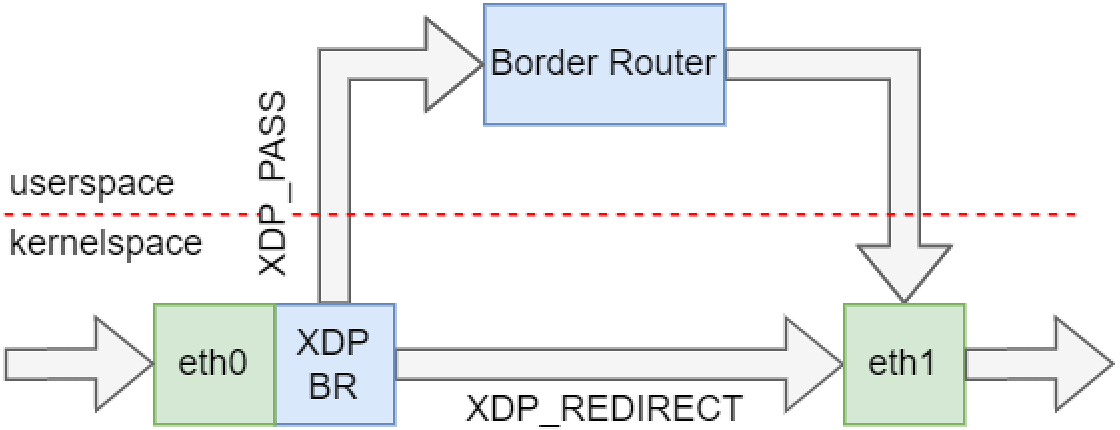
Anatomy of a SCION AS (as currently implemented in SCIONLab)

- Open-source implementation of SCION in Go at https://github.com/scionproto/scion
- A SCION AS contains control service, border router(s), and a dispatcher



Accelerate the Border Router with XDP

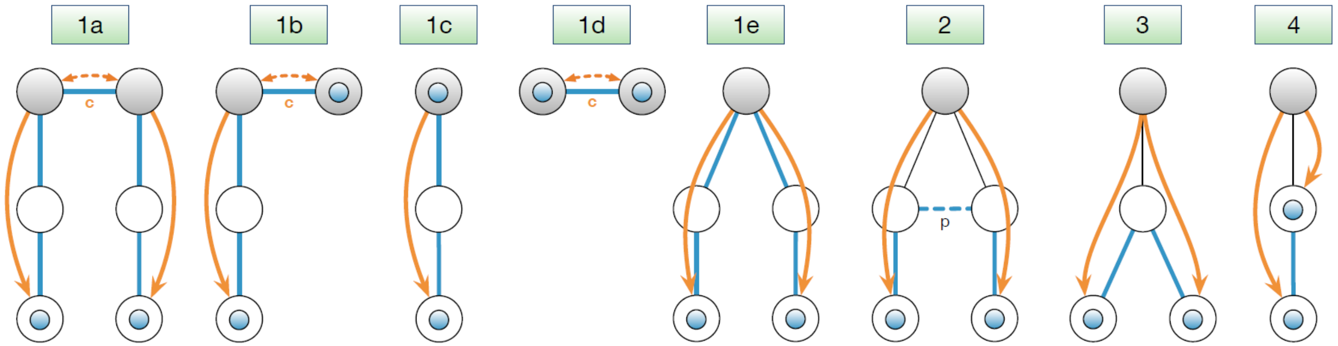
- Idea: Create an **XDP fast-path** for handling the most common packet types
- Existing border router remains as **slow-path** for less common packet types and packets that require special processing



- Challenge: SCION requires cryptographic verification of hop fields
- eBPF programming environment is rather restrictive

SCION Path Construction

- Up to three path segments are stitched together to form an AS-level end-to-end path
- SCION Header =
  - Common Header
  - Address Header
  - Forwarding Path
- Currently 5 path types:
  - Empty: AS internal only
  - OneHop: For bootstrapping
  - **SCION: "Standard" SCION**
  - EPIC, COLIBRI: Experimental extensions
- Standard SCION path consists of:
  - Up to three info fields, one for each path segment
  - A least two hop fields per path segment



From A. Perrig, P. Szalachowski, R.M. Reischuk, L. Chuat
SCION: A Secure Internet Architecture, 2017
("The SCION Book")

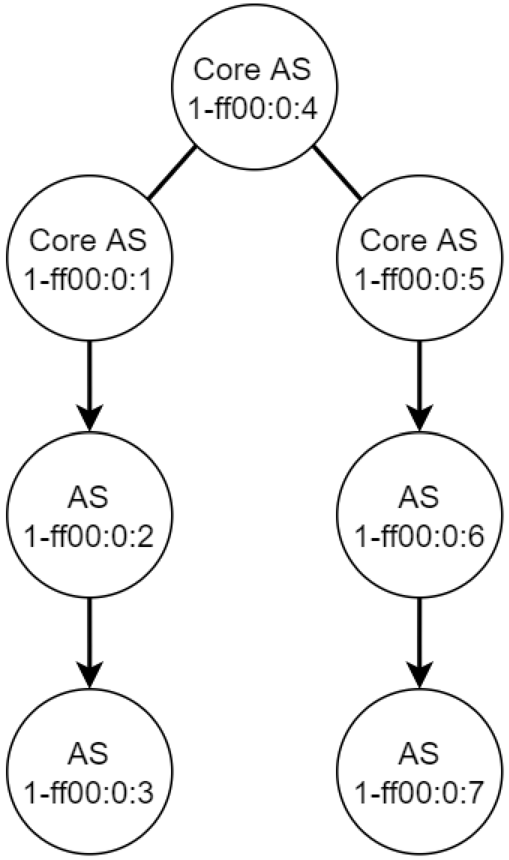
SCION Data Plane: An Example

- Let's send a UDP packet from host A in AS 1-ff00:0:3 to host B in AS 1-ff00:0:7
- Host A assembles a path to AS 1-ff00:0:7 and sends the following packet to its BR:

```
###[ Ethernet ]###
###[ IP ]###
###[ UDP ]###
###[ SCION ]###
 Version   = 0
    QoS        = 0x0
    FlowID     = 0x1
    NextHdr    = UDP
    HdrLen     = 172 bytes
    PayloadLen= 12
    PathType   = SCION
    DT         = IP
    DL         = 4 bytes
    ST         = IP
    SL         = 4 bytes
    RSV        = 0
    DstISD     = 3
    DstAS      = ff00:0:7
    SrcISD     = 1
    SrcAS      = ff00:0:3
    DstHostAddr= B's IP
    SrcHostAddr= A's IP
    \Path      \
```

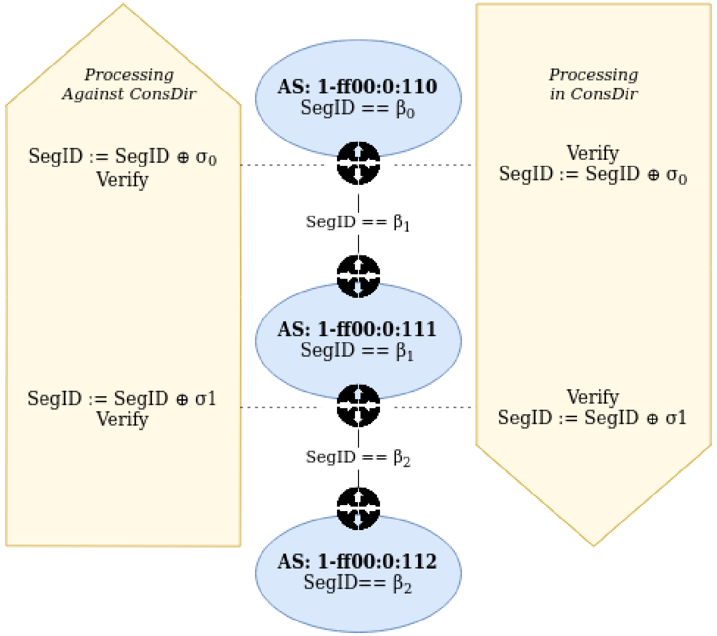
```
|###[ SCION Path ]###
|  CurrINF    = 0
|  CurrHF     = 0
|  RSV        = 0
|  Seg0Len    = 3
|  Seg1Len    = 3
|  Seg2Len    = 3
|  \InfoFields\
|   |###[ Info Field ]###
|   |  Flags     =
|   |  RSV       = 0
|   |  SegID     = 0x6b1c
|   |  Timestamp = 2022-04-06 14:59:06
|   |###[ Info Field ]###
|   |  Flags     =
|   |  RSV       = 0
|   |  SegID     = 0xf68c
|   |  Timestamp = 2022-04-06 14:59:03
|   |###[ Info Field ]###
|   |  Flags     = C
|   |  RSV       = 0
|   |  SegID     = 0x557f
|   |  Timestamp = 2022-04-06 14:59:03
|  \HopFields \
```

```
|###[ Hop field ]###
|  ConsIngress= 1
|  ConsEgress= 0
|  MAC        = 0xd8cbd8a708fd
|###[ Hop field ]###
|  ConsIngress= 1
|  ConsEgress= 2
|  MAC        = 0x91d85b07bbff
|###[ Hop field ]###
|  ConsIngress= 0
|  ConsEgress= 2
|  MAC        = 0x58e5ba761ff4
|###[ Hop field ]###
|  ConsIngress= 1
|  ConsEgress= 0
|  MAC        = 0xbb8f49c36c1
|###[ Hop field ]###
|  ConsIngress= 2
|  ConsEgress= 1
|  MAC        = 0x37f7b74b0436
|###[ Hop field ]###
[…]
###[ UDP ]###
[…]
```



Hop Field MAC Verification

- Hosts can only use path segments obtained from path servers
- Hop fields are authenticated by **Message Authentication Codes** (MAC)
- MACs are chained within a path segment to prevent reordering, removal, or insertion of hops
- MAC chaining is achieved by including the SegID field in the MAC computation and updating it with (part of) the MAC computed at the current hop
- BR compares the MAC it has computed to the MAC in the hop field
  - If they do not match, the packet is dropped
  - Otherwise it is forwarded to the next BR or the dispatcher
- MAC computation uses AES-CMAC
  - Based on AES
  - Keyed with per-AS secret symmetric key
  - SCION selling point: AES in hardware is faster than memory access (< 50 ns)
  - Unfortunately no access to AES hardware from eBPF
  - We have to rely on a software implementation of AES



https://github.com/scionproto/scion/blob/master/doc/protocols/fig/seg-id-calculation.png

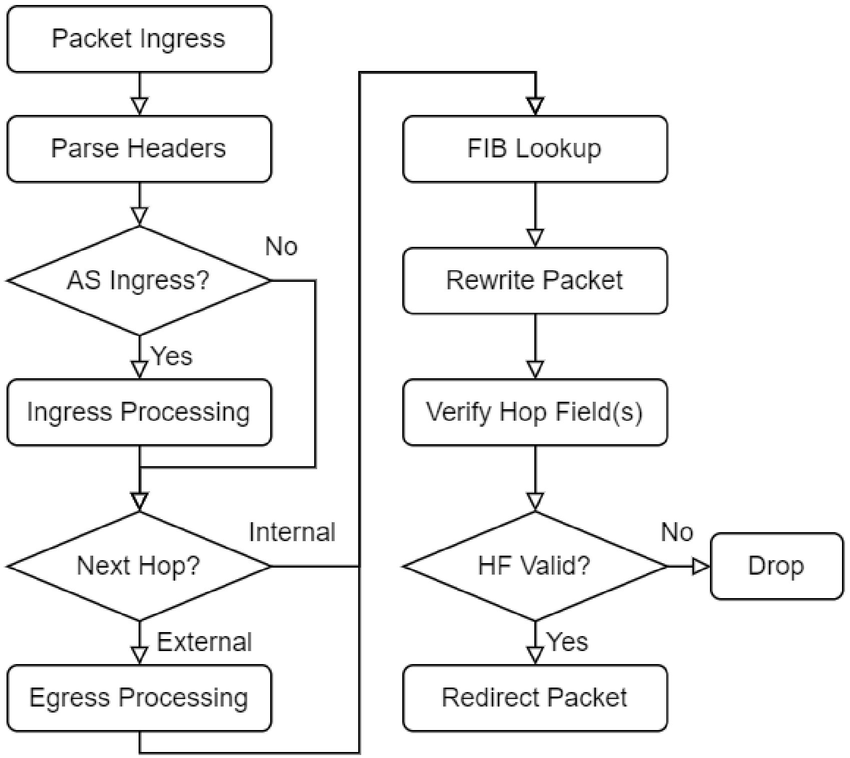
SCION XDP BR

Parsing

- Non-SCION packets, packets with non-SCION path format and packets with router alert flags are passed to the regular network stack
- Map underlay UDP connection to AS ingress IFID

SCION Path Processing

- Update SCION path headers depending on path direction and whether this is the first and/or the last BR in the current AS and path segment
- Prepare input for hop field verification
- Extract AS egress IFID from hop field



Destination Lookup

- IP of next BR is determined
- Next hop IP is looked up in kernel‘s FIB

Rewrite Packet

- Rewrite the packet only when we are sure it does not need to be passed to the regular BR
- Update checksums

Verify Hop Field

- Hop fields are verified last, so temporary variables can be cleaned from stack before AES function is invoked

Redirection

- Redirect to BR egress interface

OTTO VON GUERICKE UNIVERSITÄT MAGDEBURG
INF
FACULTY OF COMPUTER SCIENCE

SCION XDP BR Implementation

- Source code available at https://github.com/netsys-lab/scion-xdp-br
- Two binaries: **br-loader** (userspace helper) and **xdp_br.o** (actual eBPF binary)
- How to use: Run SCION AS as normal; Attach XDP/BR to all interfaces serving SCION overlay connections

```
sudo br-loader attach xdp_br.o config.toml eth0 eth1 …
```

- SCION AS configuration is read from standard topology.json configuration file
- MAC verification key has to be added manually

```
sudo br-loader key add br1-ff00_0_1-1 39OeQc0vfosfqhuhVyqxZQ==
```

- **br-loader** uses **libbpf** to load the XDP program, attach it to the selected network interfaces, and to initialize BPF maps
- In the future, the Go border router should load the XDP program by itself without the need for br-loader

config.toml

```
self = "br1-ff00_0_1-1"
topology = "gen/ASff00_0_1/topology.json"
internal_interfaces = [
   {ip = "10.2.0.0", port = 31002},
   {ip = "10.2.0.7", port = 31002}
]
```

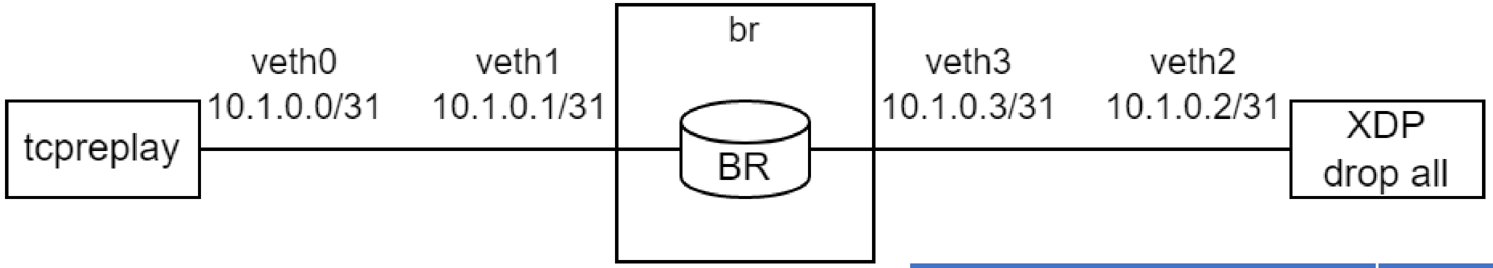
topology.json

```
"border_routers": {
 "br1-ff00_0_1-1": {
  "internal_addr": "10.2.0.0:31002",
  "interfaces": {
   "2443": {
    "underlay": {
     "public": "10.1.0.1:50000",
     "remote": "10.1.0.2:50000"
    },
    "isd_as": "1-ff00:0:2",
    "link_to": "CHILD",
    "mtu": 1472
   }
  }
 }, …
```

```
XDP Border Router br1-ff00_0_1-1
External interfaces:
 2443  veth8 local  [10.1.0.1]:50000
             remote [10.1.0.2]:50000
Sibling BR interfaces:
 2553 route to [10.2.0.2]:31004
  405 route to [10.2.0.2]:31004
  164 route to [10.2.0.4]:31006
 1305 route to [10.2.0.6]:31008
 2195 route to [10.2.0.6]:31008
Internal interfaces:
 veth0 [10.2.0.0]:31002
 veth7 [10.2.0.7]:31002
XDP-BR attached
```

Preliminary Evaluation

- Evaluated on **virtual Ethernet devices** (veths) in a **Ubuntu 21.10 VM** running on an AMD Ryzen 3700X processor
- **Traffic source:** tcpreplay
- **Traffic sink:** XDP drop program
- **Packet size:** 138 bytes
- XDP-BR is running standalone in network namespace br without the full BR
- Everything is running an a **single CPU core**

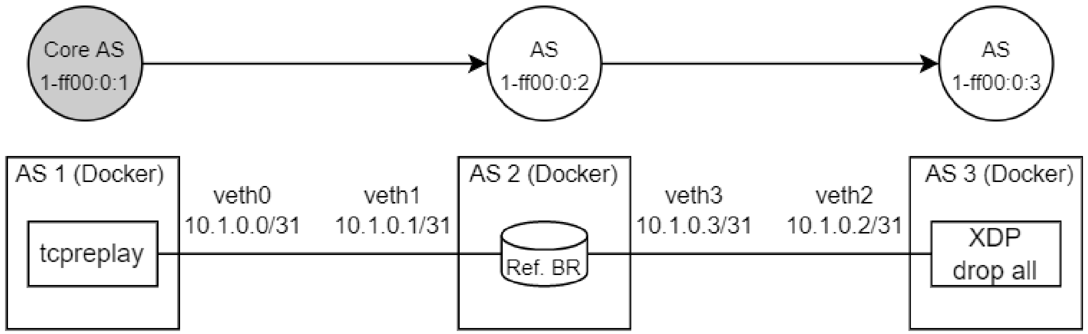


- Maximum throughput of veth is limited
- For comparison: Native bridge between veth1 and veth2
- XDP-BR without hop field verification is faster then native bridge
- Overhead for hop field verification per packet: 1.48 µs - 1.12 µs = 360 ns
- AES subroutine has a runtime of about 180 ns when running natively in userspace

Test	Throughput	Time per Packet
Direct veth0 <-> veth1	1.238 Mpps	0.81 µs
Native bridge	0.685 Mpps	1.46 µs
XDP BR with AES	0.676 Mpps	1.48 µs
XDP BR without AES	0.895 Mpps	1.12 µs

Preliminary Evaluation in Docker

- Same as before, but this time with "real" SCION ASes in Docker containers, so the reference BR is happy
- **Traffic source:** tcpreplay
- **Traffic sink:** XDP drop program
- **Packet size:** 138 bytes
- During the measurements, traffic in AS 2 is dropped to avoid a bottleneck caused by the next border router and dispatcher
- Docker causes tcpreplay to tun on its own core, so more CPU cycles are available to the border routers



- Four times increase in performance over the reference BR
- XDP BR throughput is fluctuating between 0.8 Mpps and 1.2 Mpps
- XDP BR cannot use more than one CPU core, because veths do not support multiple RX queues
- Evaluation on real hardware is needed

Test	Throughput	Time per Packet
Direct veth0 <-> veth1	1.238 Mpps	0.81 µs
Native bridge	0.685 Mpps	1.46 µs
XDP BR with AES	0.676 Mpps	1.48 µs
XDP BR without AES	0.895 Mpps	1.12 µs
Reference BR (Docker)	0.256 Mpps	3.91 µs
XDP BR (Docker)	1 Mpps	1 µs

Conclusion and Future Work

- XDP BR works as a proof-of-concept
- Four times faster than reference BR even in a limited virtualized environment
- Work is still ongoing
  - IPv6 underlay is not fully implemented
  - Add a new BPF helper function to the kernel and expose hardware accelerated AES to BPF
  - Support for EPIC and COLIBRI
- Open questions
  - How does the XDP BR perform on real hardware?
  - How does it compare to the commercial DPDK-based SCION border router?
  - How well does it scale with more CPU cores?
- Goal of future work
  - XDP as standard feature in the reference BR
  - Should be turned on automatically if compatible hardware is detected