PoC of New IP

Bhaskar Kataria
Rohit M. P.
Leslie Monis
Mohit P. Tahiliani
NITK Surathkal

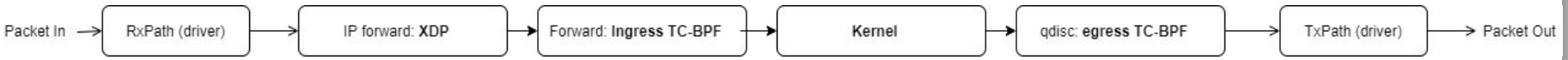
Kiran Makhijani
Futurewei Technologies, Inc.

Goals

Implement a programmable data plane for New IP packet processing

- **Address customization**: applications and routers should be able to forward packets between hosts with different address formats.
- **Design an end-to-end model to meet service delivery guarantees**: routers should be able to process various in-network New IP contracts as described by the applications.
- **Rapid experimentation** of the New IP components should be possible.

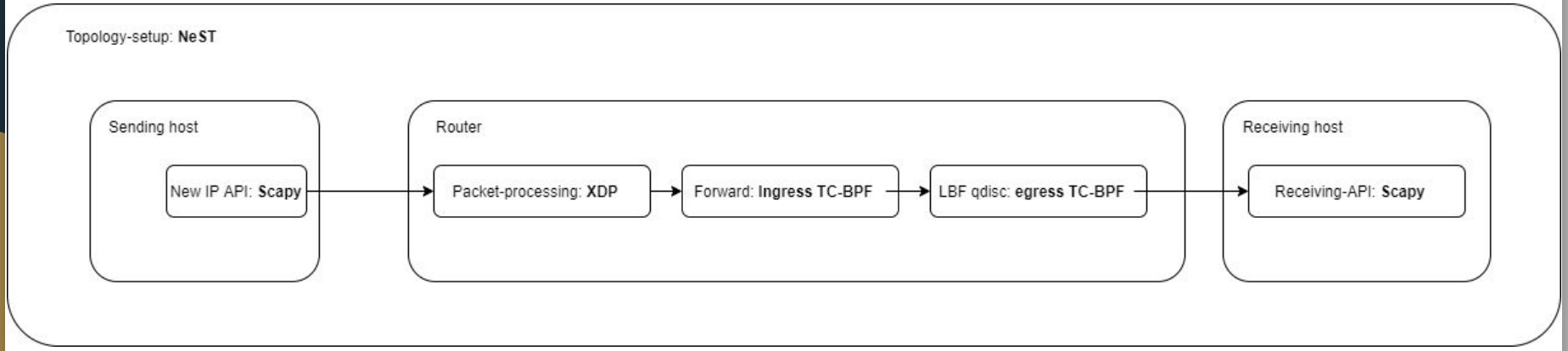
IP packet flow in Linux



Main Components

1. Topology setup - Network Stack Tester (NeST)
2. Scapy - Host side API
3. eXpress Data Path XDP - shipping and contract processing
4. Traffic Control-BPF (TC) - LBF queue discipline

New IP packet flow



Topology setup: NeST

- NeST is a python3 package designed for emulating real-world networks
- Uses Network namespaces to create the topology
- Makes testing easier

For this project, NeST is used to:

- Assign IP addresses to the nodes
- Populate the routing tables
- Form the packet by providing necessary information like MAC addresses

Host side API: Scapy

- Python library for powerful interactive packet manipulation
- Helps forming custom packet formats
- Uses Raw sockets internally

For this project, Scapy is used to:

- Craft and send the packets from the virtual interface setup by NeST
- Sniff and decode the packet at the receiver

XDP

- XDP is an eBPF based high performance data path
- Merged in the Linux kernel
- Provides packet processing at the lowest point in the software stack

For this project, XDP is used to:

- Update the Ethernet header in the packet
- Identify the interface from which the packet is to be sent using kernel routing tables or BPF maps
- Store the exit interface index (ifindex) in metadata of the packet
- Pass the packet on to the TC BPF hook

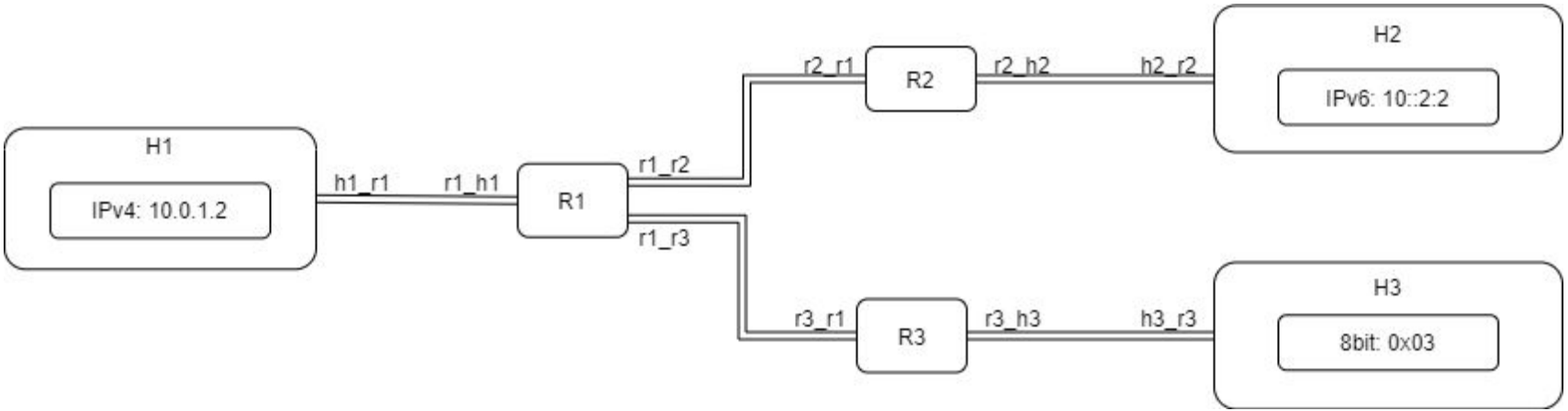
TC-BPF

- BPF programs attached to the traffic control (tc) ingress and egress hook
- Runs after the networking stack has done the initial processing of the packet
- Has access to the metadata of the packet

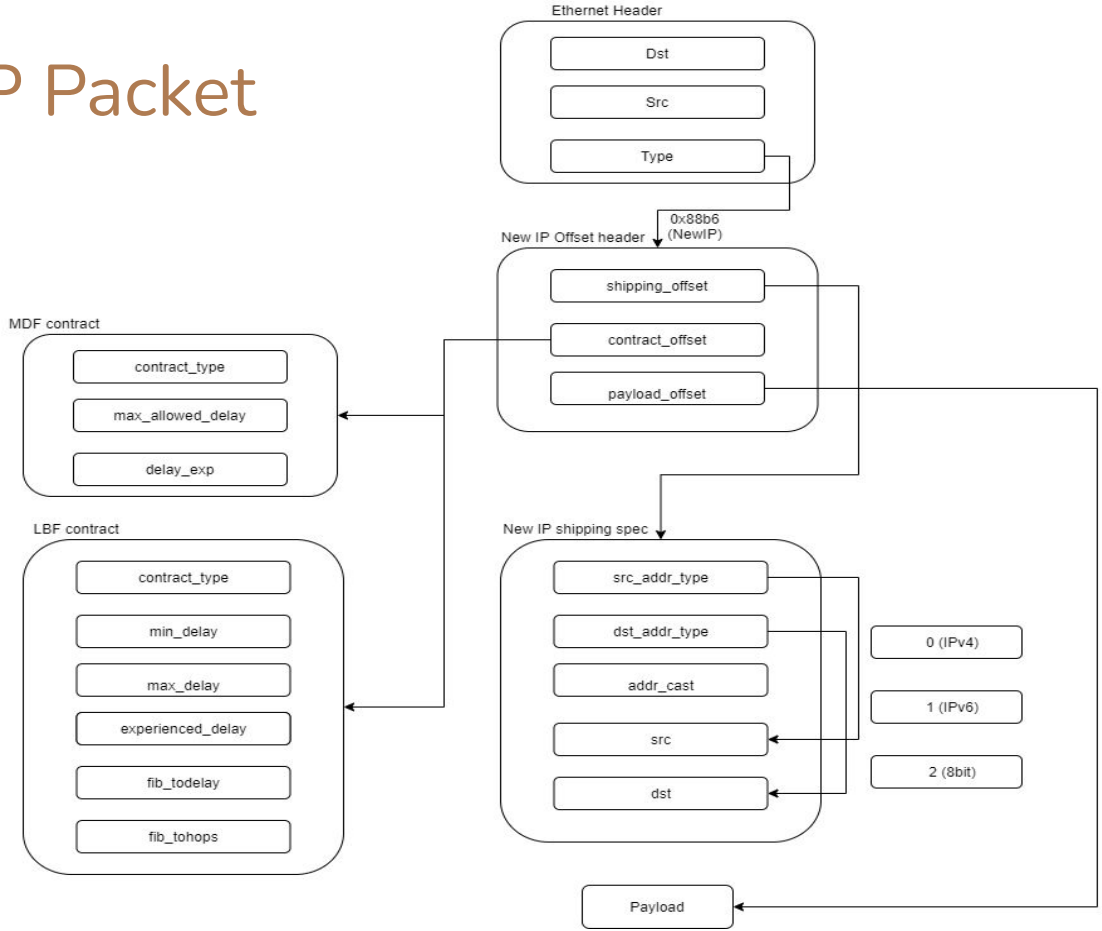
For this project, TC-BPF is used to:

- Add our queueing discipline (we cannot have queue discipline with just XDP)
- Read the ifindex from metadata at ingress hook
- Redirect the packet to the egress hook of the interface associated with ifindex
- Run our queue discipline on the egress hook

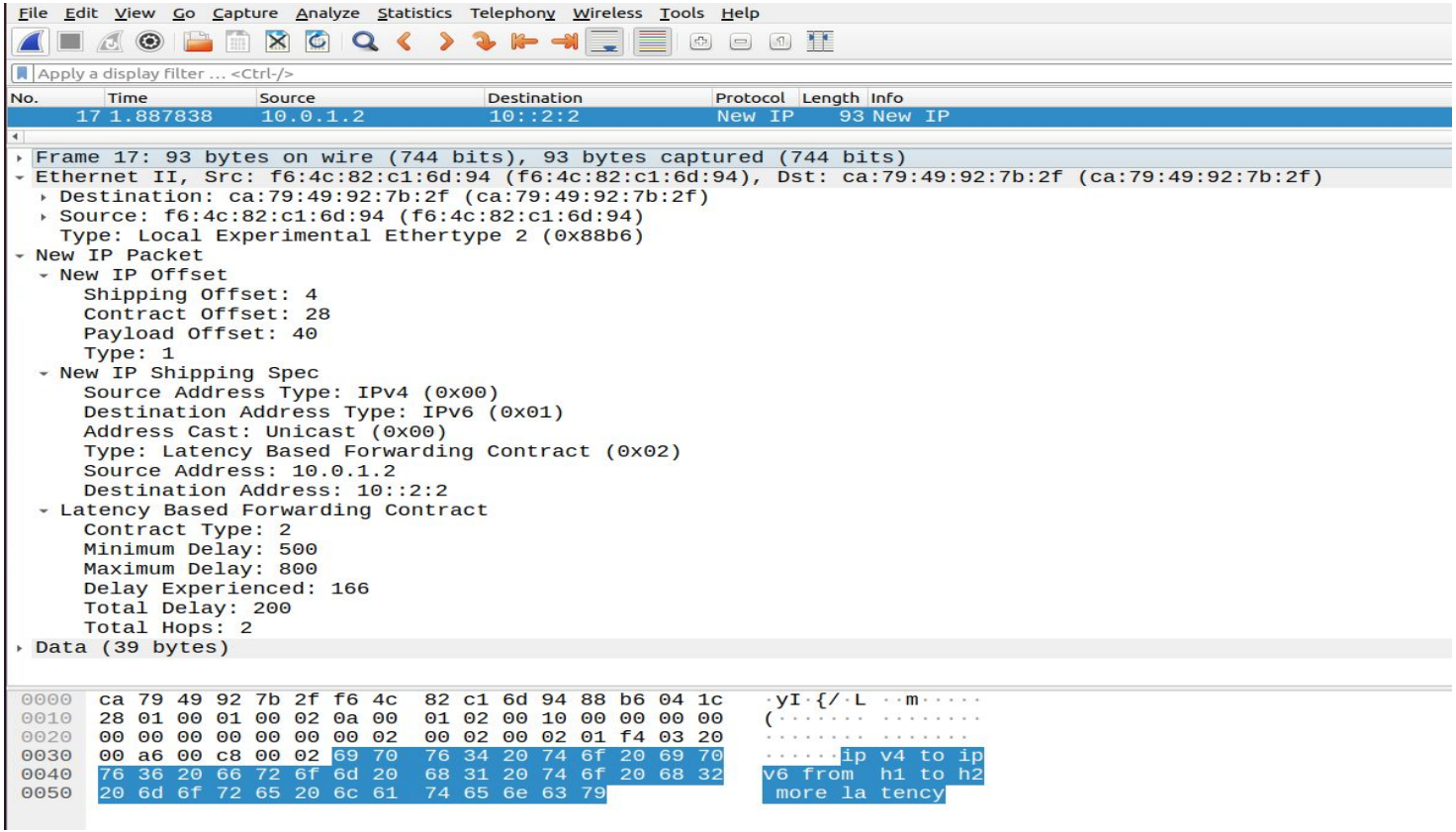
Topology Setup - NeST



New IP Packet



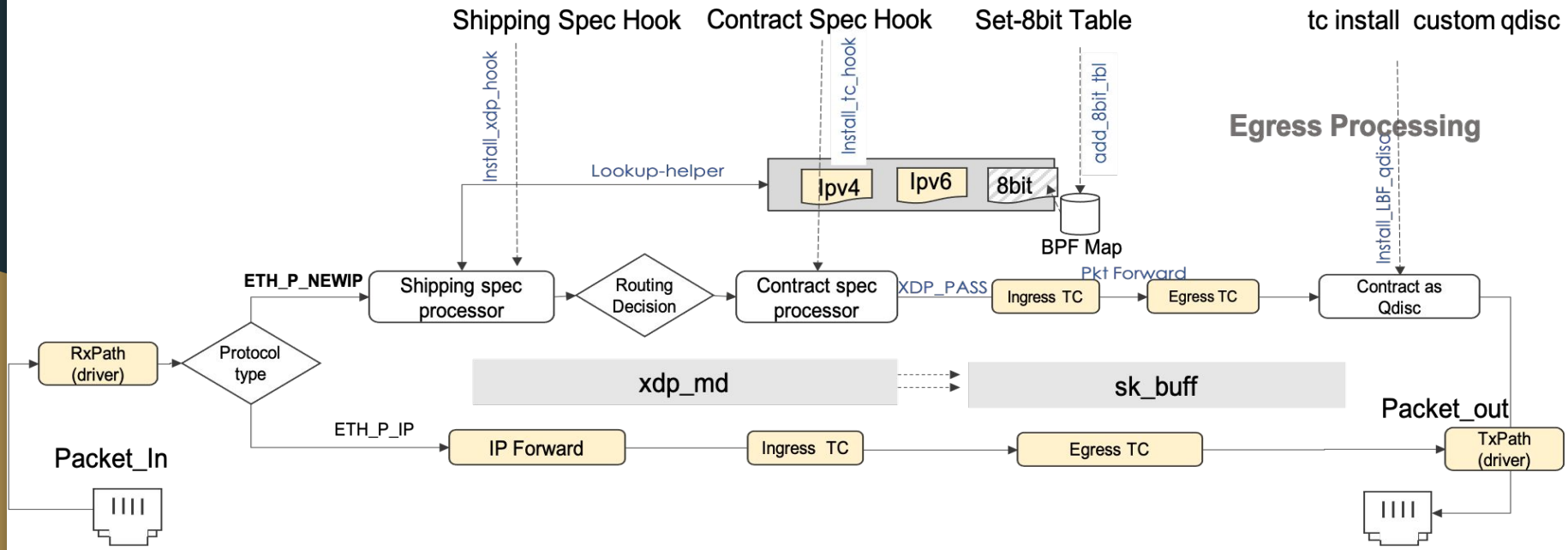
New IP Packet in Wireshark



API for Crafting a New IP packet

```
sender_obj = Sender()

sender_obj.make_packet(src_addr_type="ipv4", src_addr="10.0.1.2",
dst_addr_type="ipv6", dst_addr="10::2:2",content="ipv4 to ipv6 from h1
to h2 more latency")

lbf_contract = LatencyBasedForwarding(min_delay = 500, max_delay = 800,
fib_todelay = 0, fib_tohops = 3)

sender_obj.set_contract ([lbf_contract])

sender_obj.send_packet(iface='h1_r1', show_pkt=True)
```

New IP Packet processing



Demo

Thank you!