Skydive

Packets Probes and eBPF Filtering in Skydive

Nicolas Planel

redhat

What is Skydive ?

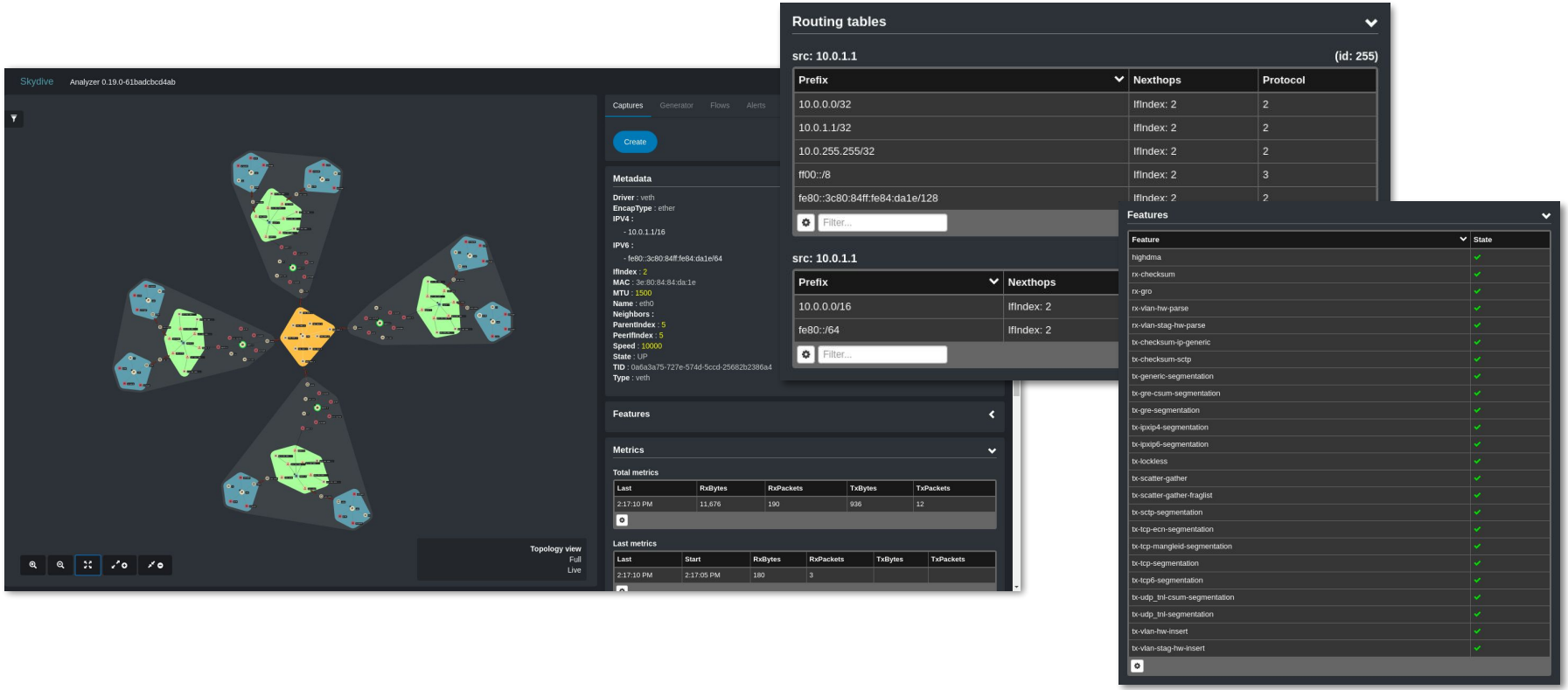
Real-time network topology and protocol analyzer

Why: Network is complex (SDN)

- Topologies (physical, overlay, and application) are dynamic
- Multiple opaque tunneling layers
- Infrastructure complexity

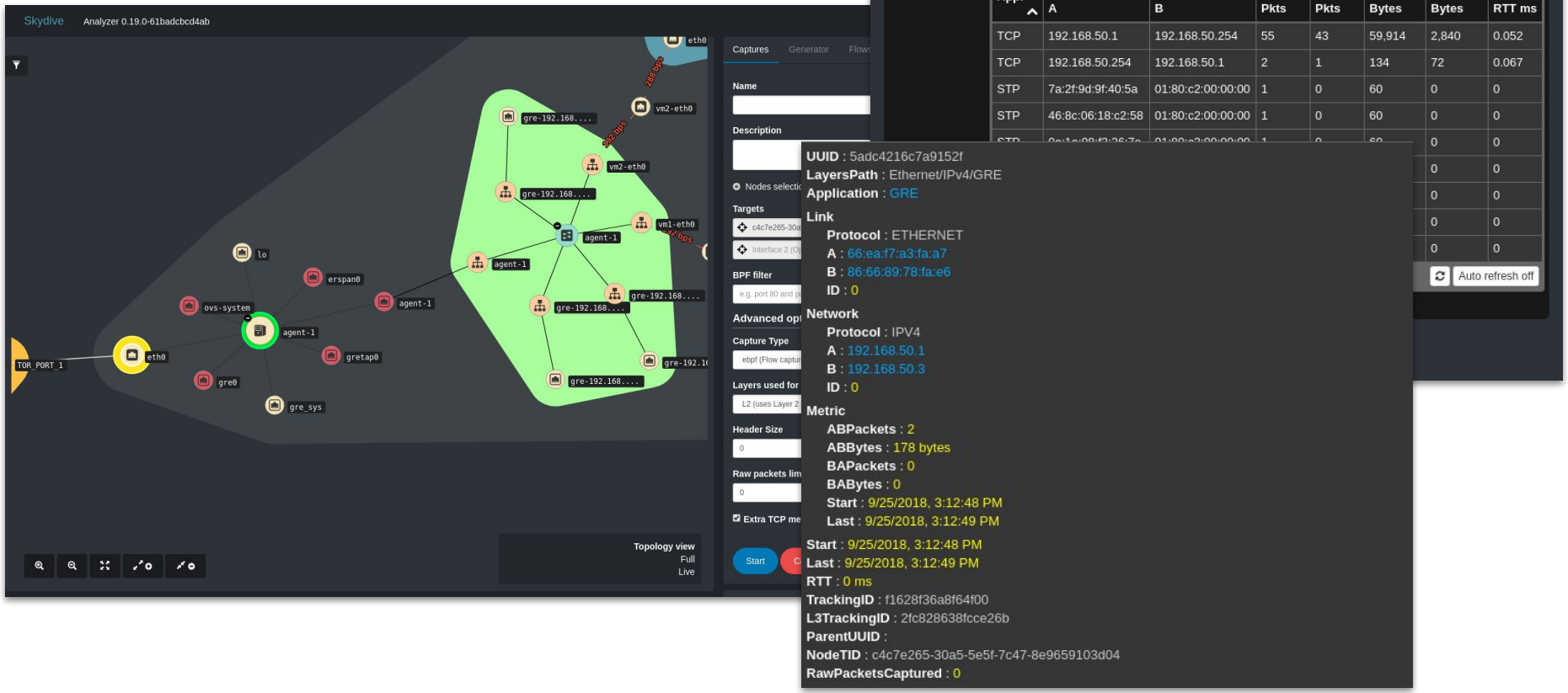
Use cases

Network Topology Visualization (Discovery)



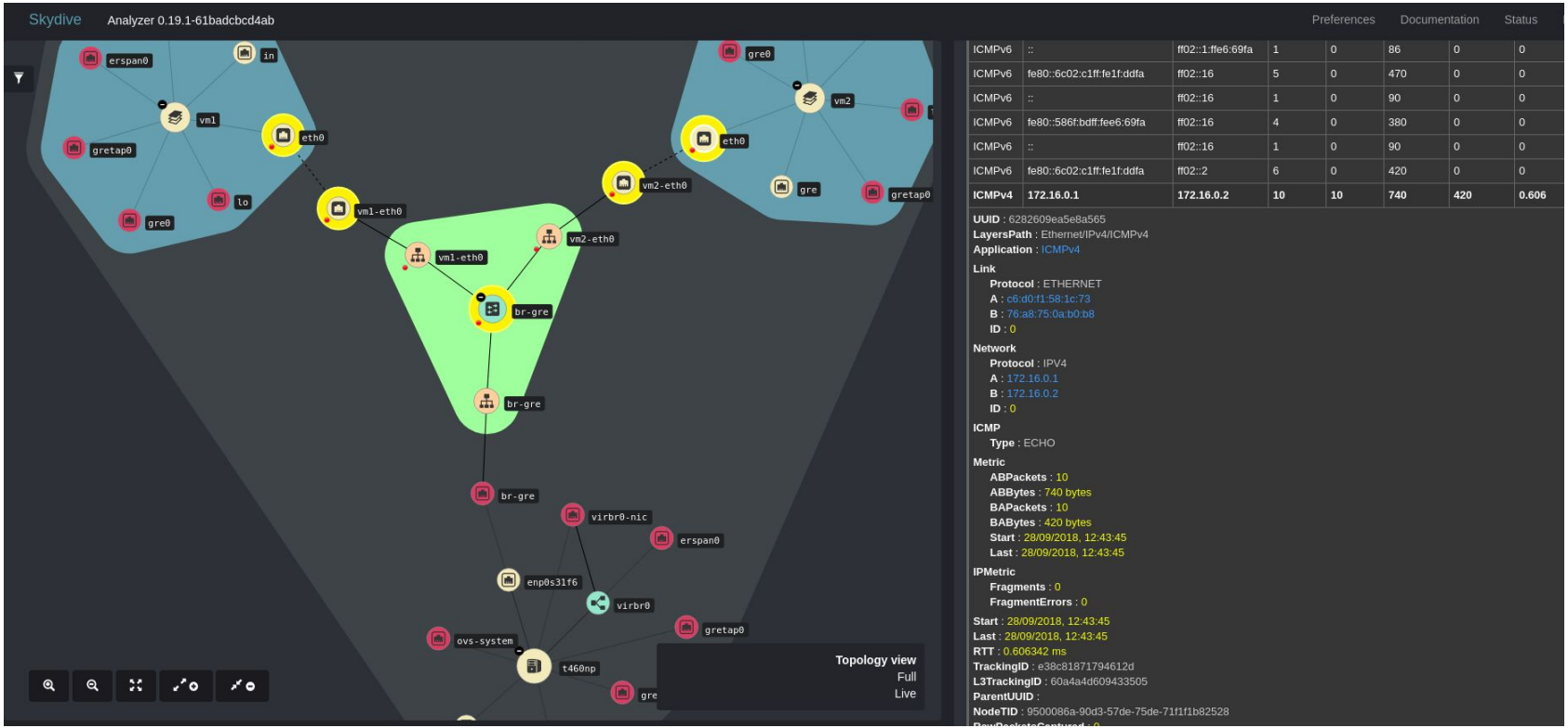
Use cases

Flow Troubleshooting (Capture)



Use cases

Flow Troubleshooting (Tracking)

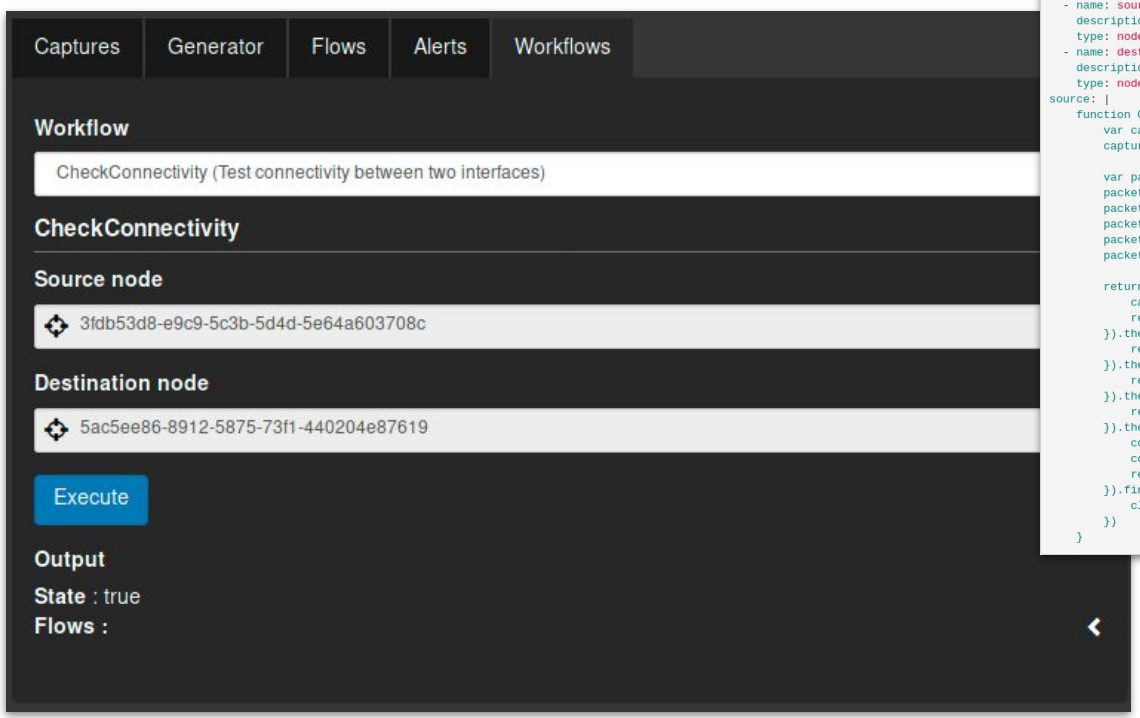


Use cases

Flow Monitoring (Grafana)

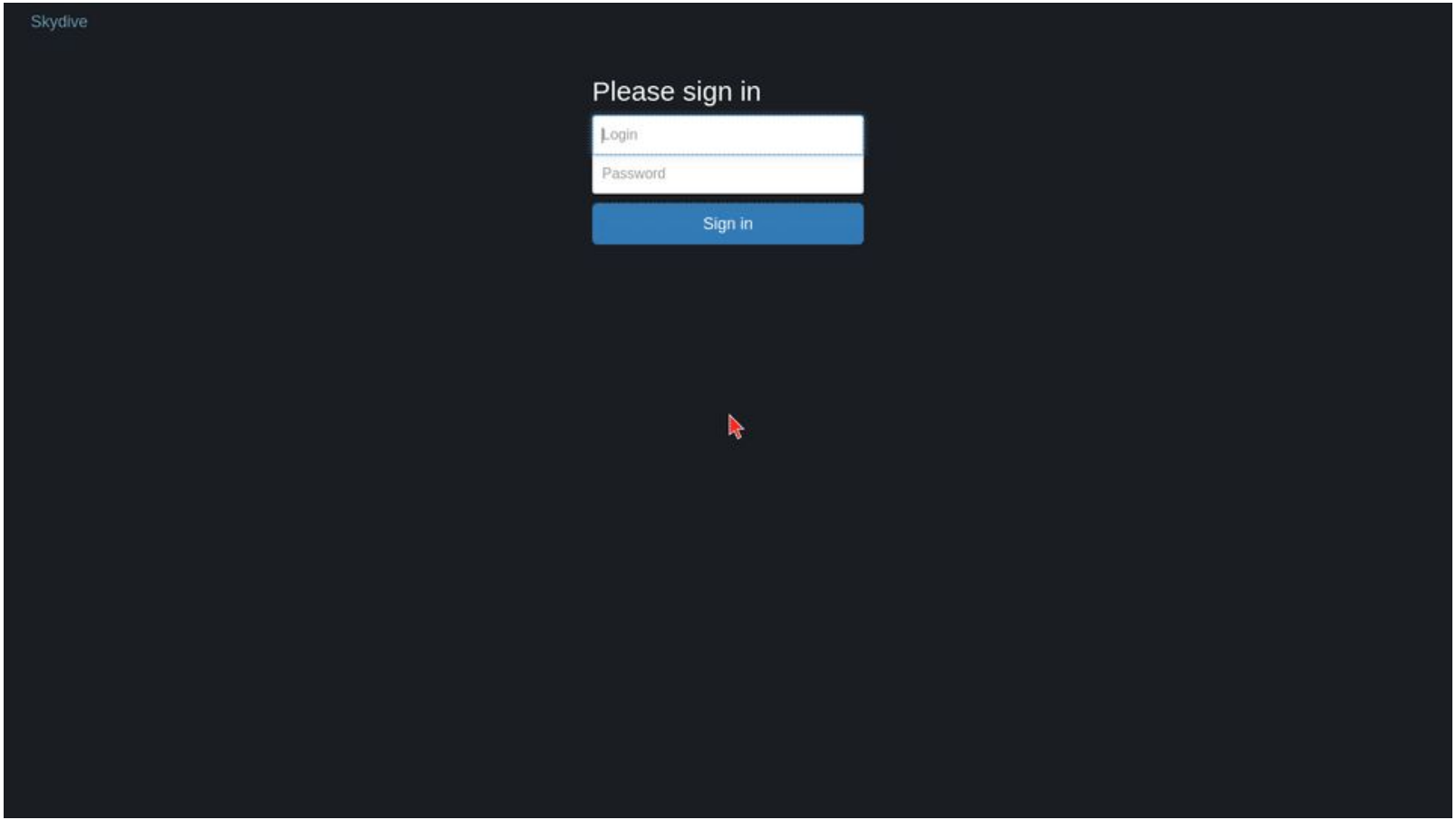
Use cases

Workflow (Validation, CI/CD, ...)

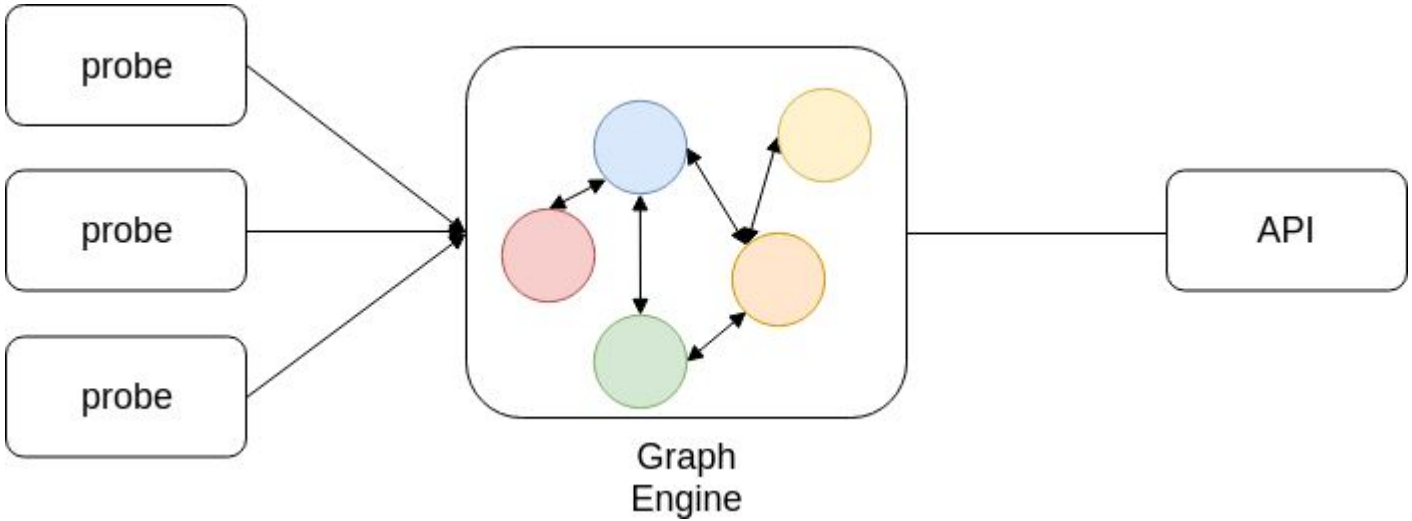



```
---
name: "CheckConnectivity"
description: "Test connectivity between two interfaces"
parameters:
  - name: source
    description: Source node
    type: node
  - name: destination
    description: Destination node
    type: node
source: |
    function CheckConnectivity(from, to) {
        var capture = new Capture();
        capture.GremlinQuery = "G.V().Has('TID', '" + from + "').ShortestPathTo(Metadata('TID', '" + to + "'))";

        var packetInjection = new PacketInjection();
        packetInjection.Src = "G.V().Has('TID', '" + from + "')"
        packetInjection.Dst = "G.V().Has('TID', '" + to + "')"
        packetInjection.Type = "icmp4"
        packetInjection.ICMPID = 123
        packetInjection.Count = 5

        return client.captures.create(capture).then(function (c) {
            capture = c
            return sleep(2000)
        }).then(function () {
            return client.packetInjections.create(packetInjection)
        }).then(function () {
            return sleep(2000)
        }).then(function () {
            return client.G.Flows().Has("ICMP.ID", 123)
        }).then(function (flows) {
            console.log("Flows requested ! :-)")
            console.log(flows)
            return { "State": true, "Flows": flows }
        }).finally(function () {
            client.captures.delete(capture.UUID)
        })
    }
```

Skydive in action



Skydive Architecture

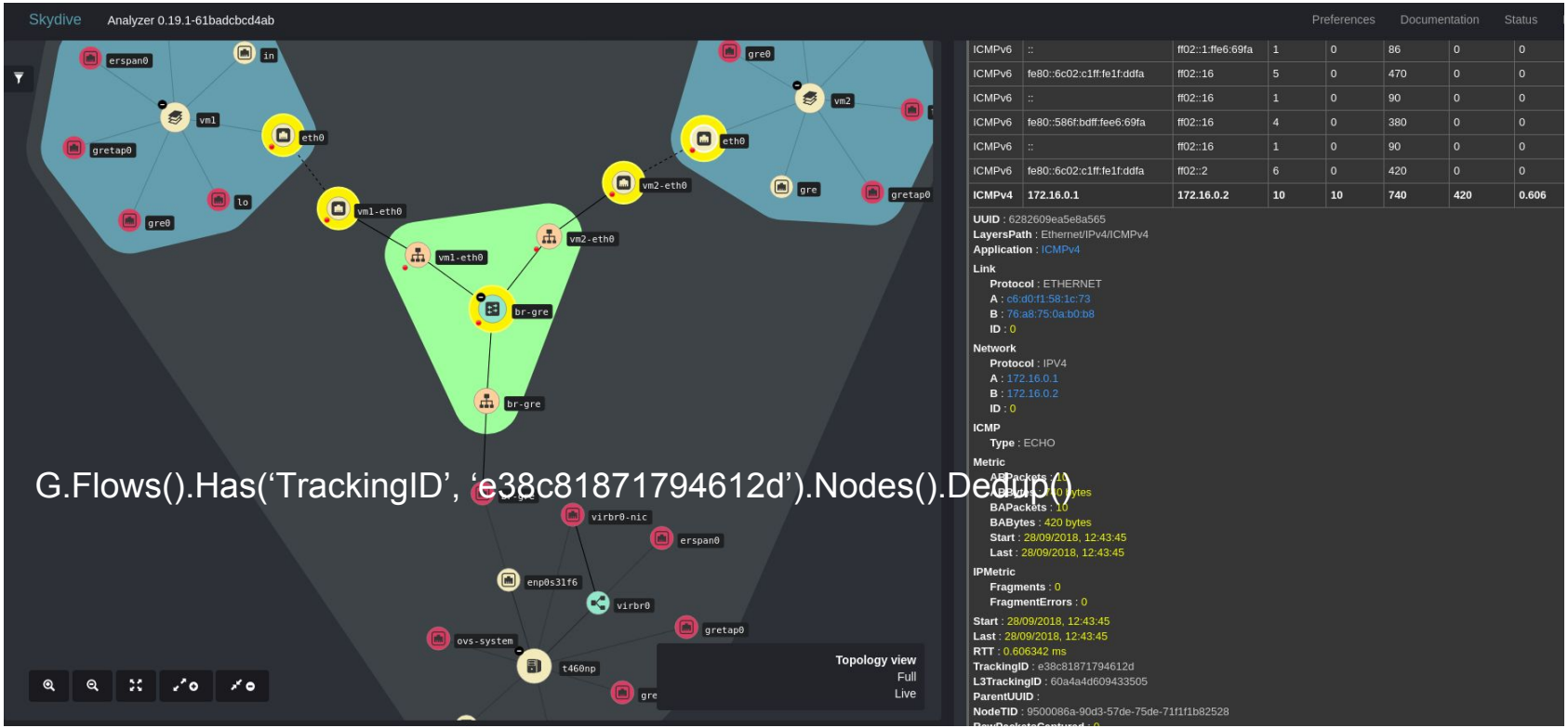


Skydive Gremlin API

```
nplanel@t460np:~/devel/go/src/github.com/skydive-project/skydive$ skydive client query 'g.V().Has("Name", "eth0")'
[
        {
                "ID": "c050a852-fbab-47b0-6087-62217ef295d8",
                "Metadata": {
                        "Capture": {
                                "ID": "fc25caac-e34f-4fa8-5d1c-a5cb4fe7c7ae",
                                "PacketsDropped": 0,
                                "PacketsReceived": 64,
                                "State": "active"
                        },
                        "Driver": "veth",
                        "EncapType": "ether",
                        "IPV4": [
                                "172.16.0.1/24"
                        ],
                        "IPV6": [
                                "fe80::c4d0:f1ff:fe58:1c73/64"
```

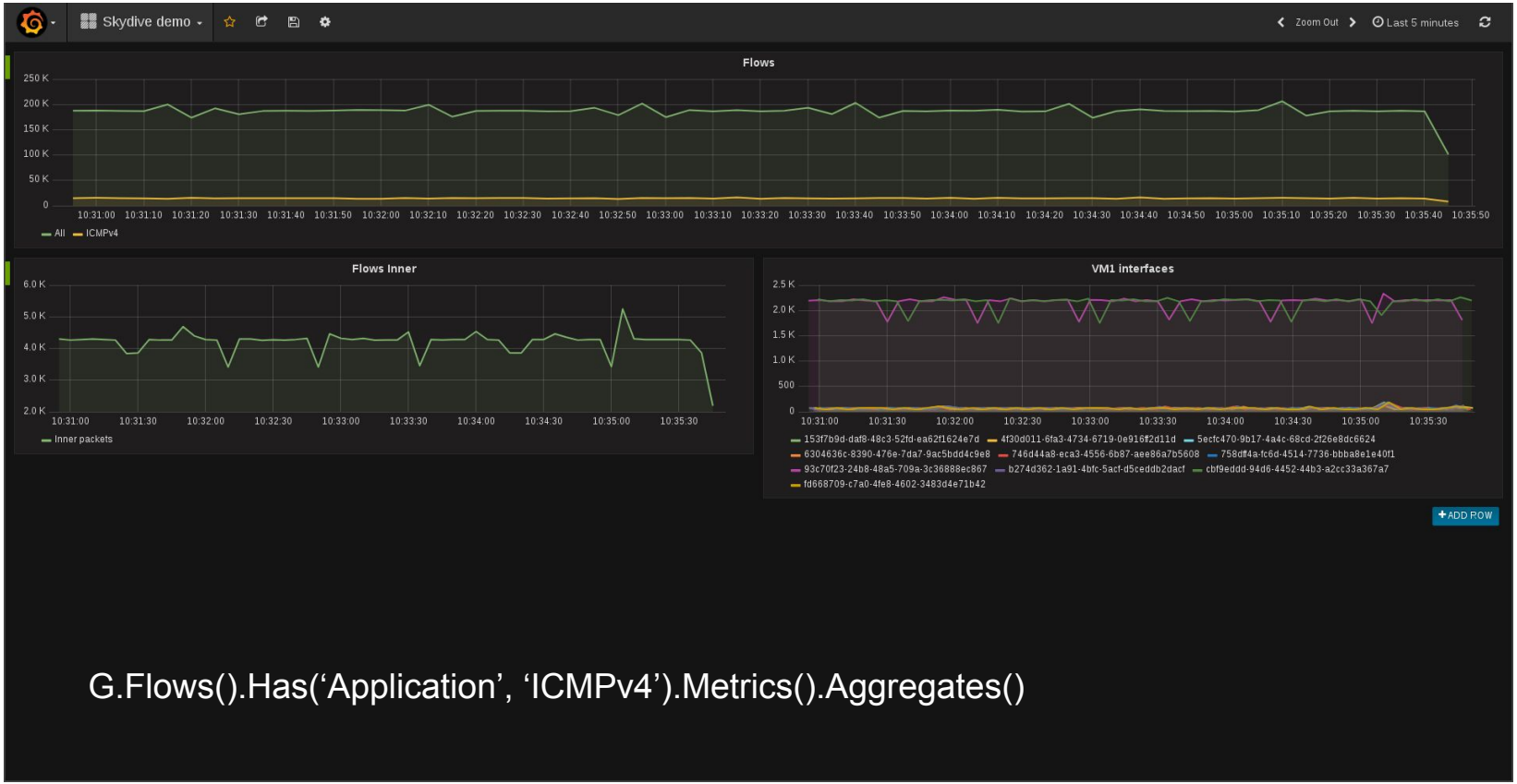
Use cases

Flow Troubleshooting (Tracking)



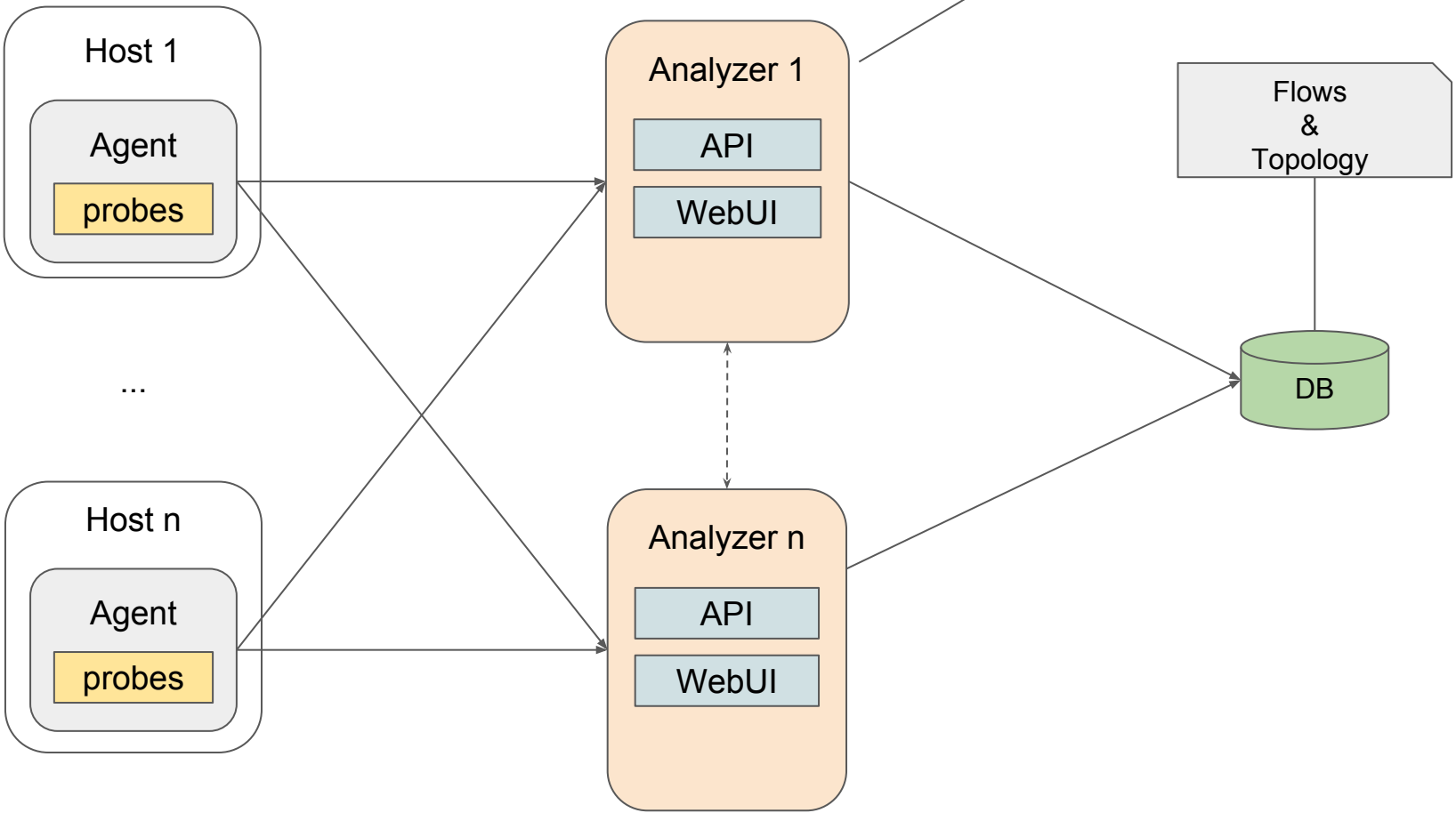
Use cases

Flow Monitoring (Grafana)



G.Flows().Has(‘Application’, ‘ICMPv4’).Metrics().Aggregates()

Skydive Architecture



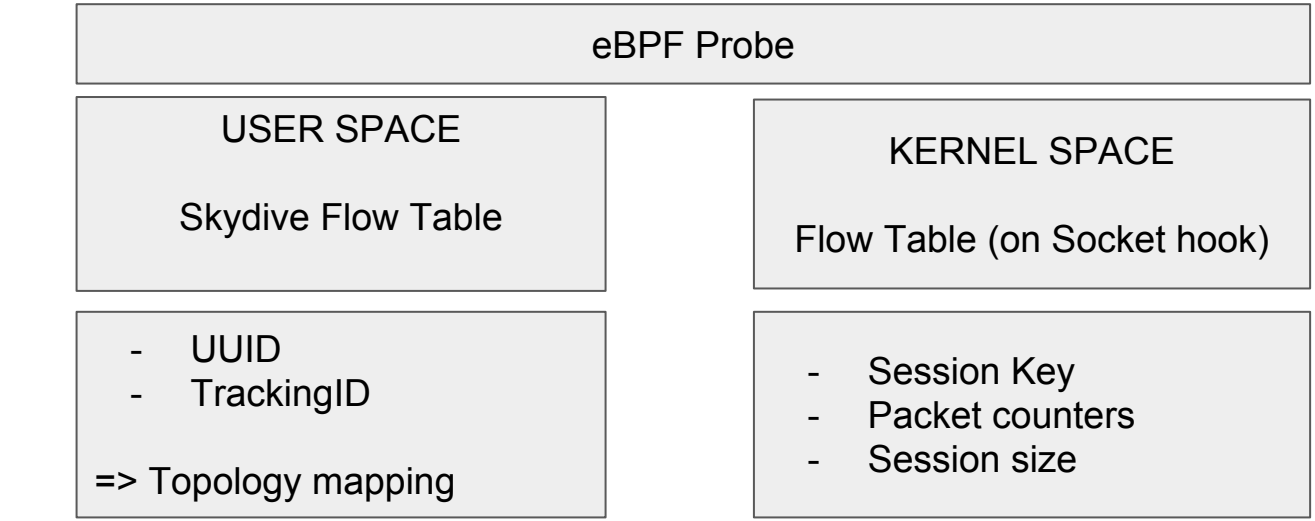
Topology Probes

- Netlink
- NetNS
- OVS
- Docker
- Kubernetes
- Neutron
- SocketInfo

Skydive capture probes

	Kernel	Overhead	Limitations
AF Packet	all	Huge	None
BPF (Syn/Fin/Rst/Ack/!Psh)	2.5+	Low	No packets metrics TCP only Start and end session
eBPF	3.18+	Low	No classification No fragmentation No tunneling (implementation)

Skydive eBPF capture



Skydive SocketInfo

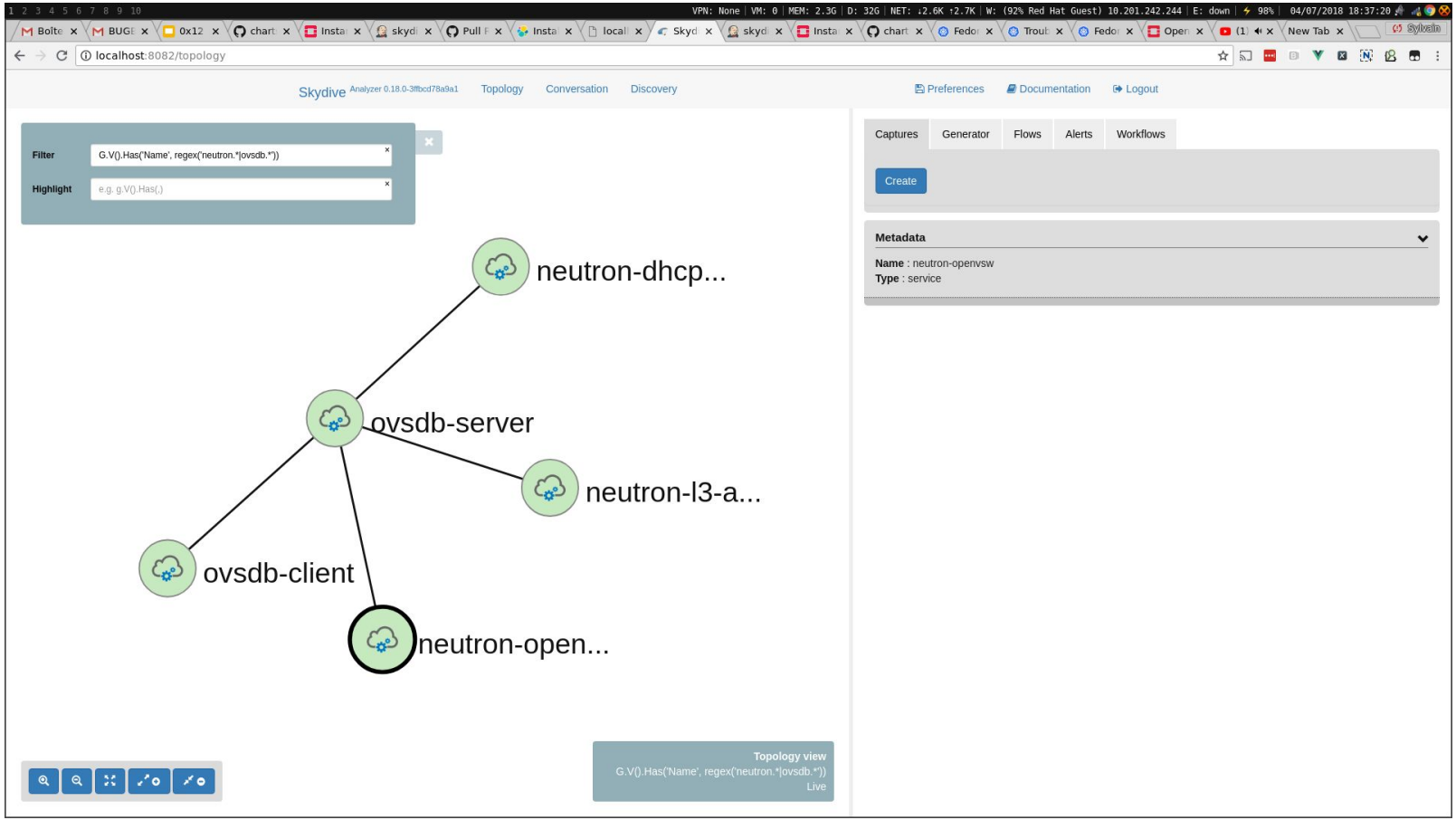
The equivalent of “ss -tunp” saved in Skydive

```
udp      ESTAB          59136      0   192.168.78.45:43243    216.58.201.110:443
users:(("chrome",pid=1992,fd=465))
tcp      CLOSE-WAIT     32         0   192.168.78.45:43912    23.38.33.253:443
users:(("chrome",pid=1992,fd=515))
tcp      ESTAB          0          0   192.168.78.45:42058    192.168.78.38:22
users:(("ssh",pid=27194,fd=3))
...
```

SocketInfo relies on Kprobes and eBPF

```
TCP,vm-2.rdocloud,10.0.0.16,6633,/usr/bin/python2.7,neutron-openvsw,
  => vm-1.rdocloud,10.0.0.15,/usr/sbin/ovs-vswitchd,ovs-vswitchd

TCP,vm-1.rdocloud,10.0.0.15,3306,/usr/libexec/mysqld,mysqld,
  => vm-1.rdocloud,10.0.0.15,/usr/bin/python2.7,nova-consoleaut
...
```

Skydive Service/Application view



Skydive roadmap

Hybrid Capture

- Capture first packets until Classified (dissectors ?)
- Continue to capture flow counters via eBPF

eBPF/Kprobe on local process

- Retrieve fragmentation / retransmission counters
- NetNS Create

Extract graph engine (Steffi)

- Could be integrated in your tool

Questions ?

https://skydive.network
https://github.com/skydive-project/skydive