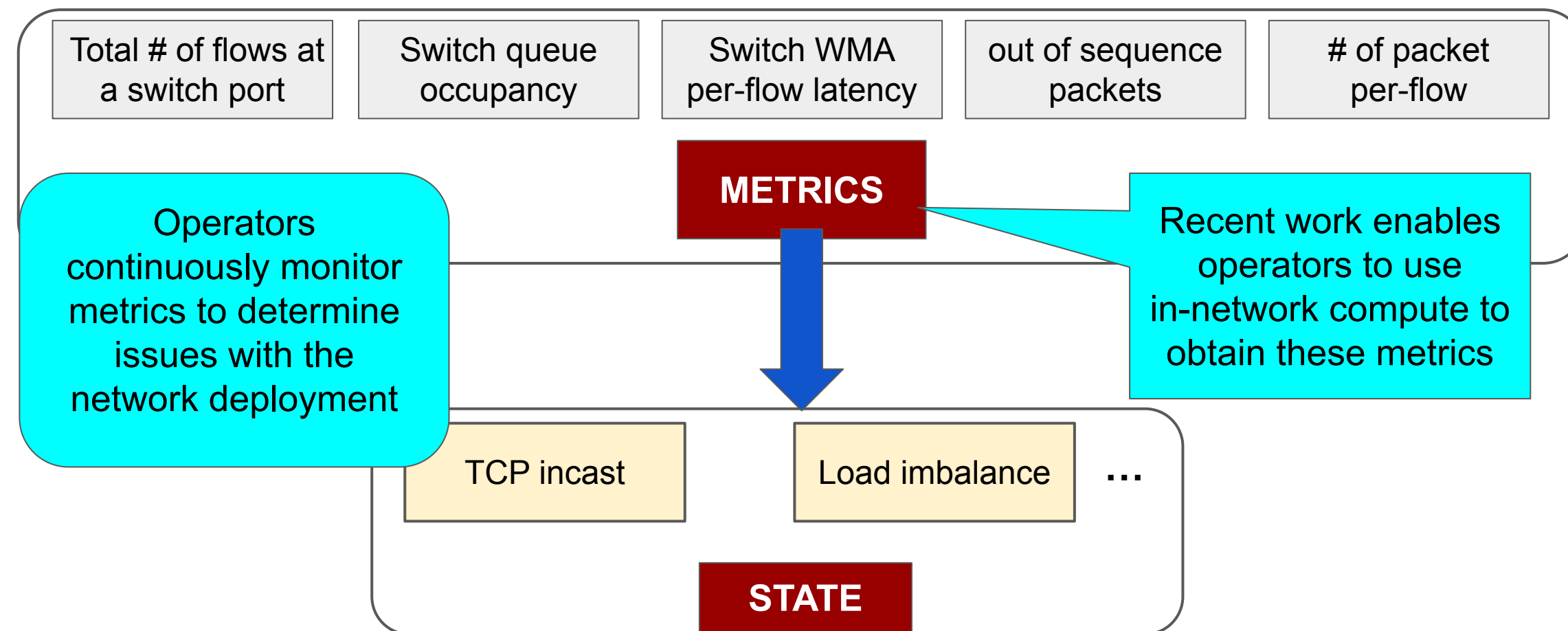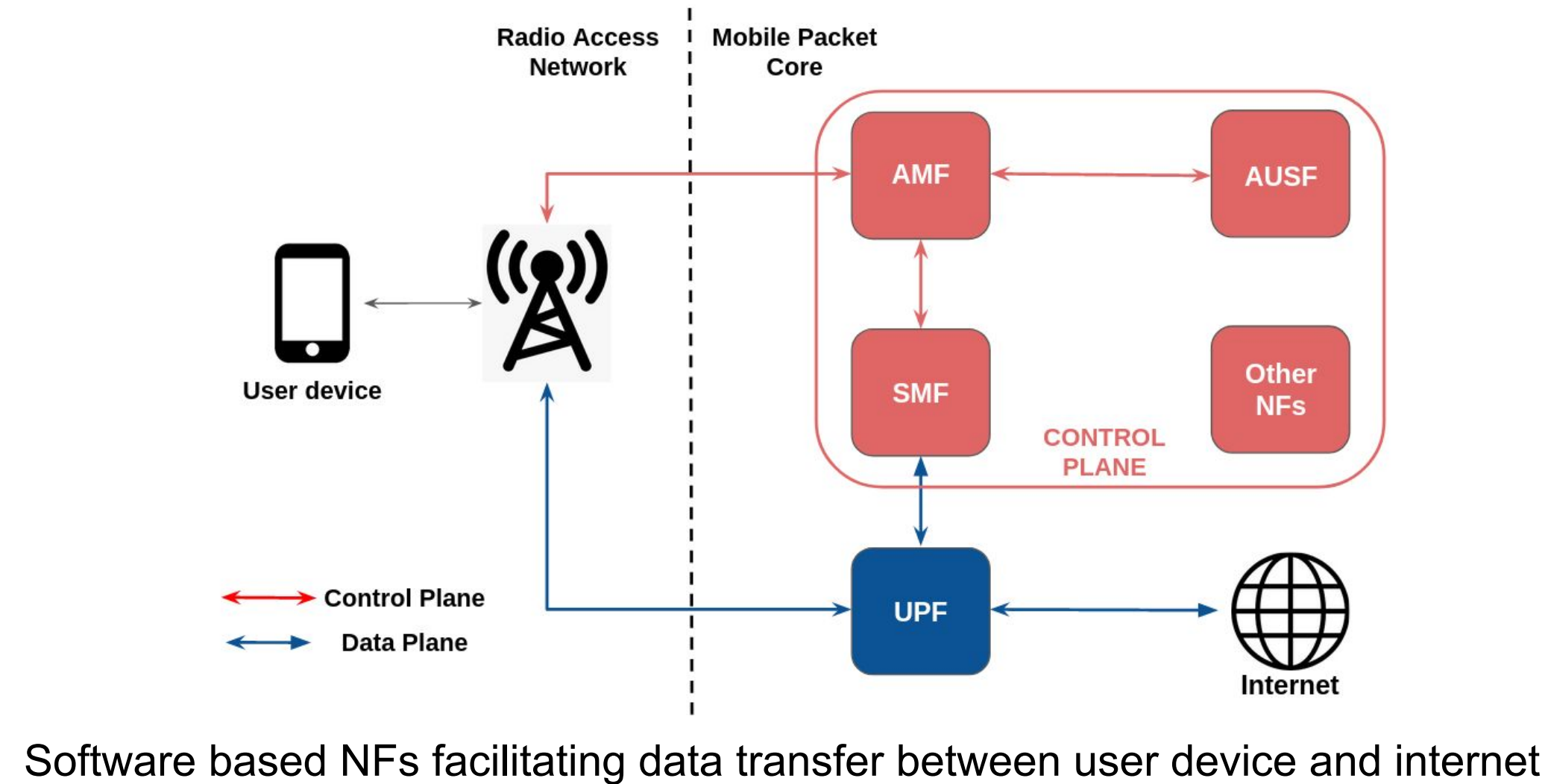# DeepSight: In-network application layer telemetry in eBPF and P4

K. Ashwin Kumar, Abhik Bose, Mythili Vutukuru
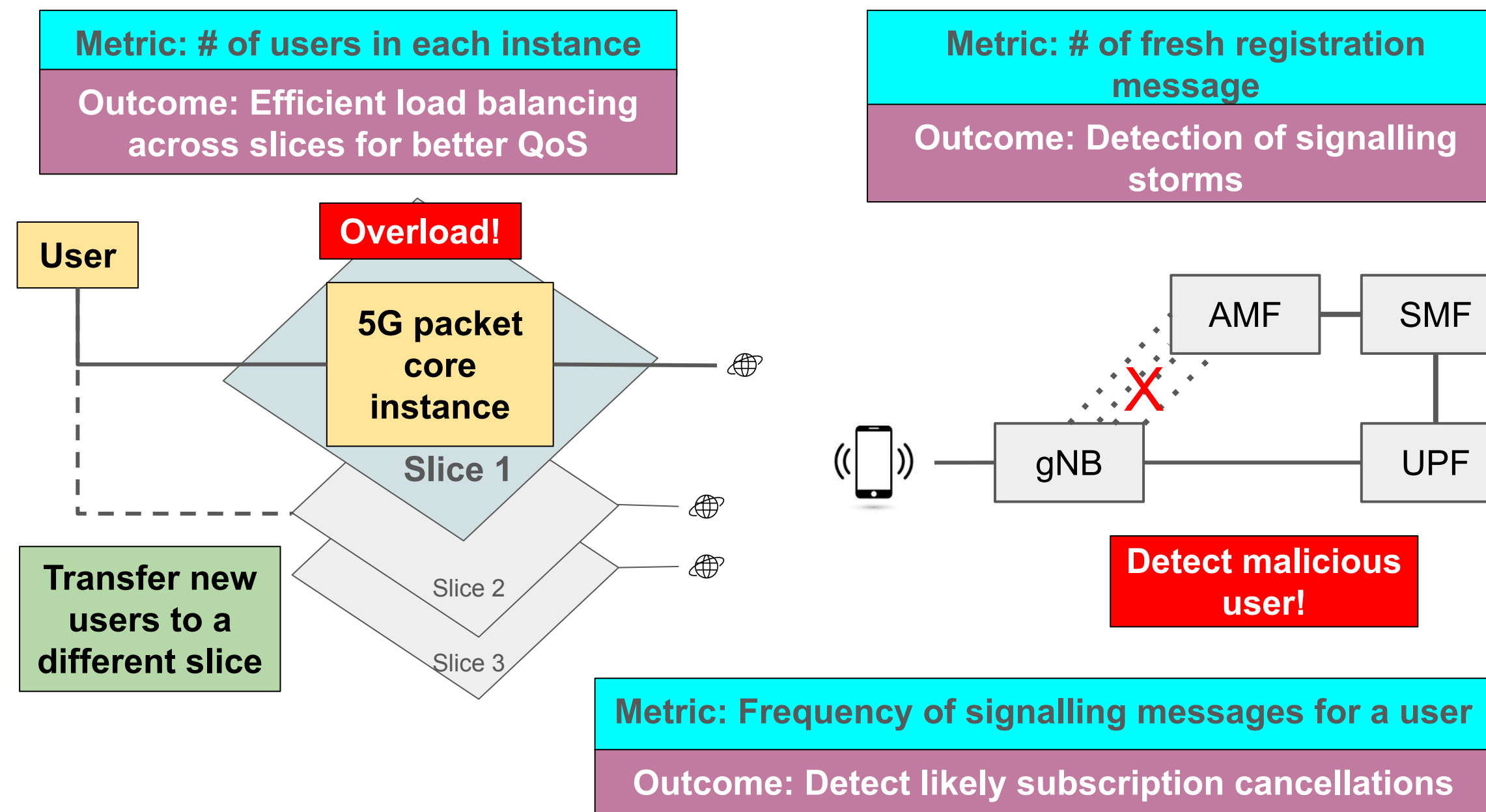
Network Telemetry

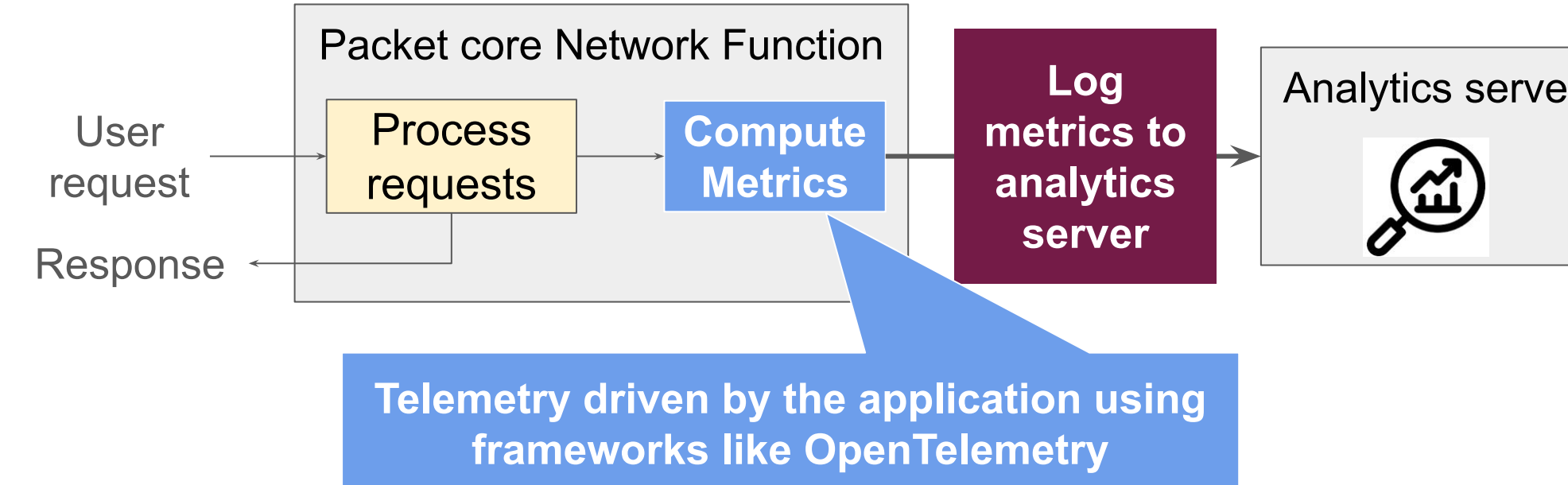| Total # of flows at a switch port	Switch queue occupancy	Switch WMA per-flow latency	out of sequence packets	# of packet per-flow

METRICS

Operators continuously monitor metrics to determine issues with the network deployment

Recent work enables operators to use in-network compute to obtain these metrics

TCP incast   Load imbalance   ...

STATE

Application Layer Telemetry: 5G packet core

Radio Access Network | Mobile Packet Core

AMF | AUSF
SMF | Other NFs
CONTROL PLANE

User device

UPF

Internet

→ Control Plane
→ Data Plane

Software based NFs facilitating data transfer between user device and internet

Operators in the 5G packet core are interested in application layer telemetry

Metric: # of users in each instance
Outcome: Efficient load balancing across slices for better QoS

Metric: # of fresh registration message
Outcome: Detection of signalling storms

User

Overload!
5G packet core instance
Slice 1
Slice 2
Slice 3

Transfer new users to a different slice

AMF | SMF
gNB
UPF

Detect malicious user!

Metric: Frequency of signalling messages for a user
Outcome: Detect likely subscription cancellations

State-of-the-art

Packet core Network Function

User request → Process requests → Compute Metrics → Log metrics to analytics server → Analytics server

Response

Telemetry driven by the application using frameworks like OpenTelemetry

Curious case of the packet core

Operators receive black box NF from the vendor → Log metrics to an analytics server → Analytics server

Operators require continuous vendor support!!
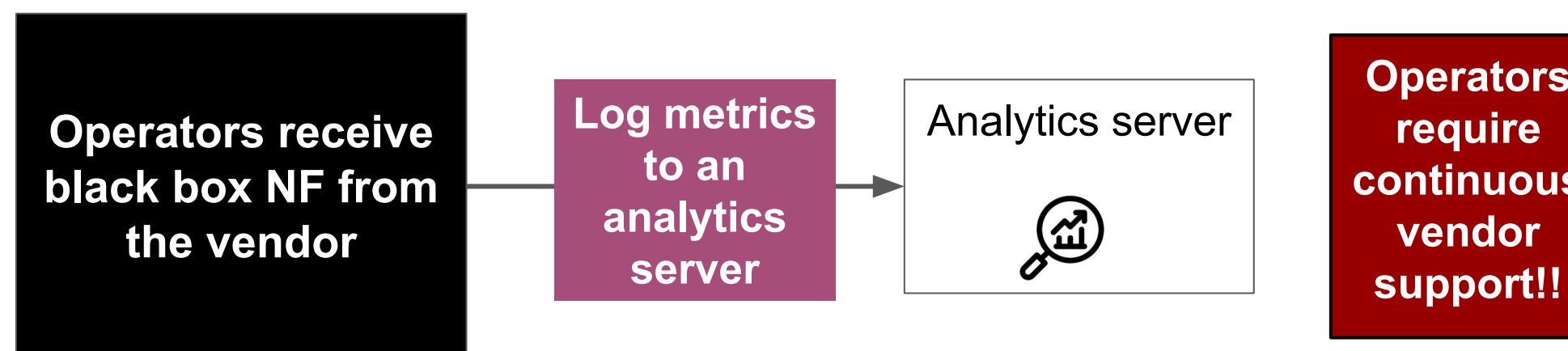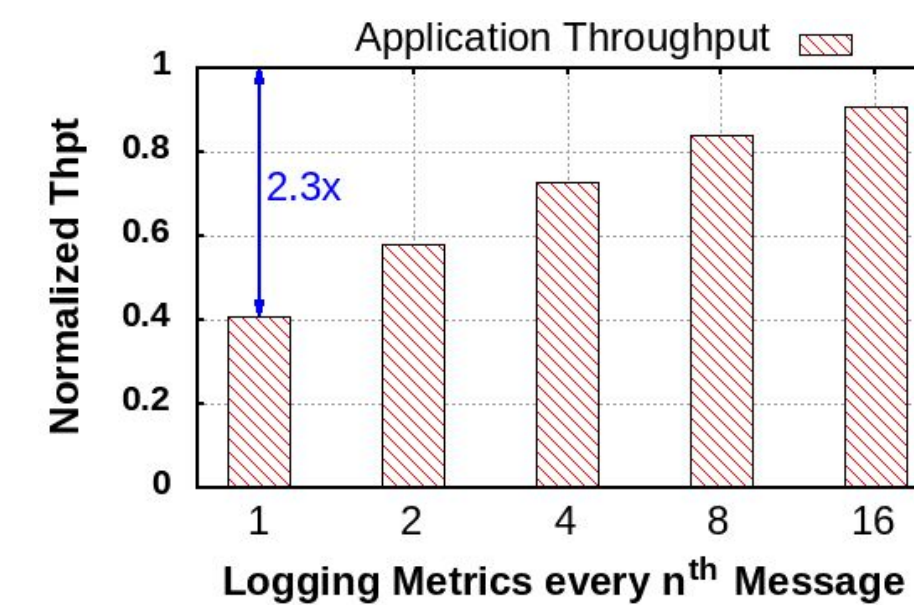
Can the operator break free from vendor support?

Source NF logging on every incoming message which would expose a superset of metrics to the operator
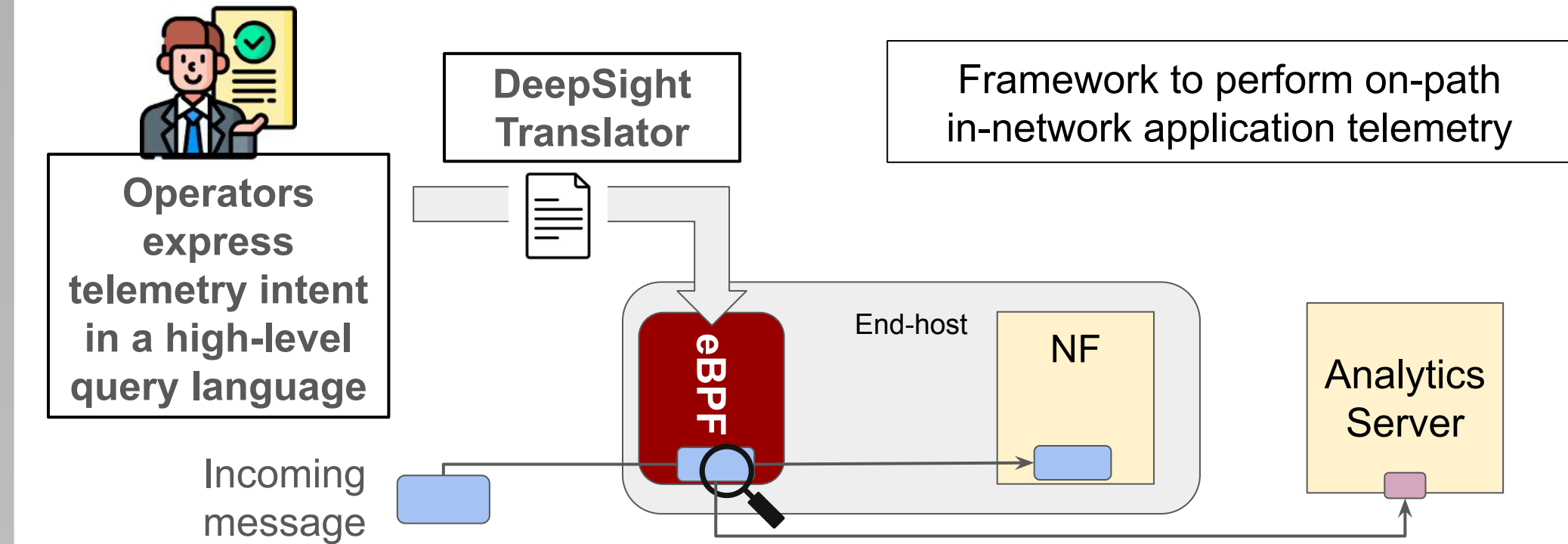

Application Throughput
2.3x

Normalized Thpt

Logging Metrics every nth Message

Logging on every message hurts NF performance

Stresses the analytics server unnecessarily

Alternatives

Incoming packet → eBPF → NF

AF_PACKET → Monitoring Application

End-Host

On-path vs off-path

Throughput(Gbps)

Parsing Size(Bytes)

eBPF
AF_PACKET

# of CPUs	eBPF (Gbps)	AF_PACKET (Gbps)
2	6.5	5
4	13	9
8	25	16
12	34	25

Off-path riddled with packet copies!

DeepSight

Operators express telemetry intent in a high-level query language

DeepSight Translator

Framework to perform on-path in-network application telemetry

eBPF | End-host | NF | Analytics Server

Incoming message

Example Queries

```
s1 = filter(msgstream, type, an_release)
count_per_user = groupby(s1, user_id, count())
emit(s1, count_per_user > 10, [user_id])
```
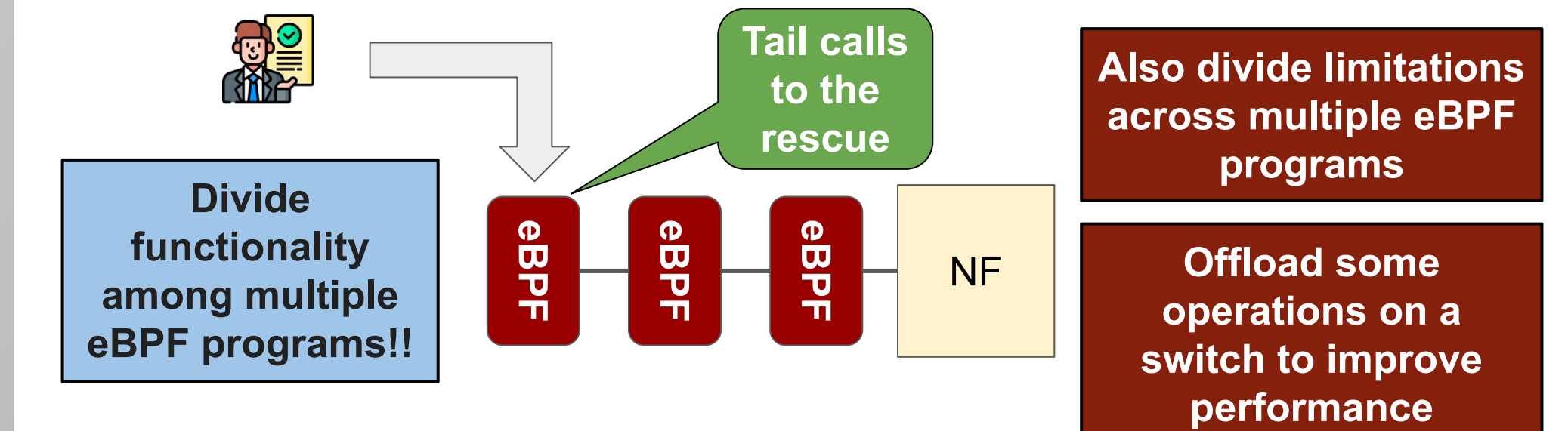
Find users going idle frequently

```
s1 = filter(msgstream, type, an_release)
s2 = filter(msgstream, type, session_estb)
session_per_user = groupby(s2, user_id, count())
s3 = join(s1, s2, [session_per_user])
emit(s3, session_per_user > 10, [user_id])
```

Find users with multiple sessions going idle

Challenges and key ideas

Application telemetry requires parsing the application payload but eBPF verifier places various restrictions on the program

Divide functionality among multiple eBPF programs!!

Tail calls to the rescue

eBPF | eBPF | eBPF | NF

Also divide limitations across multiple eBPF programs

Offload some operations on a switch to improve performance

Results

72% improvement in performance → Log metrics to an analytics server → ~ 16x reduction in load

Query: Emit user id of a user with more than N session modification messages

DeepSight can filter unnecessary messages and aggregate metrics over N messages