Leveraging open technologies to monitor packet drops in AI cluster fabrics

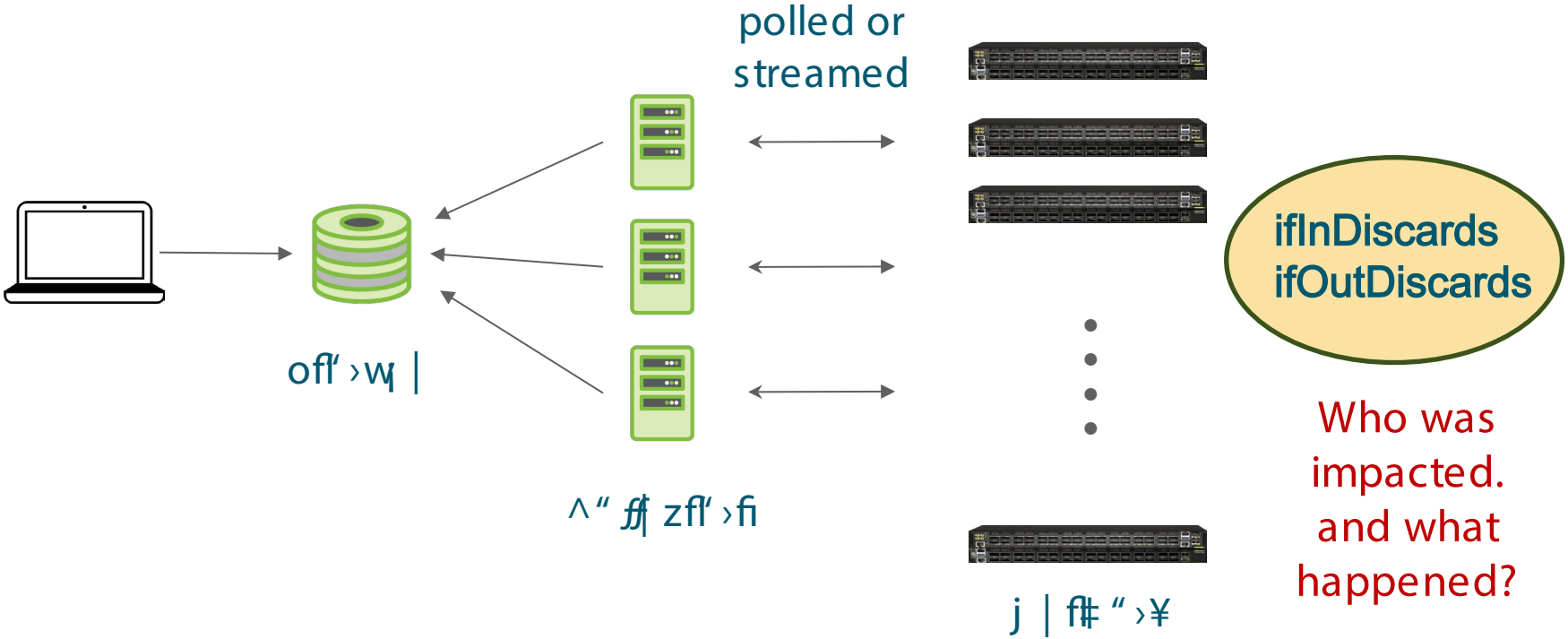
OCP GLOBAL SUMMIT | OCT 15-17, 2024
SAN JOSE, CA

NETWORKING

Leveraging open technologies to monitor packet drops in AI cluster fabrics.

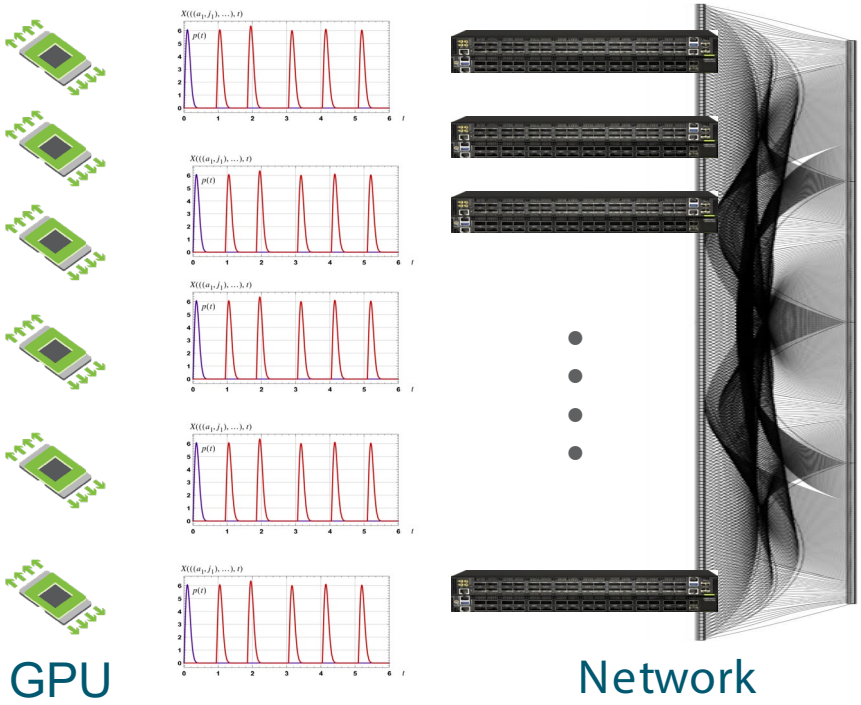
Aldrin Isaac

Classical telemetry is low fidelity at high cost



High fidelity telemetry is needed for AI training

synchronized bursts → buffer overflows within shallow-buffer DC switches



The high bandwidth and synchronous bursty traffic patterns in AI model training are not well-suited for its low tolerance to packet drops.

Low fidelity telemetry is inadequate for understanding and performance tuning AI training and fabrics.

AI collective comms needs and effects are not static

- **Model effects** : Traffic patterns associated to building an AI model for LLMs may be different from that of building a vision model or other models.
- **Placement effects** : Placement of processing within collectives can result in stresses in different parts of the AI fabric from one training run to the next as placement of processing within collectives change according to GPU availability.
- **Optimization effects** : Iterative improvements to a specific model can also change over time resulting in a different traffic expression on the network.

Adapting the network and work placement to changes in model training requires continuous high fidelity drop signals.

In-band Network Telemetry (INT), what happened?

- In 2015 the concept of "in-band network telemetry" emerged as a solution to real-time fine-grained telemetry for high performance networks.
  - Multiple "open" approaches came into discussion,
    - Examples: INT, IOAM (RFC9197/9322/9326), IFA, TCP-INT
  - Many features were considered,
  - Many papers and articles were written.

- **Eight years later, none are mainstream** . Some reasons being
  - The over-emphasis on esoteric use cases coupled with analysis-paralysis among INT innovators,
  - The variations of incompatible implementations that the industry players ultimately produced and dug their heels into.

What MOST operators wanted from INT is simple

Why did a packet that departed from a source machine not arrive at the target machine?

IOW, operators want detailed information about packets that are DROPPED

- Where they are dropped,
- Why they are dropped,
- Who was affected.

They need this in a universal format that can be easily overlaid on their view of active flows in their network.

Solution: sFlow Drop Notification

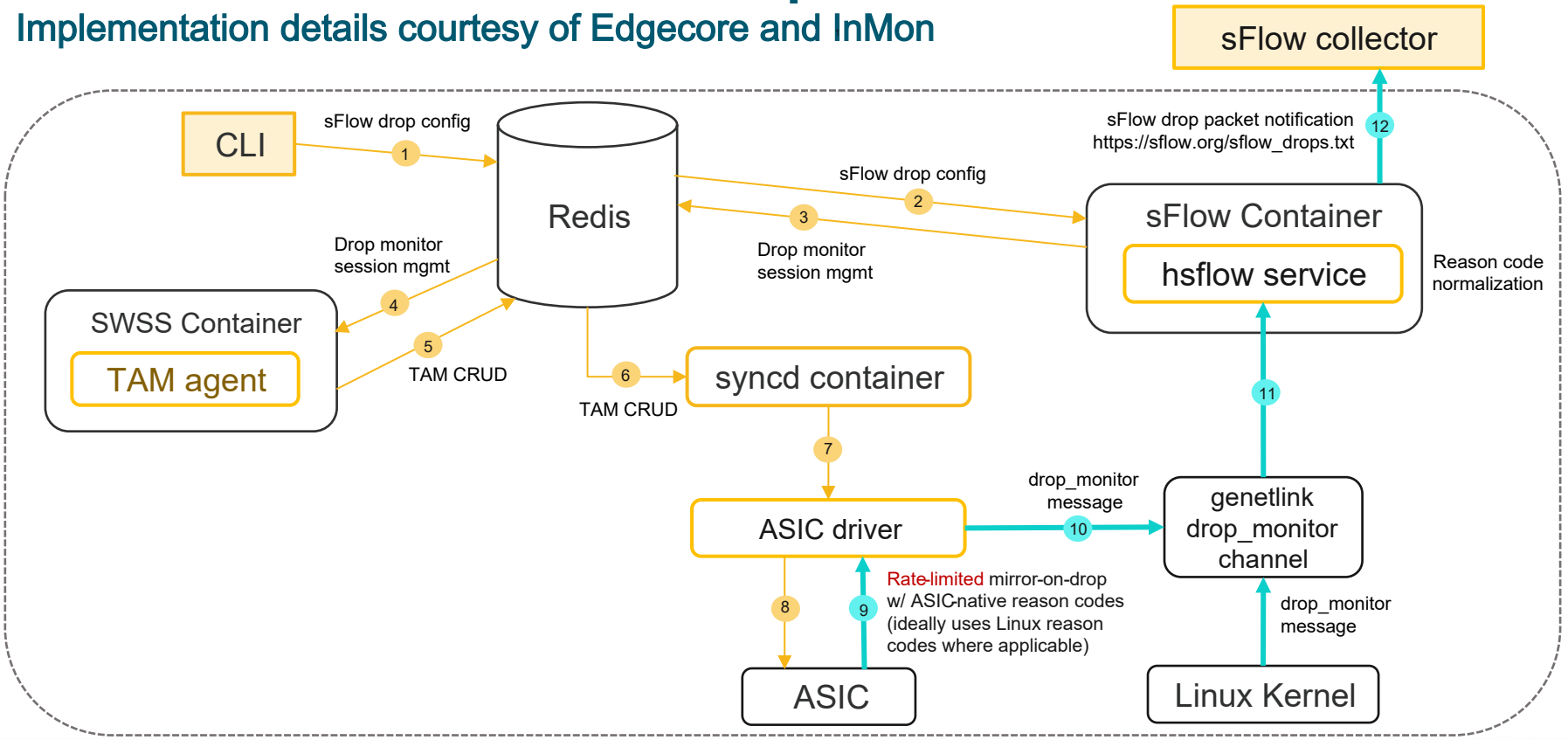
sFlow Drop Notification is another function of sFlow that leverages the mirror-on-drop capability of several DC class ASIC to capture packet drops with their reason code in a unified way.

What does sFlow Drop Notification solve?

- Unified format for drop notifications
- Unified drop reason codes
- Homogeneous device configuration regardless of ASIC
- Operationally pragmatic (ex: does not expect huge high-speed storage)
- Extension of a widely deployed flow monitoring application (sFlow)

SONiC sFlow reference implementation

Implementation details courtesy of Edgecore and InMon



Call to action: sFlow Drop Monitor support

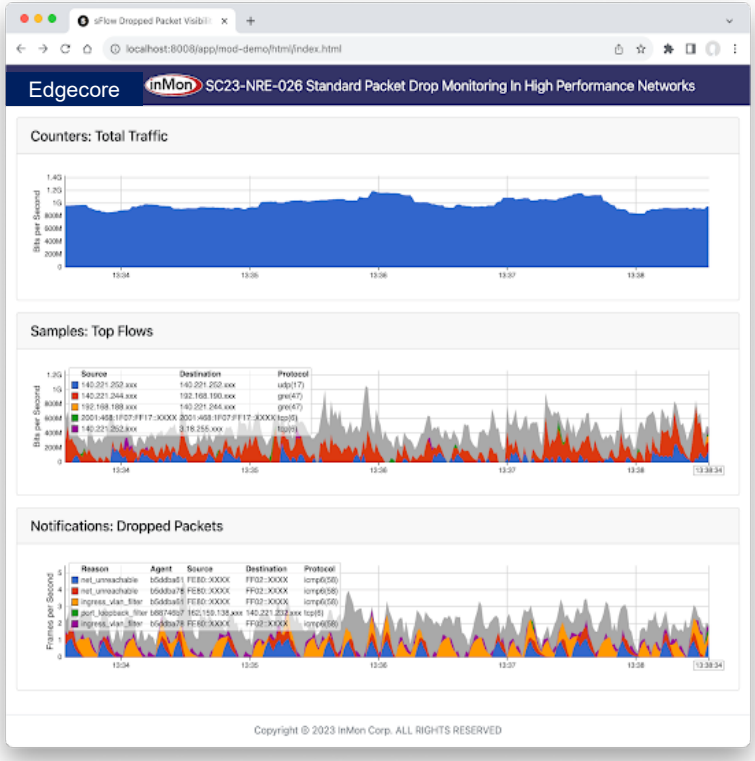
Calling on switching ASIC providers to participate in defining and implementing the SAI TAM APIs and underlying mechanisms for drop notifications.

Calling on network operators to accelerate availability of sFlow Drop Notification for SAI using their influence.

- **ASIC Support:** ASIC must provide detailed drop reasons required for sFlow Drop Notification. Vendor ASIC drivers should be free to report all drop reasons relevant to forwarding pipeline.
- **SAI Integration:** SAI must implement support for mirror-on-drop with drop reason and expose these drop reasons through the Netlink drop_monitor channel.
- **sFlow agent**: sFlow must normalize reason codes to a standardized sFlow drop reason set, as well as report ASIC-native reason codes for troubleshooting.

- Additional information (URL links):
  - sFlow agent: https://github.com/sflow/host-sflow
  - sFlow drop notification structures: https://sflow.org/sflow_drops.txt, https://sflow.org/sflow_drops_reasons.txt
  - SAI TAM drop reason definition: https://github.com/opencomputeproject/SAI/blob/master/inc/saitam.h#L1878
  - Netlink drop_monitor reasons: https://www.kernel.org/doc/html/latest/networking/devlink/devlink-trap.html
  - For more information on where to get started, contact: Madhu Paluru, Aviz Networks (madhupa@aviznetworks.com)

Call to action: Controller/NMS support



- Presently, InMon offers a tool that can capture sFlow packets and display application-related information.
- More visual tools to display drop rate with reason would be valuable for network administrators.

p¢w ¥"¶“ –

k «| ' "_ ﬀiz–ﬁﬁ£' '