Debugging Transient Faults in Data Center Networks using “Synchronized Network-wide Packet Histories”

Pravein Govindan Kannan  Nishant Budhdev  Raj Joshi  Mun Choon Chan

IBM Research  National University of Singapore
Cloud Reliability is Critical

Cloud Downtime is Expensive

$300K/ Hour - Gartner report
Data Center Network Failures

Data center incidents at Facebook from '11 to ‘18
[IMC ‘18]

Root-cause could not be determined:
1) Transient
2) Inability to correlate events
Transient Faults: Microbursts

“μbursts: periods of high utilization lasting less than 1 ms, exist in production data centers.”

“They encompass most congestion events.”

“The p90 burst duration is ≤200 μs.”
Transient Faults: Microbursts

Reasons Could be:
1) Synchronized Application Traffic
2) Network Queueing
3) Heavy Hitter
4) Network Misconfiguration

High Queueing due to Burst
What do we need from the network?

Visibility

Retrospection

Correlation
Programmable Networks

Per-packet Postcards
[NetSight*, INT-XD]

Per-packet Postcards
[NetSight*, INT-XD]

Not scalable, due to throughput and storage overheads.

Creates a post-card per packet.

Visibility  

Retrospection

In-band Network Telemetry

[INT]

Reduces goodput by upto 20% [PINT, SIGCOMM ‘20]

Attaches telemetry information in the packet
In-band Network Telemetry

[INT]

Network Faults occur infrequently

[Facebook, IMC’18]

Reactive Approaches lose History
How do we solve? : An Outline

**Visibility**
- Compressed Packet Records in switch memory

**Retrospection**
- Export record recent history of packet records
- Fault detection in the data-plane

**Correlation**
- Data-Plane Time Synchronization
  - DPTP[SOSR ‘19]
  - DTP[SIGCOMM ‘16]

**SyNDB**
Packet Records

Compression : Packet Records (precords)

<table>
<thead>
<tr>
<th>Hash</th>
<th>Time_in</th>
<th>Time_queue</th>
<th>Ingress Port</th>
</tr>
</thead>
</table>

Ring Buffer in switches

Time-Synchronization using DPTP
In-Network Fault Detection

Programmable Fault Triggers
e.g.
Queueing delay > Threshold

High Queueing due to Burst

Synchronized Fan-In Traffic
Packet record Collection

1) Notify the entire Network

2) Send Records to collector

High Queueing due to Burst
Query-based Debugging

Collector (Debugger)

High Queueing due to Burst

Packets Seen at S7

Total Packets

H1  H2  H3  H4  H5  H6

Hosts
SyNDB: Query-based Debugging

SELECT switch, time_in
FROM packetrecords
WHERE hash
IN (SELECT hash FROM packetrecords AS A
JOIN triggers as T
ON (A.time_in < T.time AND A.switch = T.switch))
AND switch IN (SELECT switch FROM switches WHERE type = "tor");

SELECT time_queue FROM packetrecords where switch=7;

Synchronized Fan-In Traffic

High Queueing due to Burst
SyNDB Runtime

```
precord {
  fields {
    field_list_1;
    field_list_2;
    ...
  }
  default_field : field_list_{x};
  history     : {y};
  future      : {z};
  time_window : {t ms};
}
trigger {
  conditions {
    c1 = condition_1;
    c2 = condition_2;
    ...
  }
  collection {
    c1 [&] c2' [&] c3' ..
  }

Network Programmer
```
SyNDB Implementation & Evaluation

Mini-testbed (Fat-Tree 2)
Barefoot Tofino Switch (Wedge100BF-32X)
  • 1900 lines of P4 code
  • 1000 lines of Control Plane code in C

SyNDB Runtime
  • ~4000 lines of RUST code for compiler and Translation to P4

Simulation (Fat-Tree 24)
SyNDB Simulator
  • Packet-level simulator
  • ~6000 lines of C++

Consistent precord captures
Debugging Microbursts, Network misconfiguration, etc
Retrospection & Correlation (Simulation)

- Fat-Tree 24 (720 Switches, 3456 Hosts) with 100G Links (172.8 Tbps)
- Traffic Model scaled based on real-world DC* (web apps)

* “Network Traffic Characteristics of Data Centers in the Wild”, T. Benson, A. Akella, and D. A. Maltz, IMC 2010
SRAM Overhead

![SRAM Usage Chart]

5 – 7 MB SRAM
SyNDB
Synchronized Network Debugger

• A first of its kind network-wide Synchronized Debugging framework for network-wide debugging.

• SyNDB can be implemented in existing switches and support several ms (100’s of RTTs) of packet histories.

• SyNDB exports packet histories only on detecting faults, thus saves storage and network overhead by a magnitude at line-rate.

https://github.com/rajkiranjoshi/sybnd-sim