TimerTasks

Towards Time-driven Execution in Programmable Dataplanes

Raj Joshi, Ben Leong, Mun Choon Chan

National University of Singapore

ACM SIGCOMM
August 19-24, 2019
Beijing, China

Problem

Programmable dataplanes provide event-driven execution

- By means of the match-action paradigm
- Event = arrival of a packet OR a *specific* packet (match specification)

Any programmed logic executes when there is a packet

- But if there is no packet, nothing happens!
- Not a problem for implementing flexible *forwarding* pipelines

Increasingly used for advanced applications

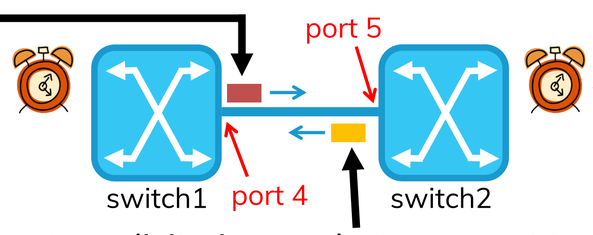
- In-network protocols
- Distributed systems that are co-designed with the network

A purely event-driven execution model is insufficient!

Need for Time-driven Execution

Fast and efficient link failure detection in the dataplane

(1) Packets from normal traffic act as "heartbeats"



(2) Timer starts when there is no normal traffic

(3) On each timeout, explicit ping packets are sent to check connectivity

Not possible to achieve with a packet-driven execution model
→ An action is required in response to the absence of a packet

Avoid aging metrics, refresh them!

- In CONGA [SIGCOMM '14] and DistCache [FAST '19], source ToR *blindly* ages the metrics when there is no reverse traffic
- Instead, destination ToR can explicitly refresh the metrics in the absence of reverse traffic

Mechanisms for reliable message delivery in the dataplane

- Msg retransmission in dataplane in the absence of an ACK (timeout)

Simple Periodic Tasks

- Periodically measure link utilization every 25µs (Zhang et al. [IMC '17])

The TimerTask Abstraction

What kind of execution semantics do we need?

- If (condition == true) for certain timeout → execute an action
- e.g. If no heartbeat packet within 100ms, inform the master
- Periodic tasks can be expressed with an always true condition

A TimerTask consists of:

- ResetMatch: same as a P4 "match" specification (just different name)
- Timeout: a positive constant
- Action: standard P4 action

```
timertask link_failure_detection {
    resetmatch = {
        standard_metadata.ingress_port: exact;
    }
    timeout = 3;  // in microseconds
    action = send_ping_request;
}
```

How does it work?

- The system runtime implements an implicit timer
- If a packet matches the ResetMatch specification: the timer is reset
- If the timer times out: the action is executed & the timer is restarted

Orchestrating TimerTasks

Challenge

- Time-driven abstraction on top of packet-driven hardware

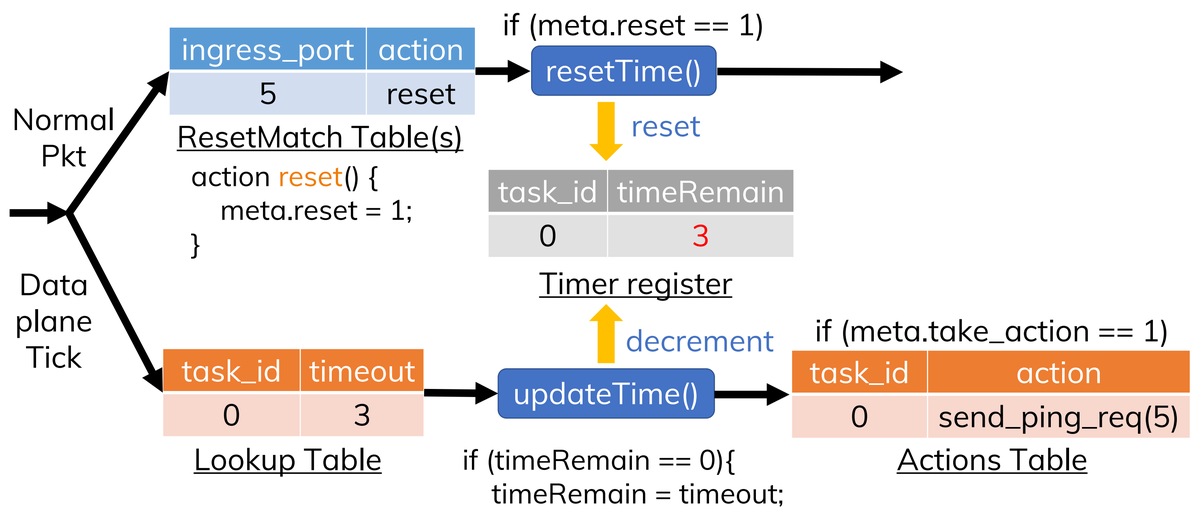
Solution approach: inspired from Linux OS

- Linux is event-driven
- Provides time-driven execution via periodic events (kernel ticks)

"Periodic-Event Framework" for the dataplane

- Dataplane ticks: periodic and regular packets
- On-chip hardware packet generator: source of dataplane ticks
- Orchestrate TimerTasks: decrement and reset timers, perform actions

TimerTask link_failure_detection for switch2 (port 5)



Evaluation Case Studies

Evaluation Setup for Periodic-Event Framework

- Barefoot Tofino switch and two x86 servers in linear topology
- Dataplane ticks: Tofino's on-chip packet generator

Dataplane tick inter-arrival time

- 99th percentile error in tick inter-arrival time (1µs to 100ms) < 0.1%

1) Fast and efficient link failure detection protocol

- Explicitly pings adjacent switch when no packet arrives for 3µs
- Detects link failures within 6µs

2) Metric refreshing mechanism

- Explicitly updates piggybacked metrics in lieu of normal traffic
- Future work: quantify application-level benefits

3) High-resolution network measurements

- Periodic task to read switch counters at 1µs interval

Future Work

P4 Program with TimerTasks → (TimerTask Compiler) → Standard P4 Program → (Target-specific Compiler) → P4 Target

TimerTask Compiler

- Timer start and stop APIs in the dataplane
- Table placement: TimerTasks interact with rest of the P4 program
- Multiplexing: single dataplane tick packet to update multiple timers
- TimerTask runtime (control plane): add/remove TimerTask "instances"

Native TimerTasks in hardware