

# TAEP

## Traffic Analysis and Experimentation in a Production Network

Marcel Neuhausler, Michael Jensen

AT&T Foundry, Palo Alto

TAEP :

A/B Experiments in Production

“By combining the power of software with the scientific rigor of **controlled experiments**, your company can create a **learning lab**. The returns you reap in **cost savings**, new revenue, and improved user experience can be huge. If you want to gain a **competitive advantage**, your firm should build an experimentation capability and **master the science of conducting online tests**.”

<https://hbr.org/2017/09/the-surprising-power-of-online-experiments>

TAEP :

## A/B Experiments in a Production Network

A/B testing on Network Traffic is harder than for UI/UX due to:

- Speed and amount of network traffic
- Missing control APIs to setup and run an experiments in an automated way
- Missing “Big Data” solutions
- Missing insights which are needed to define A, the base line
- Harder to limited impact of a failed experiment
- Missing Network Protocols

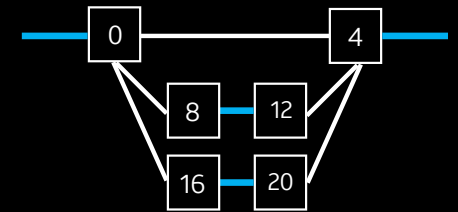
# TAEP:

## Ingredients

### Keep it simple, Cabling

Barefoot Wedge 100BF-32X

1 straight through path  
2 traffic-divert options



# TAEP:

## Ingredients

Keep it simple, P4, Forward

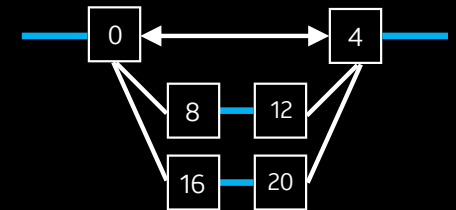
```

/*****
/* Forward Packet
*****/

action set_egr(egress_spec) {
    modify_field(ig_intr_md_for_tm.ucast_egress_port, egress_spec);
}

table forward {
    reads {
        ig_intr_md.ingress_port : exact;
    }
    actions {
        set_egr;
        _nop;
    }
    size: BAREFOOT_MAX_PORTS;
}

```



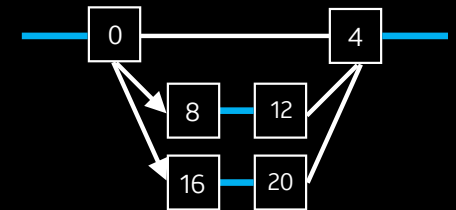
ingress port	action	egress port
0	set_egr	4
4	set_egr	0

# TAEP:

## Ingredients

Keep it simple, P4, Divert

```
/* *****  
/* Divert Packet  
/* *****  
  
table divert {  
  reads {  
    ig_intr_md.ingress_port : exact;  
    ipv4.dstAddr : ternary;  
    ipv4.srcAddr : ternary;  
  }  
  actions {  
    set_egr;  
    _nop;  
  }  
  size: BAREFOOT_MAX_PORTS;  
}
```



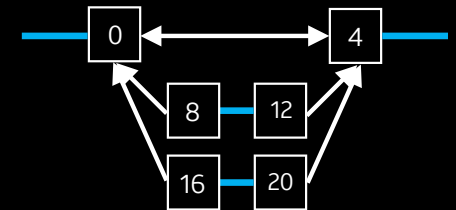
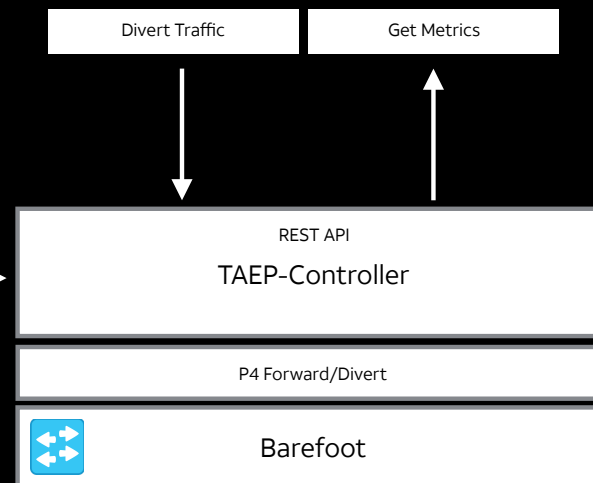
ingress port	ip address	prefix length	action	egress port
0	192.168.1.12	32	set_egr	8
0	192.169.1.0	24	set_egr	16

# TAEP:

## Ingredients

Keep it simple, Controller

```
enable-labeling: true
api-port: 8100
ports:
  - number: 0
    speed: 40
    autoneg-disabled: true
  - number: 4
    speed: 40
    autoneg-disabled: true
  - number: 8
    speed: 40
  - number: 12
    speed: 40
  - number: 16
    speed: 40
  - number: 20
    speed: 40
connections:
  - from: 0
    to: 4
    type: bidirectional
  - from: 12
    to: 4
    type: unidirectional
  - from: 20
    to: 4
    type: unidirectional
  - from: 8
    to: 0
    type: unidirectional
  - from: 16
    to: 0
    type: unidirectional
```



# TAEP:

## Ingredients

Keep it simple, Analytics Stack

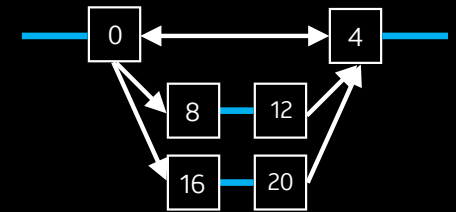
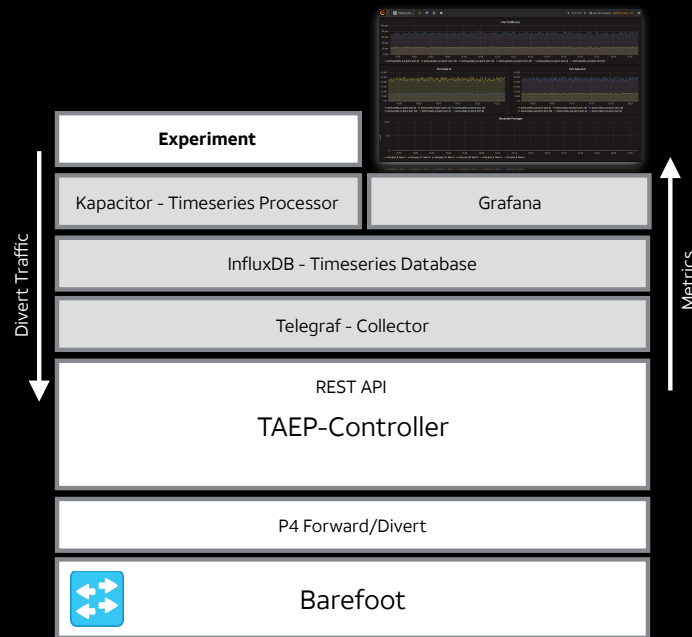
Open-Source Projects:

- Timeseries Stack from InfluxData
- Dashboard by Grafana

Experiment:

- Written in Python
- Metrics pushed by Kapacitor
- Talks to Controller via REST API

Everything runs on the Barefoot Switch





# TAEP:

## Reinforcement Learning

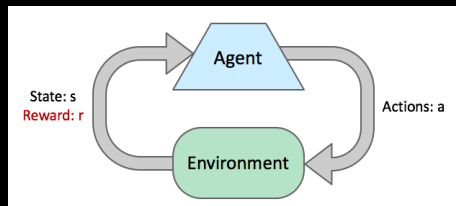
### Do some advanced experiments

Goal:

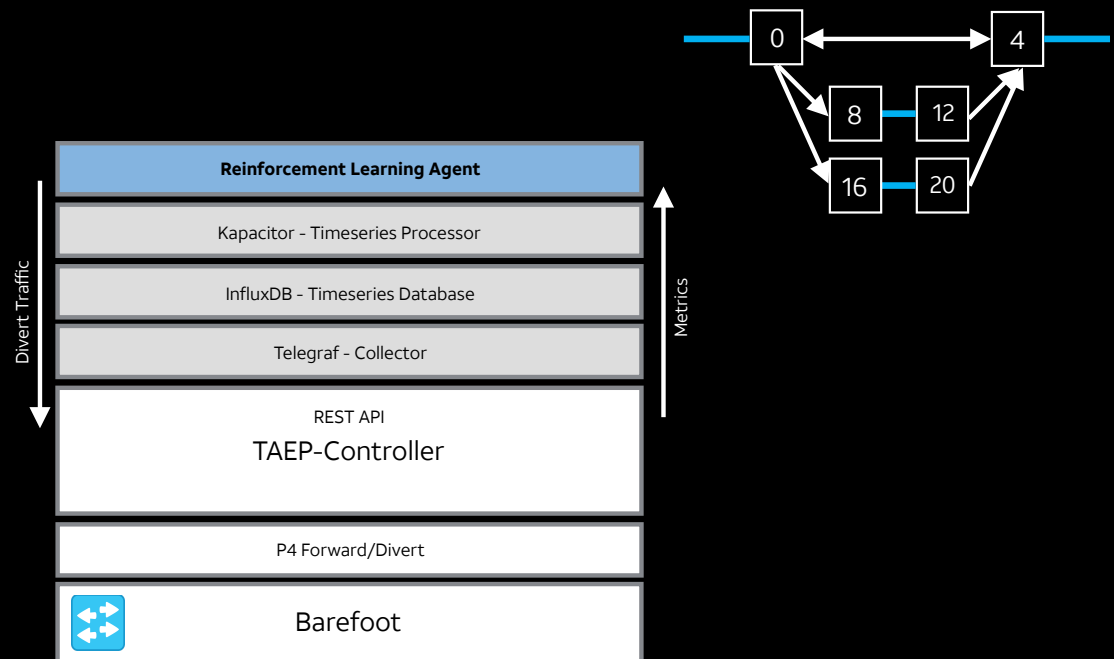
Balance traffic between two links in a 0.3:0.7 ratio using a Reinforcement Learning Agent

Setup:

- Provide a set of pre-defined divert rules to the agent as action-space
- Agent explores different combinations of divert rules based on action- and state-space
- Agent calculates reward based on link measurements



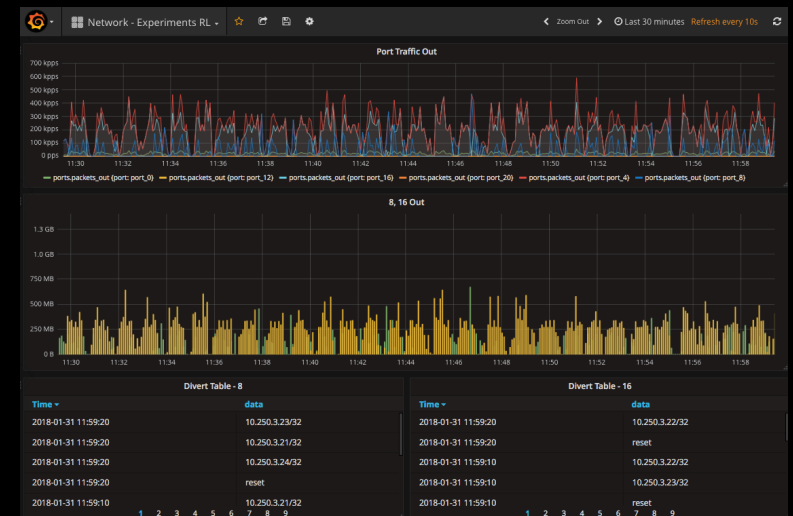
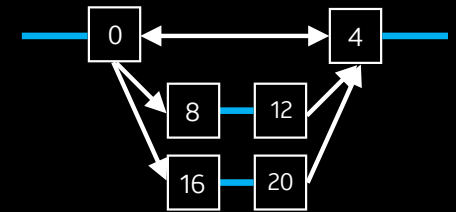
[http://ai.berkeley.edu/lecture\\_slides.html](http://ai.berkeley.edu/lecture_slides.html)



# TAEP:

# Reinforcement Learning

## Some Lab Results



# TAEP:

## Heavy Hitter Detection

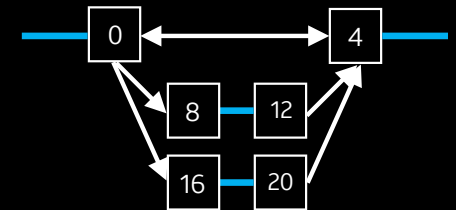
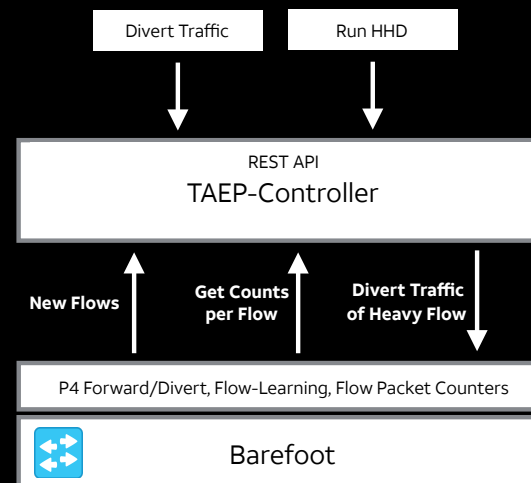
### Do some advanced experiments

Goal:

Automatically divert traffic of “heaviest” flow

Setup:

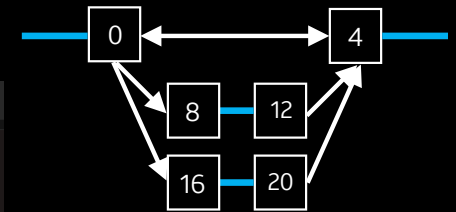
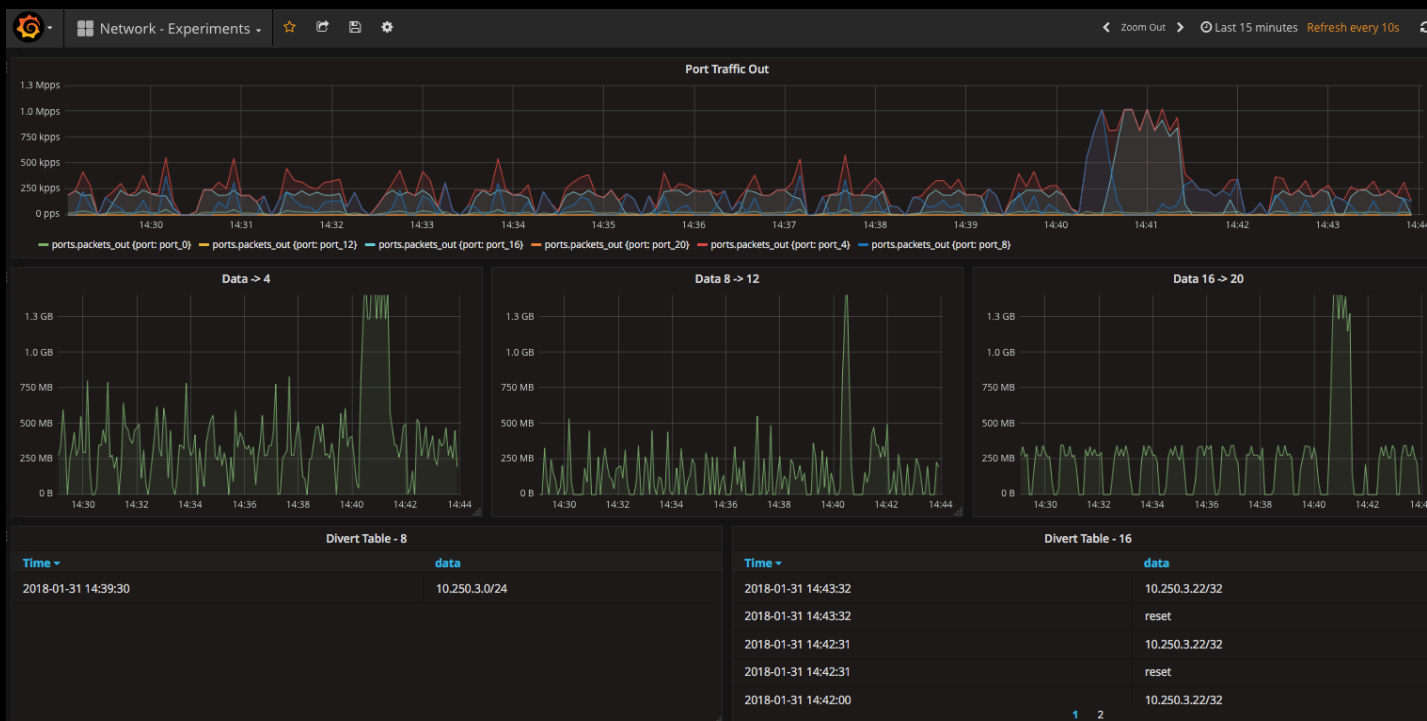
- Divert traffic of a subnet (test traffic) through 8-12
- During a predefined time window count packets of each flow in that test traffic (count-min sketch)
- Resolve flow with most packets
- Divert packets for heaviest flow during the next time window through 16-20



# TAEP:

## Heavy Hitter Detection

### Some Lab Results



TAEP :

Lessons Learned

(Preliminary) Lessons Learned from running TAEP in a live network:

- *It works*, no packets got hurt during our 4 months of live tests
- Barefoot/Tofino ASIC offers a fast and stable runtime for P4 programs
- *It's hard*, confronted with a new set of challenging but interesting problems
- How to control a P4 program running at high-speed in the data-plane with a massively slower control plane
- How to split your application logic (example HHD) across data-plane (P4) and control-plane
- *It's fun*, watching the Reinforcement Learning Algorithm trying to do its magic on production traffic

TAEP:

Traffic Analysis and Experimentation Platform



Questions



<https://github.com/att-innovate/taep>

# TAEP:

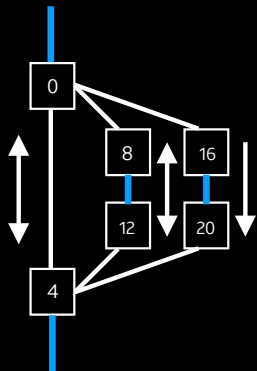
# Traffic Analysis and Experimentation Platform

Barefoot Wedge 100BF-32X

1 straight through path  
2 traffic-divert options

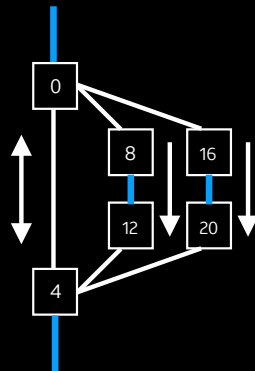


## 1. Traffic Analysis



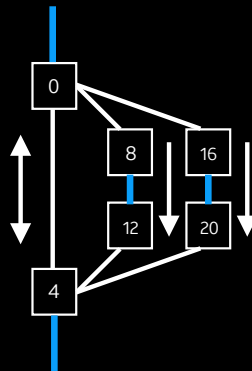
Divert areas of interest  
Example:  
Traffic from or to specific sub-net

## 2. Heavy Hitter Detection



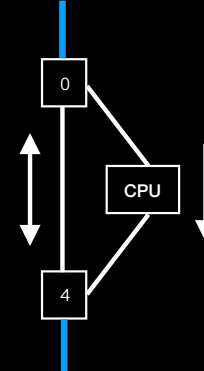
Divert test traffic through 8-12  
Find heaviest flow  
Divert heaviest flow through 16-20

## 3. Reinforcement Learning



Use Reinforcement Learning Agent  
Balance test traffic at a 0.3:0.7 ratio  
between 8-12 and 16-20

## 4. Chaos Engineering



Drop certain traffic  
Modify certain traffic  
Add latency to certain traffic

