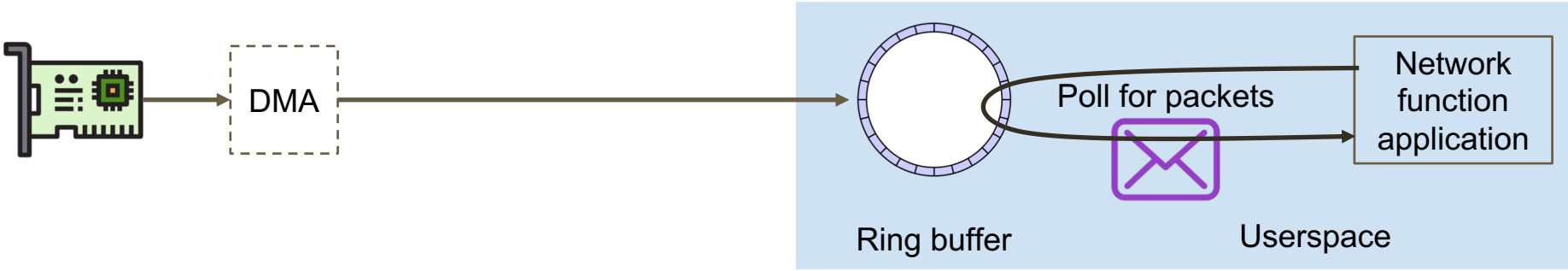



# A Black-Box Approach for Estimating Utilization of Polled IO Network Functions

**Harshit Gupta**[+], Abhigyan Sharma*,
Alex Zelezniak*, Minsung Jang*,
Umakishore Ramachandran[+]

[+]*Georgia Tech, *AT&T Labs Research*

# Polled I/O for efficient Network Functions (NFs)

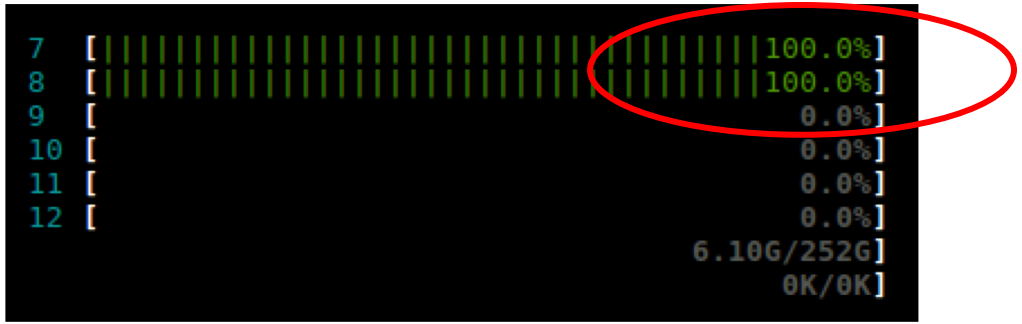


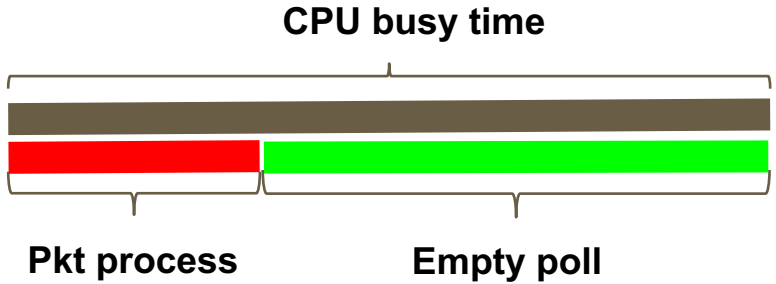

- Packet DMA from NIC to ring **buffer in userspace**
- **Polling** for incoming packets (**no interrupts**)
- **No kernel processing overheads** like context switching, packet copying
  - **Order of magnitude improvement** in packet processing throughput

Thriving open-source projects (e.g. DPDK, fd.io, Open vSwitch)

# Polling makes util% estimation of NFs difficult

- Tools like top shows cores always at **100% utilization even at zero traffic!**




INSIGHT: Instead of $\frac{\text{busy time}}{\text{total time}}$ use $\frac{\textbf{pkt process time}}{\textbf{total time}}$ as utilization metric

# Existing approaches

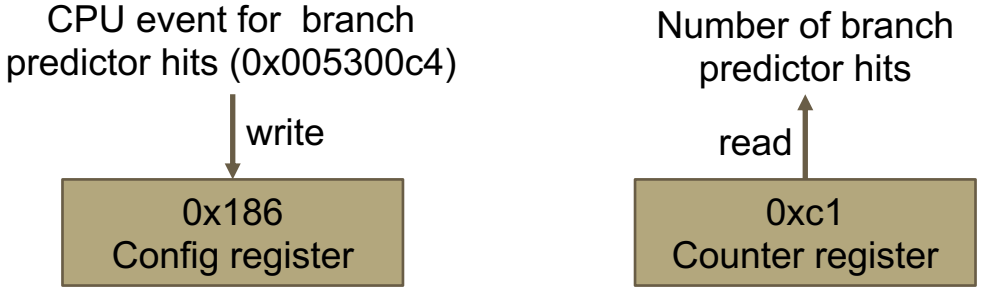
- **Niccolini ATC12** Driver reports NIC queue occupancy
  - **Driver modification, app recompilation**
- **Trifonov TR17** NF instrumentation to output empty polls
  - **App modification** for all NFs
- **Cao HotCloud17** Learn variation of application metrics exposed by NF (e.g. num_http_req for HTTP proxy)
  - Application **metrics known and exposed** to network provider

**Not well suited for network providers** like **AT&T** where NF is provided as **a black box** by vendors

Can we learn how the CPU behaves when it is busy processing packets ?

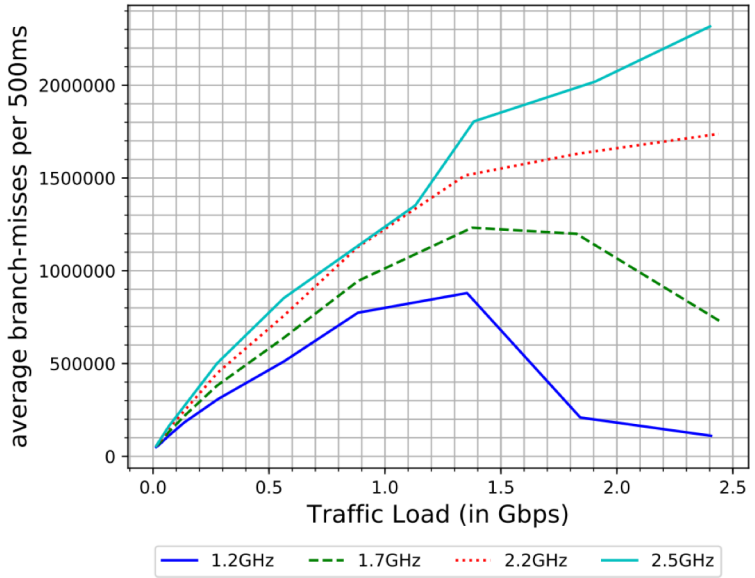
# Hardware performance counters

- Are programmable, per-core CPU **registers for counting CPU events**
- Can count 100s of events, e.g., branch (mis)predictions, cache hit/miss ...
  - ... **but only few at a time** up to the number of hardware registers
- Enable **low overhead (tens of cycles)** collection of program execution metrics
- Are available in any Intel or AMD server CPU

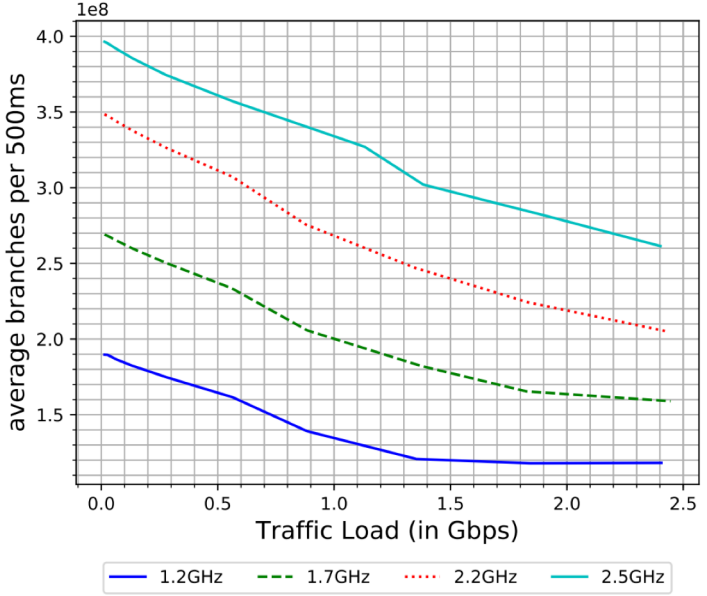


# High correlation of counters with load on NF

- **270 out of 714** events show **> 95% correlation** with traffic load



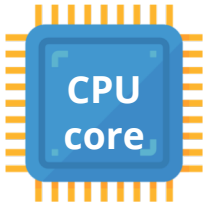

Branch prediction misses (midpredictions)

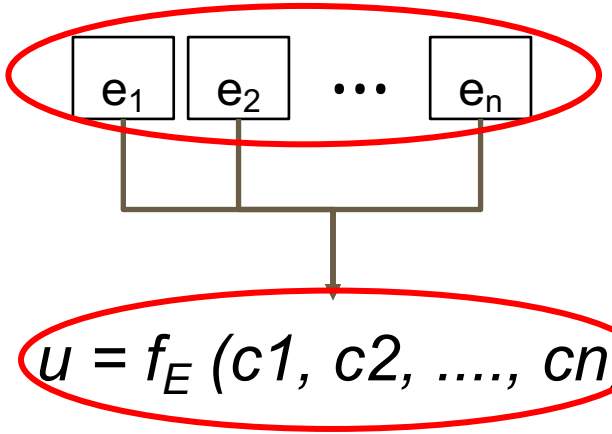



Branch instructions

*Number of events every 500ms Intel Xeon E5. CPU, for DPDK L3FWD app*

# Hardware counter based estimator functions




**Problem 1 : Choosing right set of events E**

**Problem 2 : Building accurate and robust mapping function**

$f_E$ is an estimator function that maps counter values to a utilization value **u**

Input of $f_E$ : counter values **(c1, c2, …, cn)** of event set **E = {e1, e2, …, en }**
($E \subseteq E\_ALL$)

Output of $f_E$ : Utilization **u** $\in [0, 100]$

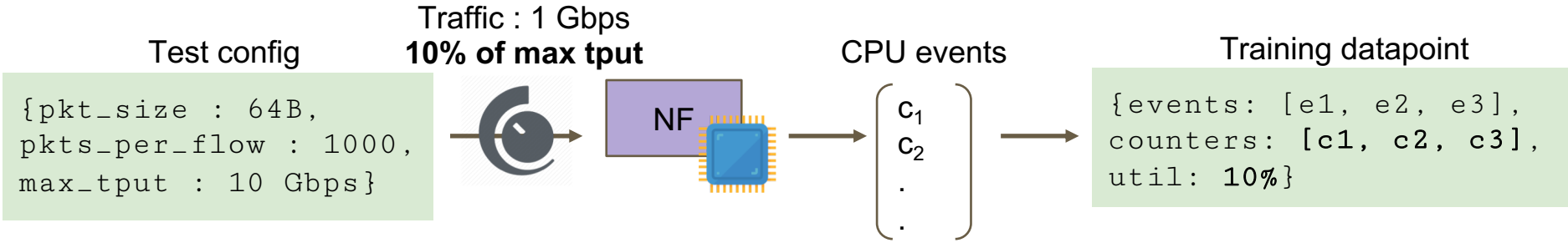
# Data-driven estimator selection

- Build 3 families of estimator functions $\mathbf{f_E}$ with **|E| = 1, 2, 3** (E ⊆ E_ALL)
  - Each $\mathbf{f_E}$ is implemented as a linear regression model

- From each family select $\mathbf{f_E}$ that minimizes average estimation error $\mathbf{|f_E(.) - u|}$
  - Evaluate error using **3-fold cross-validation**

# Generating the training dataset

Example : Generating training data for estimator $\mathbf{f_E}$ , E = {e1, e2, e3}




- Use **multiple test configurations** (diff packet sizes, flow lengths, payload size)
  - **Representative of production** workloads
- **Vary traffic load** for each test config (0%, 10%, 20%, ... , 100%)

# Assumptions

- Fixed hardware (same set of CPU events, BIOS settings)
- Run-to-completion packet processing model
- CPU as the bottleneck resource

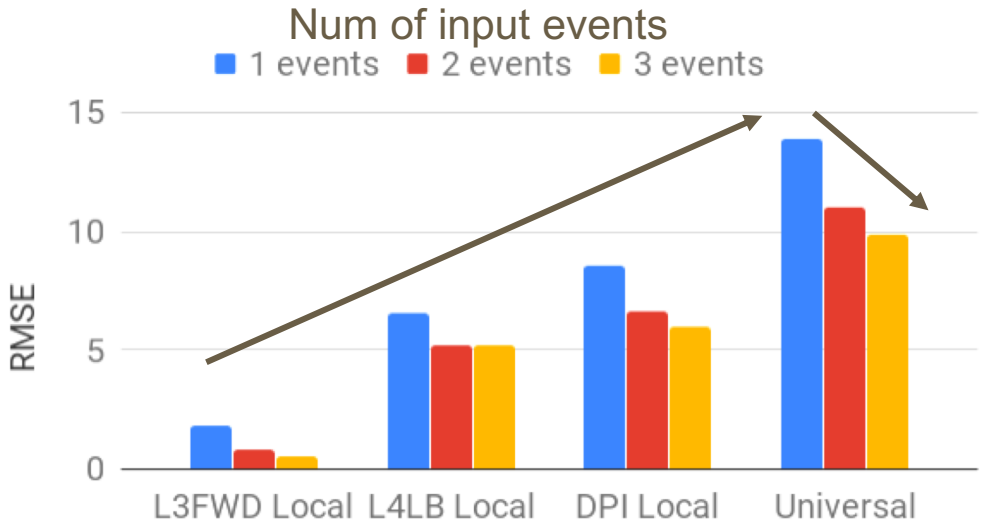
# Network Functions evaluated

1. **Stateless NF**
   *L3FWD (Layer 3 forwarder)*
   Lookup routing table for every packet


2. **Stateful NF**
   *L4LB (Layer 4 load balancer)*
   Per-flow state storing backend server


3. **Compute-intensive NF**
   *DPI (Deep Packet Inspection)*
   Pattern matching for each flow

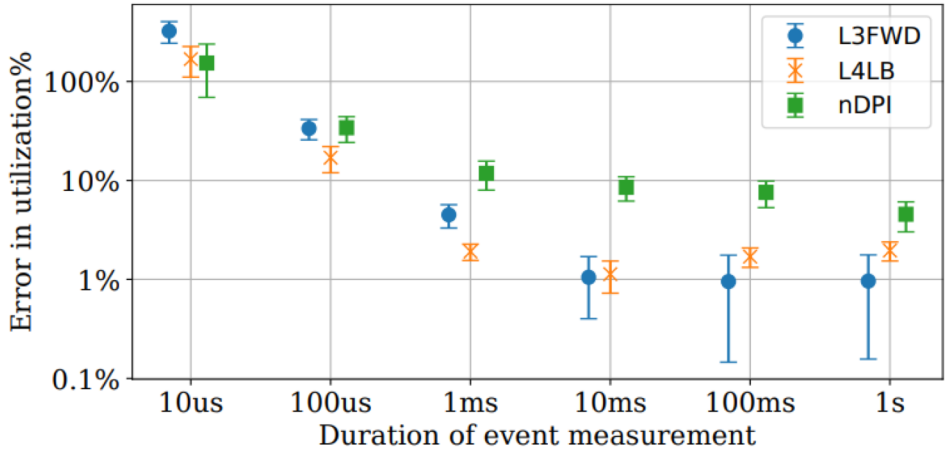
# Util% estimation error of best estimators




- **Local estimators have lower error** than universal estimators.
- Estimation **error increases with NF complexity** (DPI > L4LB > L3FWD)
- Using **more events** as input **improves error**
- Best universal estimator has **<10% error**

# Testing on unseen network traffic profiles

Result for heterogeneous workload with mix of pkt sizes and flow lengths




Util% estimation **error remains below 10%** for best estimators

Counting events for **10 ms** provides the best **tradeoff between latency and accuracy**

# Summary

- Proposed polled IO utilization estimation based on hardware counters
- Works purely at host level for black box NFs
- Shows low errors (< 10%) on stateless, stateful and compute-intensive NFs

# Future Work

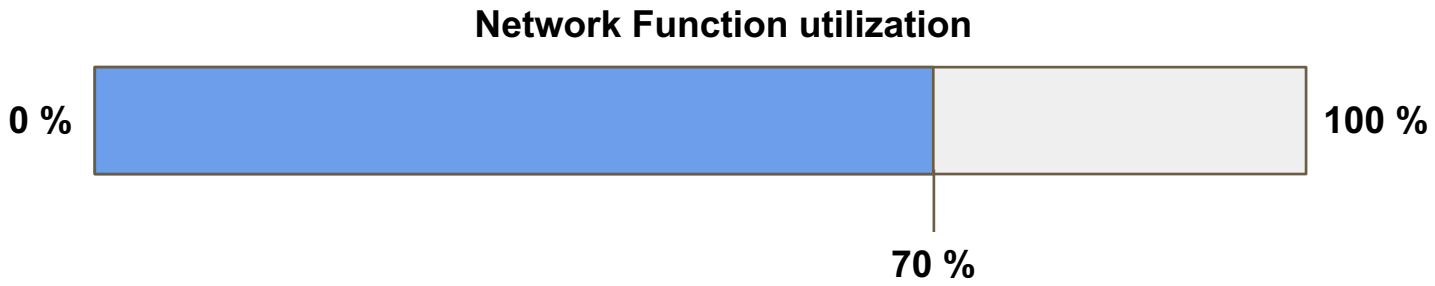
- Universal estimators (more input events, neural network-based estimators)
- Additional NFs (virtualized NFs, cross-core packet processing NFs)
- Use cases (power management, load balancing, workload placement)

Georgia Tech

Harshit Gupta (harshitg@gatech.edu)

# Vision

```
$ nf-top --cores=6,7 --events=0x005300c4,0x005300c5
        --mapping-function=./firewall.map
```



# Backup slides

# Counting CPU event – “branch prediction hits”

```
msr_fd = open("/dev/cpu/2/msr", O_RDWR | O_SYNC);
    // descriptor for registers on CPU core 2
c = 0x005300c4; // CPU event for branch hits
ret = pwrite(msr_fd, c, sizeof(c), 0x186);
    // configuration register 0x186 will record branch hits
...
ret = pread(msr_fd, &val, sizeof(val), 0xc1);
    // read counters from counter register 0xc1
```

# Why not use all CPU events available ?

- > 700 CPU events available
- A lot of them are not correlated with input traffic ⇒ no useful data
- There are limited number of performance monitoring registers on CPUs
  - Can only count so many events at once
  - To count 8 events for 2 sec each on 4 registers , we need (8/4)*2secs

# Test configurations used

- L3FWD (Stateless NF) :
  - Diverse packet sizes (64B – 512B)
- L4LB (Stateful NF) :
  - Diverse flow lengths (10 – 1000 pkts per flow)
  - Smallest size packets (64B for max load on NF)
- DPI (Compute-intensive NF) :
  - HTTP connections of varying lengths (1 KB – 1024 KB)
  - Patterns cover IP, transport and application layer

# Experimental testbed

- Setup consists of 2 servers
  - Server 1 : Device-under-test
  - Server 2 : Traffic Generator
  - 2x10G network interfaces (Intel 82599ES 10 Gbps SFI/SFP+ NIC)
  - 2 CPUs (Intel Xeon E5-2680 v3).
- Connected by 10G switch
- TRex packet generator used for generating traffic
  - L4LB and DPI are tested using TRex stateful mode

# Input events of best estimators

| NF | CPU events |
|---|---|
| L3FWD | ILD_STALL.LCP |
| | L2_TRANS.RFO |
| L4LB | L1D_PEND_MISS.REQUEST_FB_FULL |
| | BR_INST_EXEC.TAKEN_CONDITIONAL |
| DPI | CYCLE_ACTIVITY.CYCLES_L2_PENDING |
| | MEM_LOAD_UOPS_RETIRED.L2_MISS |

# Distribution of error across all possible estimators