

INVEST: Flow-Based Traffic Volume Estimation in Data-Plane Programmable Networks

Damu Ding[1,2], **Marco Savi**[3], **Federico Pederzolli**[1], **Domenico Siracusa**[1]

[1] *Fondazione Bruno Kessler, Trento, Italy* [2] *University of Bologna, Bologna, Italy*
[3] *University of Milano-Bicocca, Milano, Italy*

IFIP Networking conference
22nd June, 2021

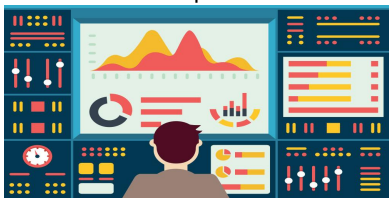
FONDAZIONE BRUNO KESSLER

ALMA MATER STUDIORUM A.D. 1088

UNIVERSITÀ DEGLI STUDI DI MILANO BICOCCA



Network monitoring functionalities in Software-Defined Networks



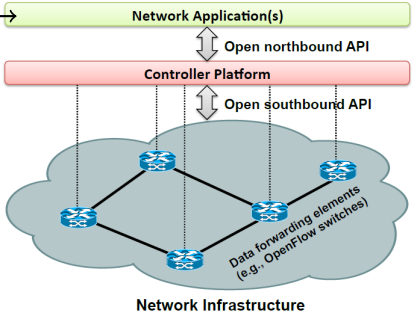
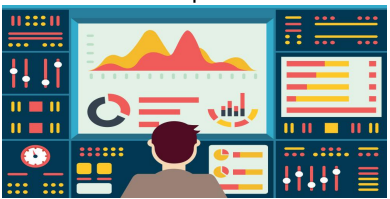
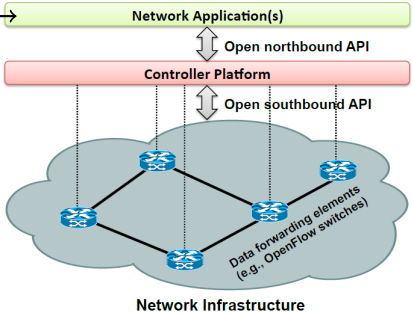


Figure source: Kreutz, Diego, et al. "Software-defined networking: A comprehensive survey." Proceedings of the IEEE 103.1 (2015): 14-76. and https://n0where.net/real-time-network-monitoring-cyberprobe

Network monitoring functionalities in Software-Defined Networks





1. Significant communication overhead
2. High latency caused by interaction
3. Cannot perform monitoring at line-rate speed (Up to 100 Gbps)

Figure source: Kreutz, Diego, et al. "Software-defined networking: A comprehensive survey." Proceedings of the IEEE 103.1 (2015): 14-76. and https://n0where.net/real-time-network-monitoring-cyberprobe

Network monitoring functionalities in Software-Defined Networks

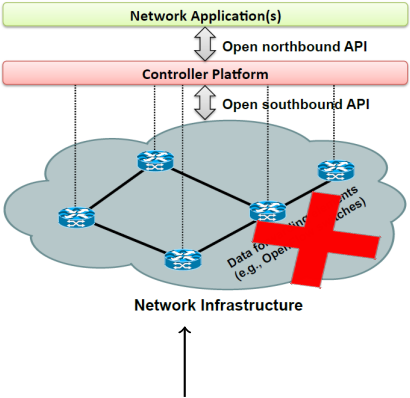
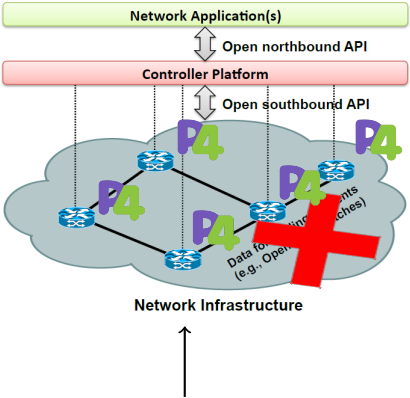


Figure source: Kreutz, Diego, et al. "Software-defined networking: A comprehensive survey." Proceedings of the IEEE 103.1 (2015): 14-76. and https://n0where.net/real-time-network-monitoring-cyberprobe

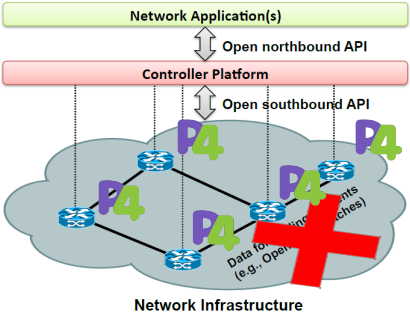
Network monitoring functionalities in Software-Defined Networks



Data plane programmable switches

Figure source: Kreutz, Diego, et al. "Software-defined networking: A comprehensive survey." Proceedings of the IEEE 103.1 (2015): 14-76. and https://n0where.net/real-time-network-monitoring-cyberprobe

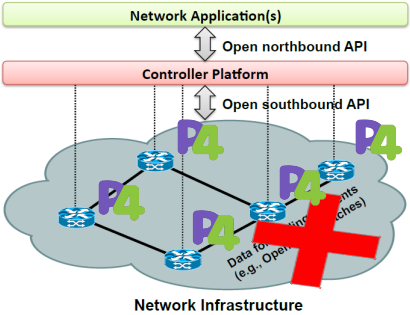
Network monitoring functionalities in Software-Defined Networks



Data plane programmable switches

Figure source: Kreutz, Diego, et al. "Software-defined networking: A comprehensive survey." Proceedings of the IEEE 103.1 (2015): 14-76. and https://n0where.net/real-time-network-monitoring-cyberprobe

Network monitoring functionalities in Software-Defined Networks

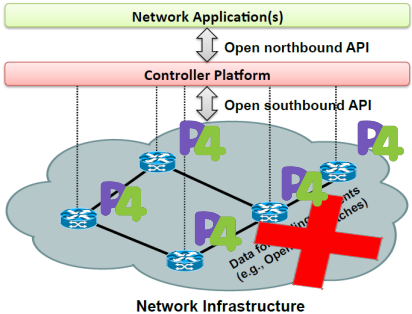


Heavy-hitter detection
Flow cardinality estimation
Network traffic entropy estimation
Traffic volume estimation
Volumetric DDoS detection

Data plane programmable switches

Figure source: Kreutz, Diego, et al. "Software-defined networking: A comprehensive survey." Proceedings of the IEEE 103.1 (2015): 14-76. and https://n0where.net/real-time-network-monitoring-cyberprobe

Network monitoring functionalities in Software-Defined Networks



Heavy-hitter detection

Flow cardinality estimation

Network traffic entropy estimation

Traffic volume estimation

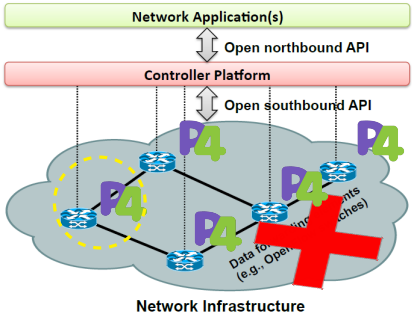
Volumetric DDoS detection

Data plane programmable switches

Figure source: Kreutz, Diego, et al. "Software-defined networking: A comprehensive survey." Proceedings of the IEEE 103.1 (2015): 14-76. and https://n0where.net/real-time-network-monitoring-cyberprobe

Damu Ding et al.ding@fbk.eu

Network monitoring functionalities in Software-Defined Networks



Heavy-hitter detection

Flow cardinality estimation

Network traffic entropy estimation

Traffic volume estimation

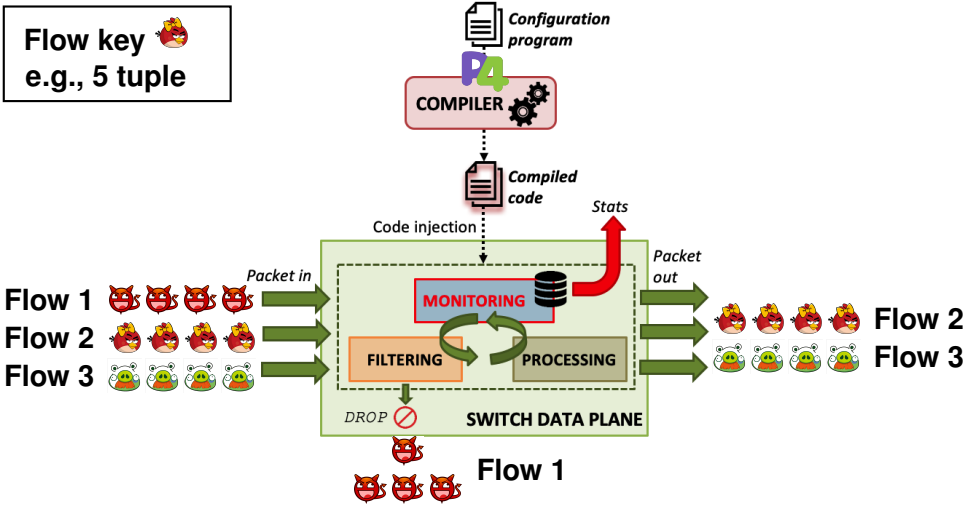
Volumetric DDoS detection

Data plane programmable switches

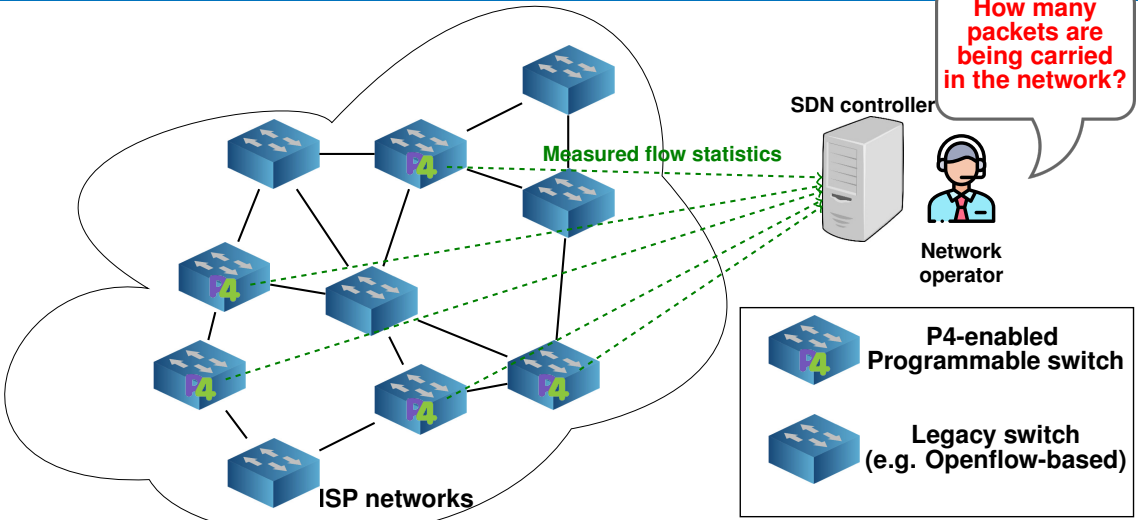
Figure source: Kreutz, Diego, et al. "Software-defined networking: A comprehensive survey." Proceedings of the IEEE 103.1 (2015): 14-76. and https://n0where.net/real-time-network-monitoring-cyberprobe

Damu Ding et al.ding@fbk.eu

P4-enabled programmable data plane for monitoring



Traffic volume estimation

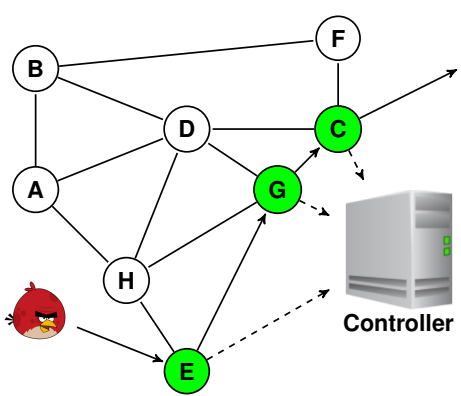


Why traffic volume estimation?

Traffic volume is required by many network-wide monitoring tasks (i.e. monitor the whole network state instead of a single node)

- Heavy-hitter detection
  - Set an adaptive threshold (i.e. a fraction of network traffic volume)
- Heavy-changer detection
  - Set an adaptive threshold (i.e. a fraction of network traffic volume changes in two consecutive time intervals)
- Network traffic entropy estimation
  - One parameter to compute Shannon entropy, which indicates the network traffic distribution

Packet double counting problem



Node E	
ID	#pkts
	5000
	2000
	200

Node G	
ID	#pkts
	200
	120
	60

Node C	
ID	#pkts
	4000
	200
	120

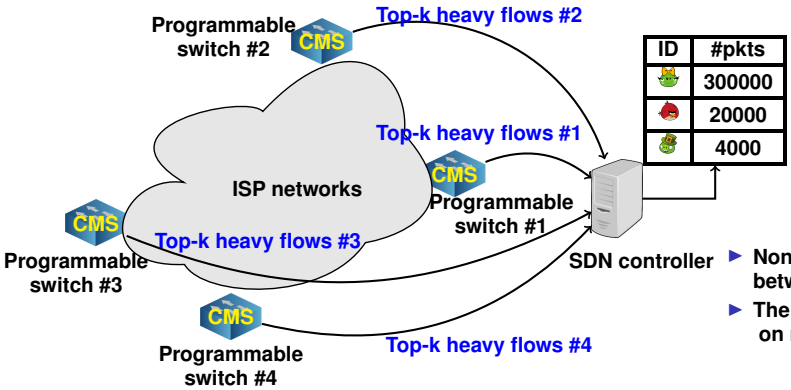
Sum

Controller
Overall number of pkts
5000+2000+ 200+200+120+60+4000+200+120?

The same packet are counted multiple times by the switches along the forwarding path

Ben Basat, Ran, et al. "Network-wide routing-oblivious heavy hitters." Proceedings of the 2018 Symposium on Architectures for Networking and Communications Systems. 2018.

Sample-based solution



Pick the smallest packet count according to flow key and sum them up

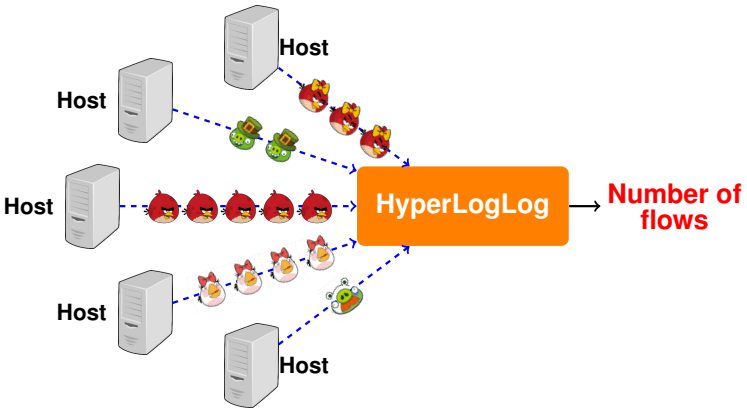
Limitations:

- Non-trivial communication overhead between controller and switches
- The accuracy significantly depends on number of reported flows

---

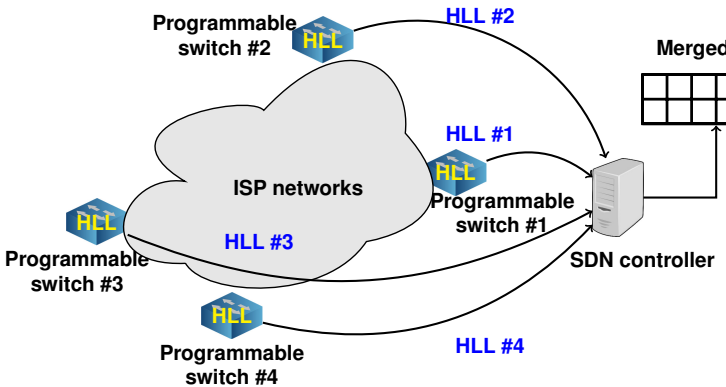
Damu Ding, Marco Savi, Gianni Antichi, and Domenico Siracusa. *An incrementally-deployable P4-enabled architecture for network-wide heavy-hitter detection.* IEEE Transactions on Network and Service Management (TNSM) 17.1 (2020): 75-88.

HyperLogLog (HLL)



- **Fast:** Time complexity is only O(1)
- **Efficient and accurate:** 1.5KB can estimate 10^9 numbers with standard error 2%.
- **Mergeable:** Multiple HLL data structures can be merged into a single one to perform the estimation of union
- **Update operation (i.e. store flow statistics) is implementable in P4**

Flajolet, Philippe, et al. "Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm." Discrete Mathematics and Theoretical Computer Science. Discrete Mathematics and Theoretical Computer Science, 2007.

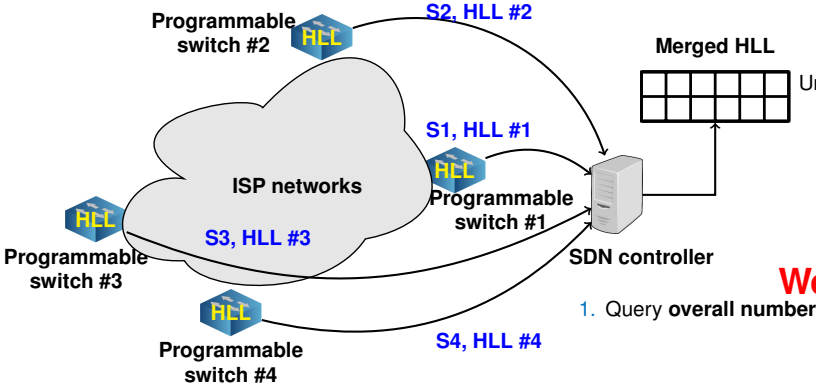


Limitations:

- Not easy (or impossible) to find a unique flow key for each packet
- How to define the flow key by using the fields in the packet header?

Ben Basat, Ran, et al. "Routing Oblivious Measurement Analytics." 2020 IFIP Networking Conference (Networking). IEEE, 2020.

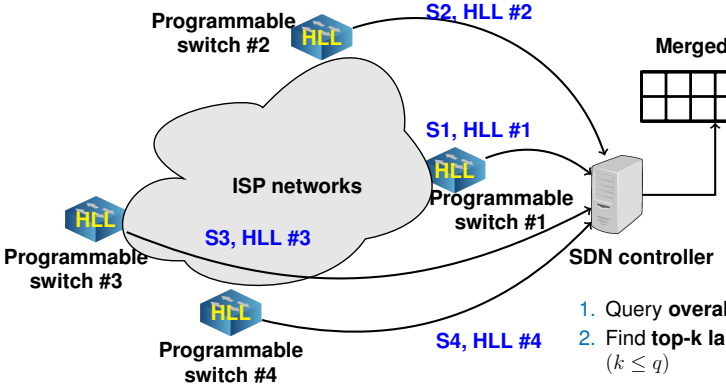
Flow-based traffic volume estimation



Workflow

1. Query **overall number of flows** $\hat{n}_{tot}$ using merged HLL

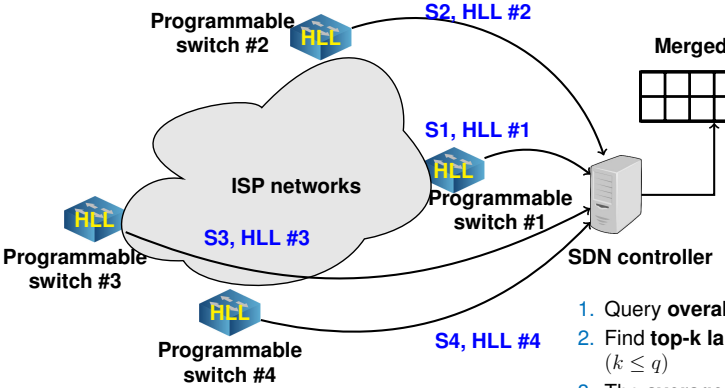
Flow-based traffic volume estimation



Workflow

1. Query **overall number of flows** $\hat{n}_{tot}$ using merged HLL
2. Find **top-k largest** $\hat{n}_i$ that satisfies $\hat{n}_1 \cup \hat{n}_2 \cup \cdots \hat{n}_k = \hat{n}_{tot}$ $(k \leq q)$

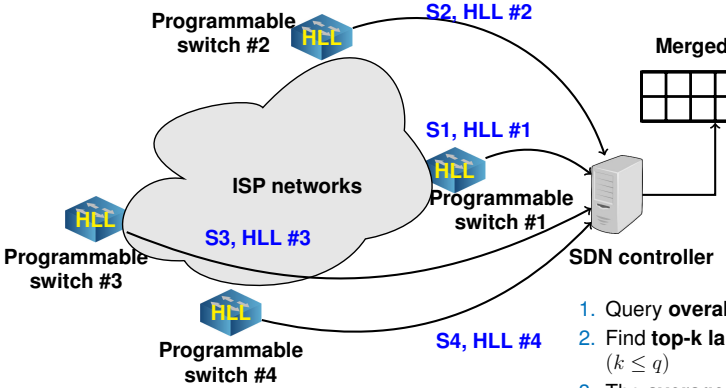
Flow-based traffic volume estimation



Workflow

1. Query **overall number of flows** $\hat{n}_{tot}$ using merged HLL
2. Find **top-k largest** $\hat{n}_i$ that satisfies $\hat{n}_1 \cup \hat{n}_2 \cup \cdots \hat{n}_k = \hat{n}_{tot}$ $(k \leq q)$
3. The **average number of packets per flow** $\hat{R}_i = \frac{|S_i|}{\hat{n}_i} \quad \forall i \in \{1, \ldots, k\}$

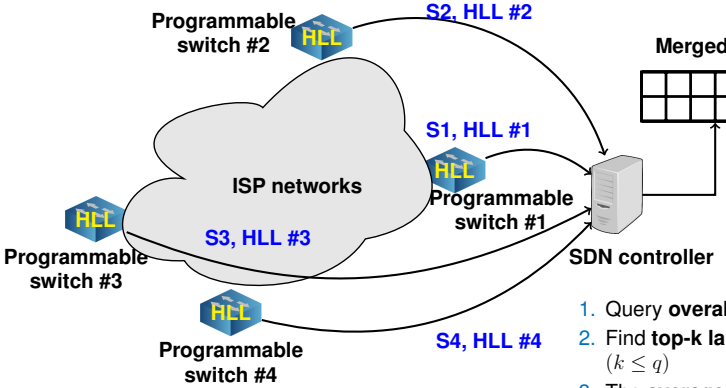
Flow-based traffic volume estimation



Workflow

1. Query **overall number of flows** $\hat{n}_{tot}$ using merged HLL
2. Find **top-k largest** $\hat{n}_i$ that satisfies $\hat{n}_1 \cup \hat{n}_2 \cup \cdots \hat{n}_k = \hat{n}_{tot}$ $(k \leq q)$
3. The **average number of packets per flow** $\hat{R}_i = \frac{|S_i|}{\hat{n}_i} \quad \forall i \in \{1, \ldots, k\}$
4. The **average flow size** $\hat{R}_{tot} = \frac{1}{k}\sum_{i=1}^{k} \hat{R}_i = \frac{1}{k}\sum_{i=1}^{k} \frac{|S_i|}{\hat{n}_i}$

Flow-based traffic volume estimation



$\hat{\boldsymbol{n}}_{\boldsymbol{i}} = Query(HLL\#i)$

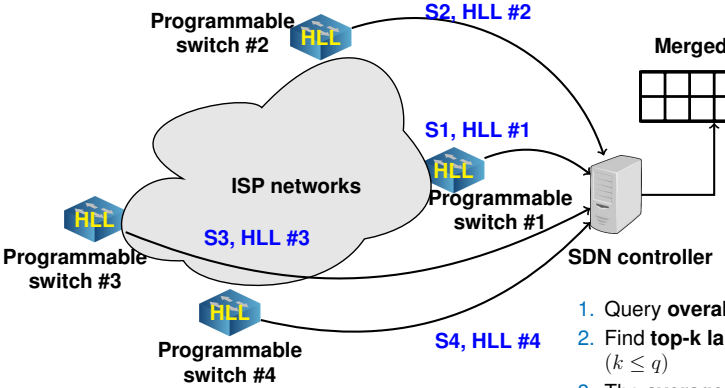
Unique flow key is not needed

$\hat{\boldsymbol{n}}_{\boldsymbol{tot}} = Query(HLL\#1 \cup HLL\#2 \cup \cdots \cup HLL\#q)$

Workflow

1. Query **overall number of flows** $\hat{n}_{tot}$ using merged HLL
2. Find **top-k largest** $\hat{n}_i$ that satisfies $\hat{n}_1 \cup \hat{n}_2 \cup \cdots \hat{n}_k = \hat{n}_{tot}$ $(k \leq q)$
3. The **average number of packets per flow** $\hat{R}_i = \frac{|S_i|}{\hat{n}_i} \quad \forall i \in \{1, \ldots, k\}$
4. The **average flow size** $\hat{R}_{tot} = \frac{1}{k}\sum_{i=1}^{k} \hat{R}_i = \frac{1}{k}\sum_{i=1}^{k} \frac{|S_i|}{\hat{n}_i}$
5. The **traffic volume** $|\hat{S}_{tot}| = \hat{n}_{tot}\hat{R}_{tot} = \frac{\hat{n}_{tot}}{k}\sum_{i=1}^{k} \frac{|S_i|}{\hat{n}_i}$

Flow-based traffic volume estimation



$\hat{\boldsymbol{n}}_{\boldsymbol{i}} = Query(HLL\#i)$

Unique flow key is not needed

$\hat{\boldsymbol{n}}_{\boldsymbol{tot}} = Query(HLL\#1 \cup HLL\#2 \cup \cdots \cup HLL\#q)$

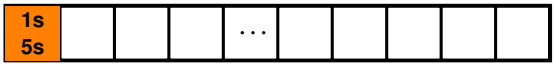
Workflow

1. Query **overall number of flows** $\hat{n}_{tot}$ using merged HLL
2. Find **top-k largest** $\hat{n}_i$ that satisfies $\hat{n}_1 \cup \hat{n}_2 \cup \cdots \hat{n}_k = \hat{n}_{tot}$ $(k \leq q)$
3. The **average number of packets per flow** $\hat{R}_i = \frac{|S_i|}{\hat{n}_i} \quad \forall i \in \{1, \ldots, k\}$
4. The **average flow size** $\hat{R}_{tot} = \frac{1}{k}\sum_{i=1}^{k} \hat{R}_i = \frac{1}{k}\sum_{i=1}^{k} \frac{|S_i|}{\hat{n}_i}$
5. The **traffic volume** $|\hat{S}_{tot}| = \hat{n}_{tot}\hat{R}_{tot} = \frac{\hat{n}_{tot}}{k}\sum_{i=1}^{k} \frac{|S_i|}{\hat{n}_i}$

Our theoretical analysis proves that INVEST has high accuracy on traffic volume estimation

Simulation settings

Normal traffic (CAIDA 2018)



Flow key: {srcIP, dstIP}

45-nodes GEANT[1] and 100-nodes DEFO[2] ISP network

average value of $RE = \left| \frac{|\hat{S}_{tot}| - |S_{tot}|}{|S_{tot}|} \right| \cdot 100\%$ in all the consecutive time intervals

- Each node is assigned in a uniformly random way
- Each packet is forwarded from the ingress point to the egress point following the shortest path

[1] https://sites.uclouvain.be/defo

[2] https://www.geant.org/Networks/Pan-European_network/Pages/GEANT_topology_map.aspx

Comparing to SOTA

Table: **Comparison of INVEST with existing strategies**

Estimation method	Relative error			
	GEANT		DEFO	
	$T_{int} = 1$s	$T_{int} = 5$s	$T_{int} = 1$s	$T_{int} = 5$s
INVEST	2.33%	2.05%	2.44%	2.19%
Sum	333.01%	323.00%	412.79%	412.89%
Sample △ [3]	33.69%	38.78%	25.78%	30.71%
AROMA* [4]	0.48%	3.10%	0.48%	3.10%

△: Feasible but requires large amounts of memory in the switch

*: Requires unique flow key for each packet

[3] Damu Ding, Marco Savi, Gianni Antichi, and Domenico Siracusa. *An incrementally-deployable P4-enabled architecture for network-wide heavy-hitter detection.* IEEE Transactions on Network and Service Management (TNSM) 17.1 (2020): 75-88.

[4] Basat, Ran Ben, et al. "Routing Oblivious Measurement Analytics." 2020 IFIP Networking Conference (Networking). IEEE, 2020.

Sensitivity analysis

flow key type

Flow key	# Distinct flows n_{tot} in T_{int}		Relative error			
			GEANT		DEFO	
	$T_{int} = 1$s	$T_{int} = 5$s	$T_{int} = 1$s	$T_{int} = 5$s	$T_{int} = 1$s	$T_{int} = 5$s
$srcIP$	$\sim$ 27K	$\sim$ 67K	20.15%	26.43%	24.80%	32.35%
$dstIP$	$\sim$ 22K	$\sim$ 58K	23.94%	28.64%	29.13%	34.80%
$\{srcIP, dstIP\}$	$\sim$ 47K	$\sim$ 147K	2.33%	2.05%	2.44%	2.19%
$\{srcIP, dstIP, prot\}$	$\sim$ 47.1K	$\sim$ 147.6K	2.36%	2.73%	2.56%	2.37%
Unique packet id (AROMA)	$\sim$ 450K	$\sim$ 2300K	0.48%	3.10%	0.48%	3.10%

Programmable hardware switch

32x 100Gbps
QSFP ports

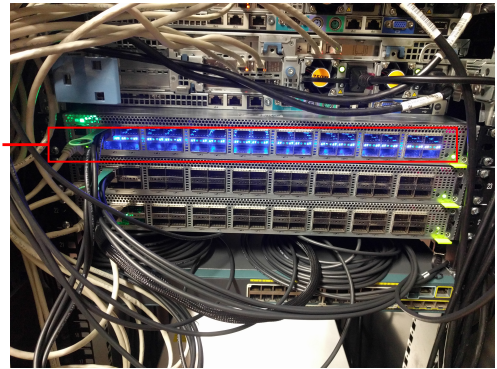


Figure: Edgecore Wedge-100BF-32X switch equipped with Barefoot Tofino ASIC in FBK's lab

Pros:

1. Higher monitoring throughput

Cons:

1. Limited hardware resources
2. Computational constraints

Evaluation of resource usage and processing time

Table: Normalized switch resource usage of INVEST

Strategy	No. stages	SRAM	TCAM	No. ALUs	PHV size	Additional proc. time w.r.t. simple forwarding
Simple forwarding	16.67%	2.5%	8.33%	4.17%	7.30%	-
INVEST_Update + Simple forwarding	41.67%	3.23%	9.03%	8.33%	7.68%	45ns

Conclusion

- We designed INVEST, a flow-based method that exploits programmable data planes to estimate the traffic volume while solving the double counting problem
- We theoretically analyzed INVEST and experimentally evaluated it with simulations
- We implemented the HyperLogLog Update procedure for flow cardinality estimation in P4
- We developed a prototype of INVEST, installed it in a carrier-grade programmable switch with Tofino Application Specific Integrated Circuit (ASIC), and evaluated its performance in a physical testbed

Thank you!