Enabling Event Triggered Monitoring of Traffic Clusters

Gianni Antichi

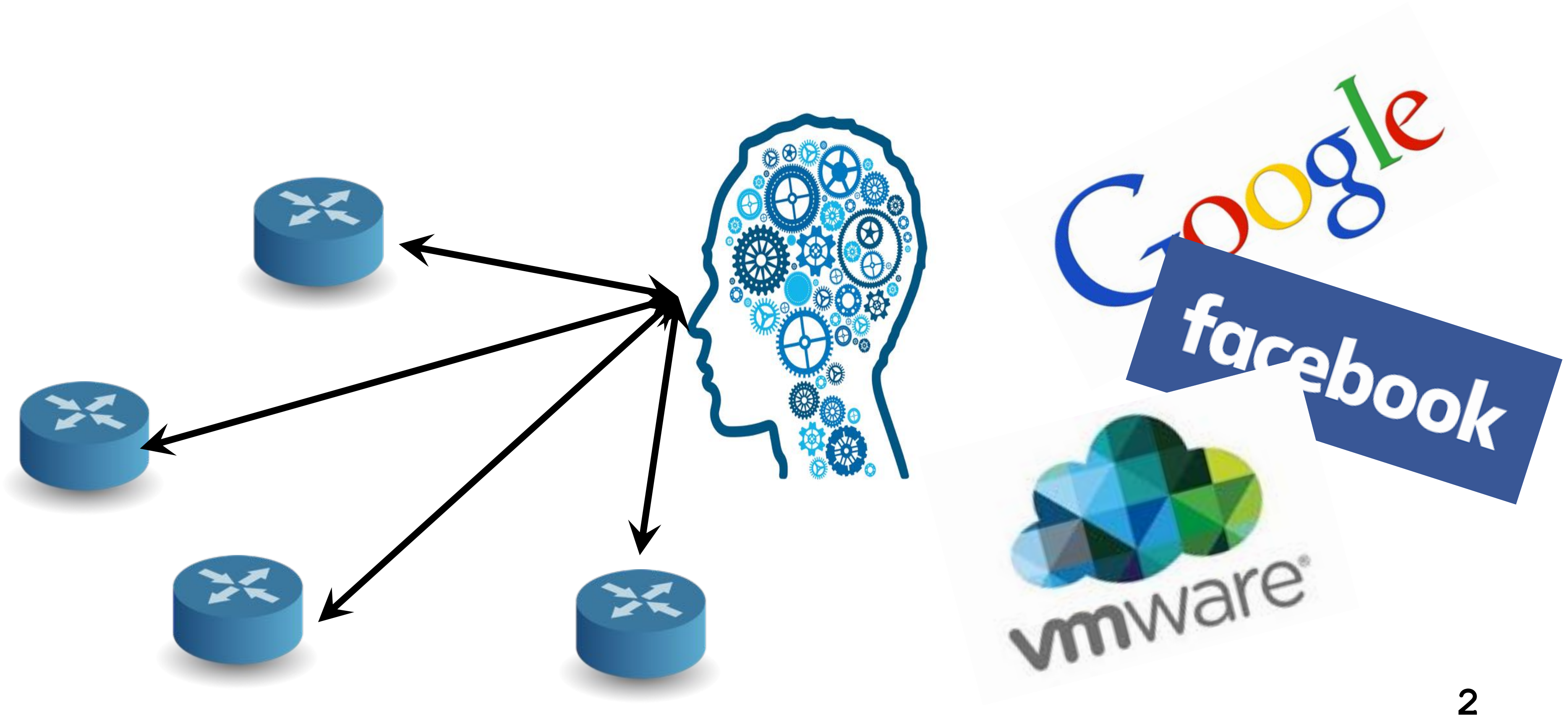
gianni.antichi@cl.cam.ac.uk

University of Cambridge

in collaboration with:

J.Kucera, D.A.Popescu, J.Korenek and A.W.Moore

Centralised Control

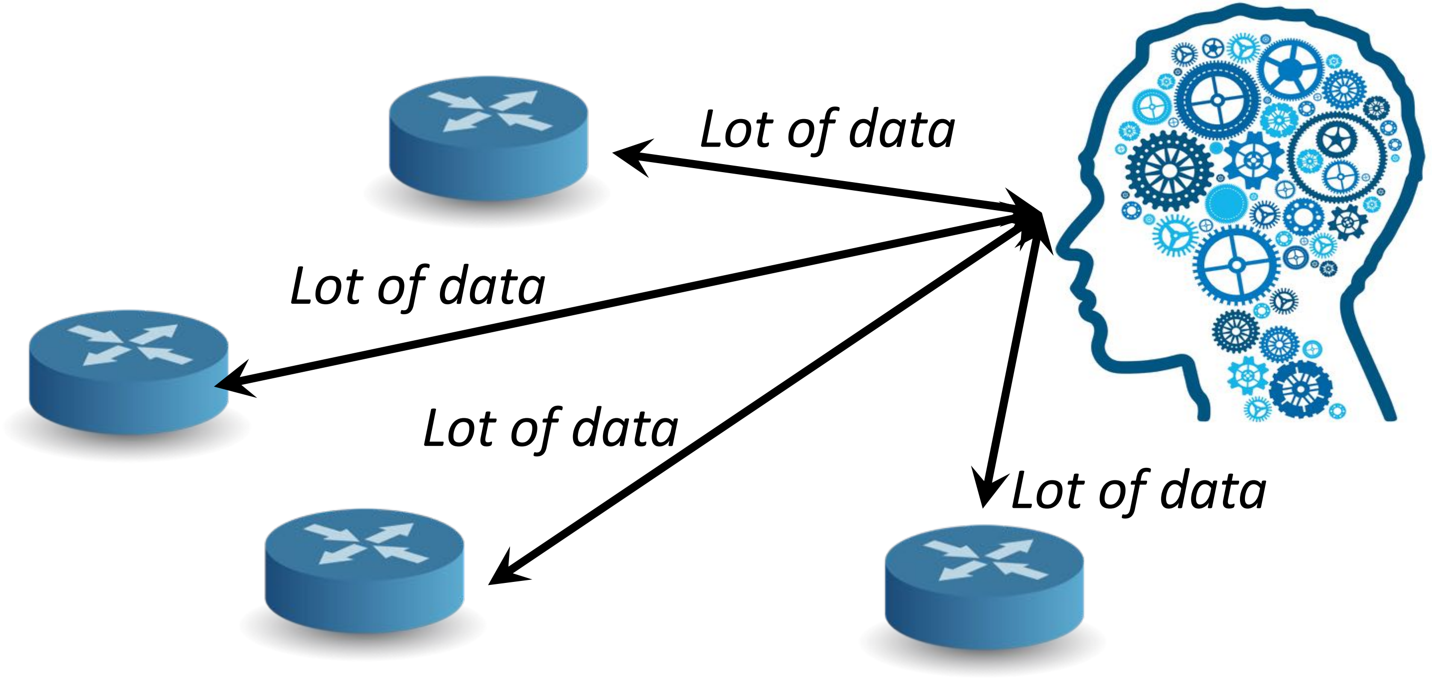


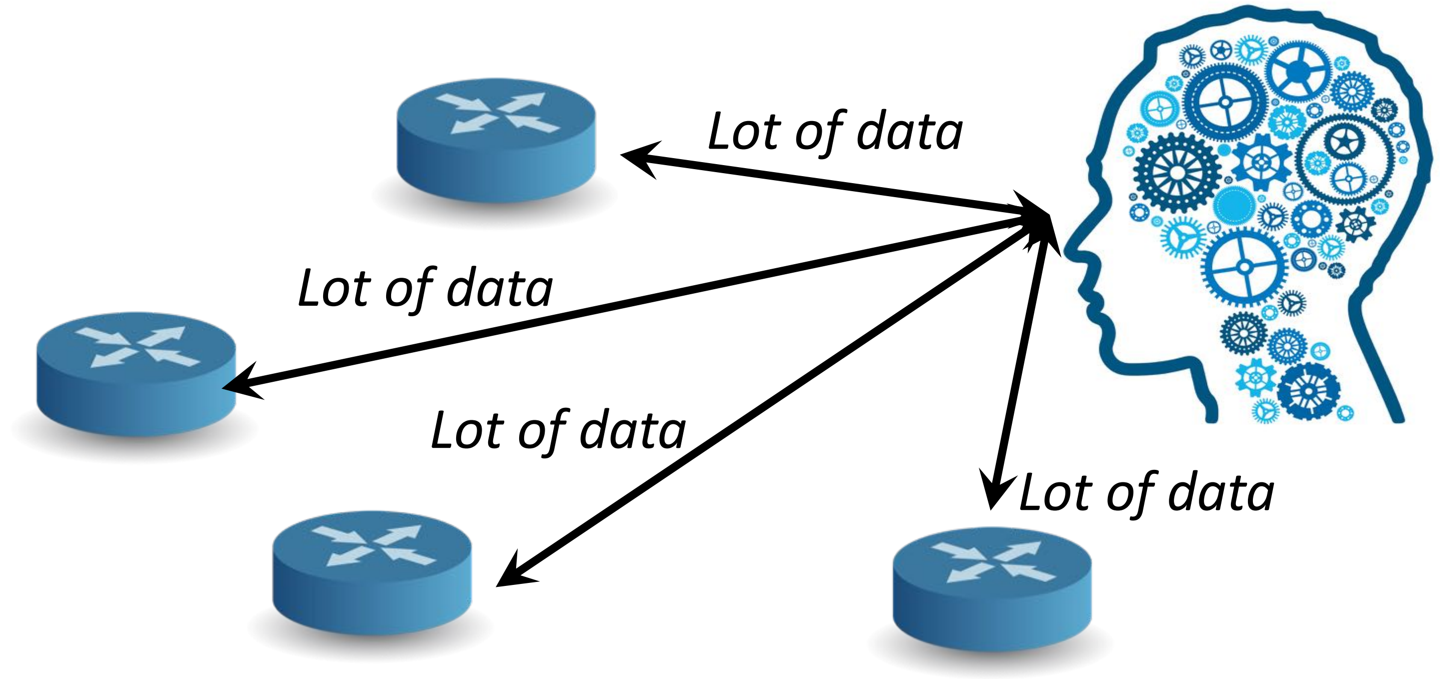
Centralised Control is your friend.

- Easy to change the behavior of a network
- You can easily couple network monitoring with network function

Centralised Control is your friend.

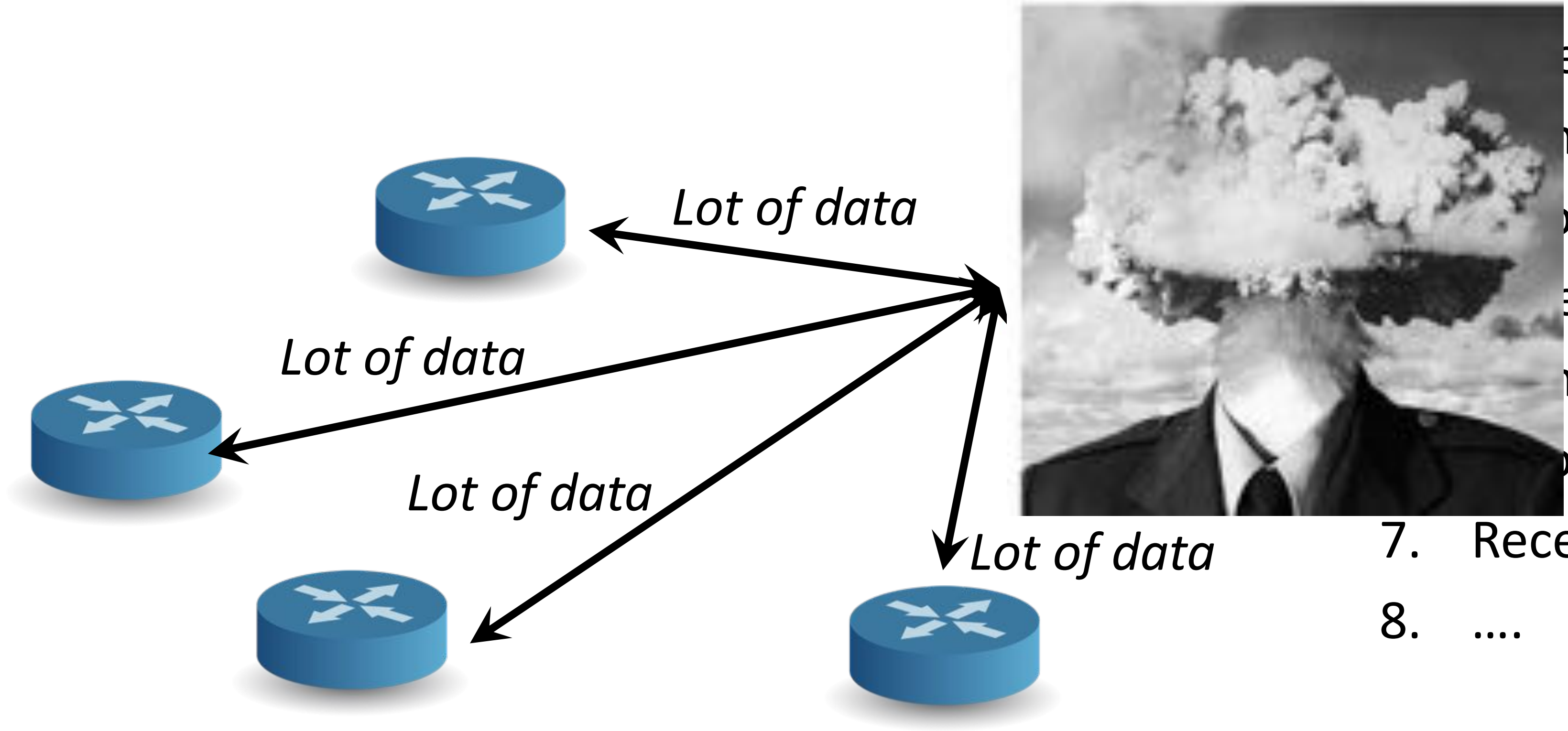
- Easy to change the behavior of a network
- You can easily couple network monitoring with network function


Lot of data
Lot of data
Lot of data
Lot of data



1. Receive
2. Compute
3. Enforce
4. Receive
5. Compute
6. Enforce
7. Receive
8. ….

The main OpenFlow monitoring mechanism exposes the per-port and per-flow counters available in the switches.



eive

npute

rce

eive

npute

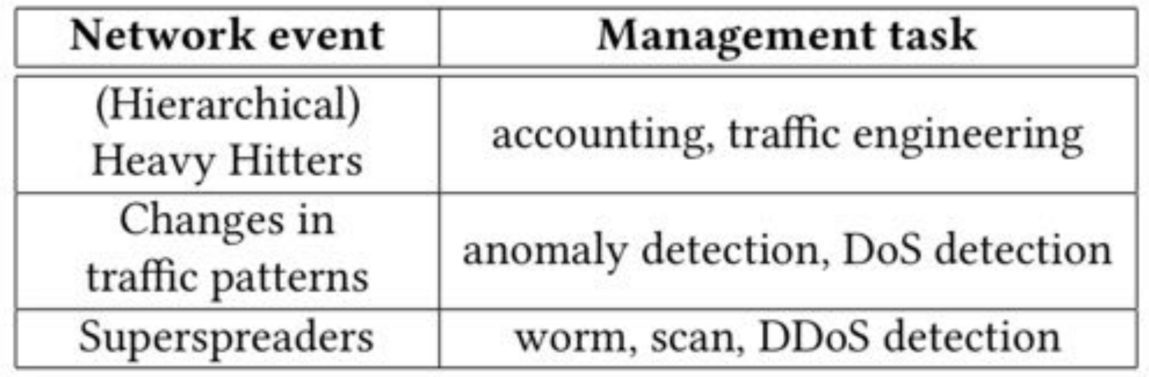
rce

7. Receive
8. ....

The main OpenFlow monitoring mechanism exposes the per-port and per-flow counters available in the switches.

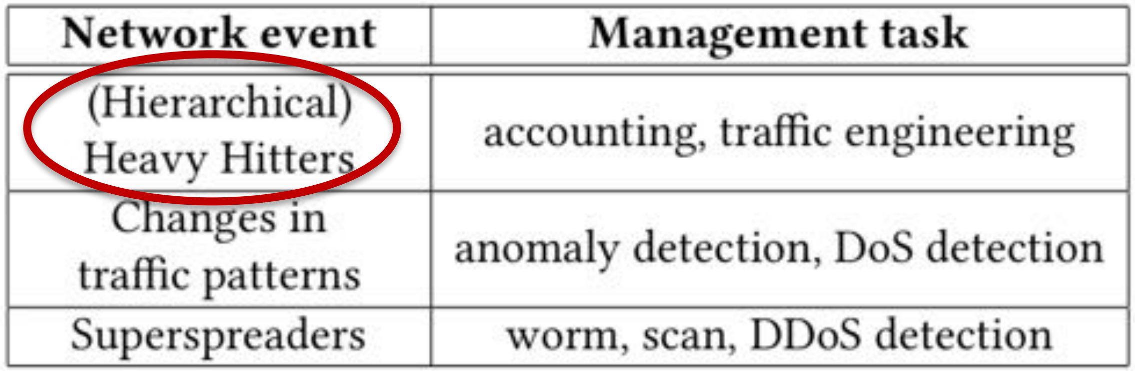
Can we send monitoring information to the controller only when it makes sense?

Let’s focus on **high-volume traffic clusters**



Network event	Management task
(Hierarchical) Heavy Hitters	accounting, traffic engineering
Changes in traffic patterns	anomaly detection, DoS detection
Superspreaders	worm, scan, DDoS detection

A common ground first

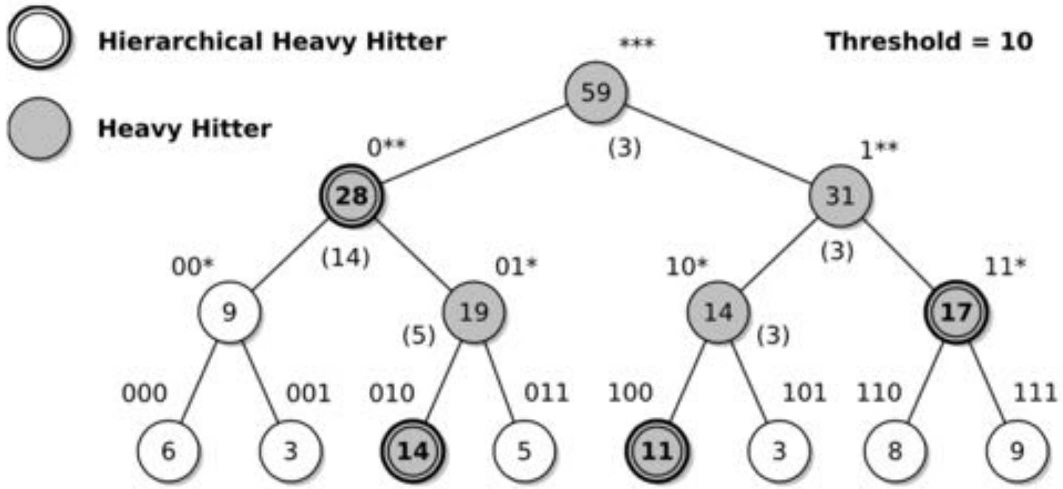


Network event	Management task
(Hierarchical) Heavy Hitters	accounting, traffic engineering
Changes in traffic patterns	anomaly detection, DoS detection
Superspreaders	worm, scan, DDoS detection

(Hierarchical) Heavy Hitters

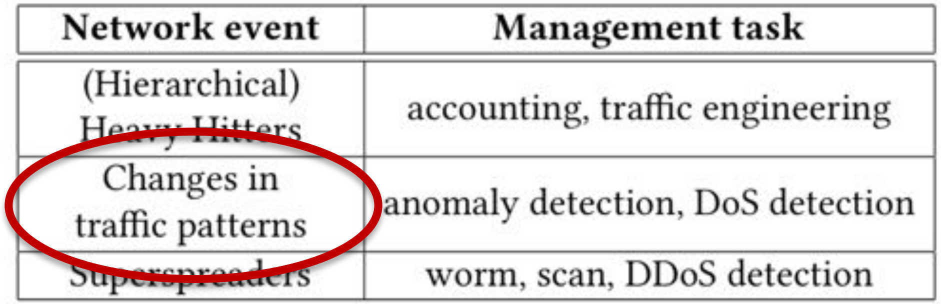
Heavy Hitters (HH): a prefix that contribute with a traffic volume larger than a given threshold T during a fixed time interval t .

Hierarchical Heavy Hitter (HHH): a prefix that exceeds a threshold T after excluding the contribution of all its HHH descendants.



Changes in traffic patterns

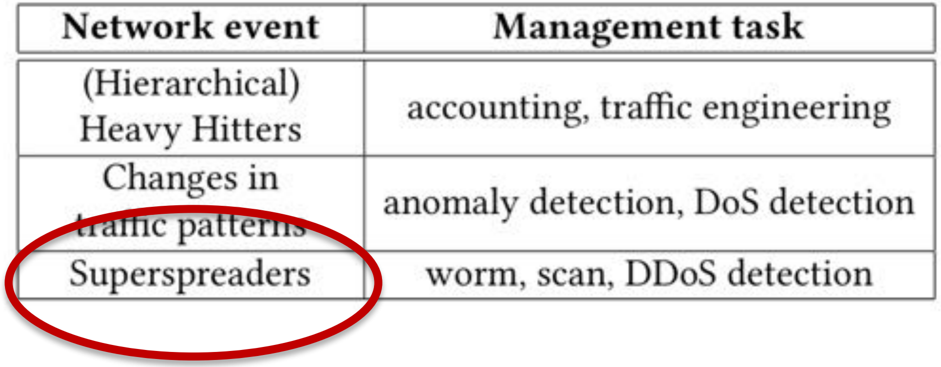
Identifying the flows that contribute the most for the changes in the traffic patterns over two consecutive time intervals.



Network event	Management task
(Hierarchical) Heavy Hitters	accounting, traffic engineering
Changes in traffic patterns	anomaly detection, DoS detection
Superspreaders	worm, scan, DDoS detection

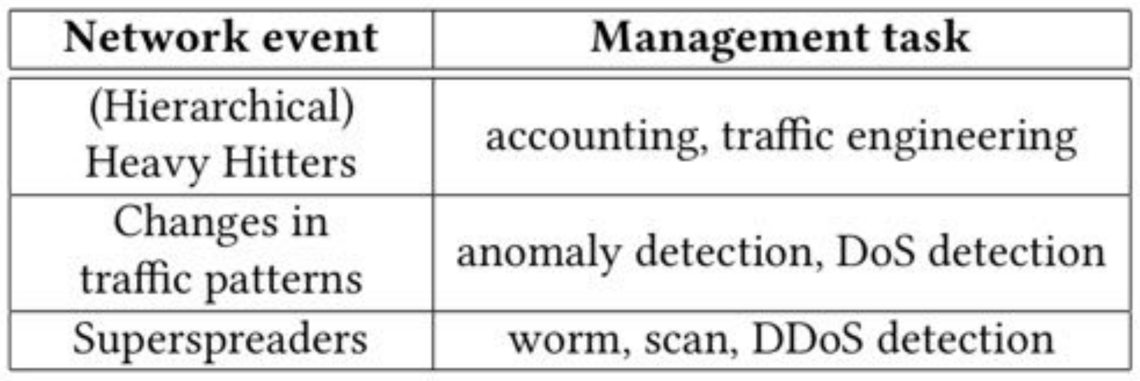
Superspreaders

A host that contacts at least a given number of distinct destinations over a short time period.



Network event	Management task
(Hierarchical) Heavy Hitters	accounting, traffic engineering
Changes in traffic patterns	anomaly detection, DoS detection
Superspreaders	worm, scan, DDoS detection

All those network events can be seen as a **<u>traffic cluster detection problem</u>**



Network event	Management task
(Hierarchical) Heavy Hitters	accounting, traffic engineering
Changes in traffic patterns	anomaly detection, DoS detection
Superspreaders	worm, scan, DDoS detection

HHH and **change detection**: packets or bytes per second.

Superspreaders: flows per second.

Can we leverage **dataplane programmability** to **assist** in the detection of those events?

Can we leverage **dataplane programmability** to **assist** in the detection of those events?

Can we leverage **dataplane programmability** to **assist** in the detection of those events?

HashPipe [1]: calculate within the dataplane the top-k heavy hitters at fixed intervals.

[1] Heavy-Hitter Detection Entirely in the Data Plane, V. sivaraman, S. Narayana, O. Rottenstreich, S. Muthukrishnan, J. Rexford. In ACM SOSR 2017.

Can we leverage **dataplane programmability** to **assist** in the detection of those events?

HashPipe [1]: calculate within the dataplane the top-k heavy hitters at fixed intervals.

Focus on heavy hitter only.

[1] Heavy-Hitter Detection Entirely in the Data Plane, V. sivaraman, S. Narayana, O. Rottenstreich, S. Muthukrishnan, J. Rexford. In ACM SOSR 2017.

Can we leverage **dataplane programmability** to **assist** in the detection of those events?

HashPipe [1]: calculate within the dataplane the top-k heavy hitters at fixed intervals.

Focus on heavy hitter only.

Univmon [2]: assist the controller by exporting smart representation of aggregated statistics.

[1] Heavy-Hitter Detection Entirely in the Data Plane, V. sivaraman, S. Narayana, O. Rottenstreich, S. Muthukrishnan, J. Rexford. In ACM SOSR 2017.
[2] One Sketch to Rule Them All: Rethinking Network Flow Monitoring with UnivMon, Z. Liu, A. Manousis, G. Vorsanger, V. Sekar, V. Braverman. In ACM SIGCOMM 2016.

Can we leverage **dataplane programmability** to **assist** in the detection of those events?

HashPipe [1]: calculate within the dataplane the top-k heavy hitters at fixed intervals.

Focus on heavy hitter only.

Univmon [2]: assist the controller by exporting smart representation of aggregated statistics.
The actual detection is performed in the control plane.

[1] Heavy-Hitter Detection Entirely in the Data Plane, V. sivaraman, S. Narayana, O. Rottenstreich, S. Muthukrishnan, J. Rexford. In ACM SOSR 2017.
[2] One Sketch to Rule Them All: Rethinking Network Flow Monitoring with UnivMon, Z. Liu, A. Manousis, G. Vorsanger, V. Sekar, V. Braverman. In ACM SIGCOMM 2016.

Can we leverage **dataplane programmability** to **<u>assist</u>** in the detection of those events?

HashPipe [1]: calculate within the dataplane the top-k heavy hitters at fixed intervals.

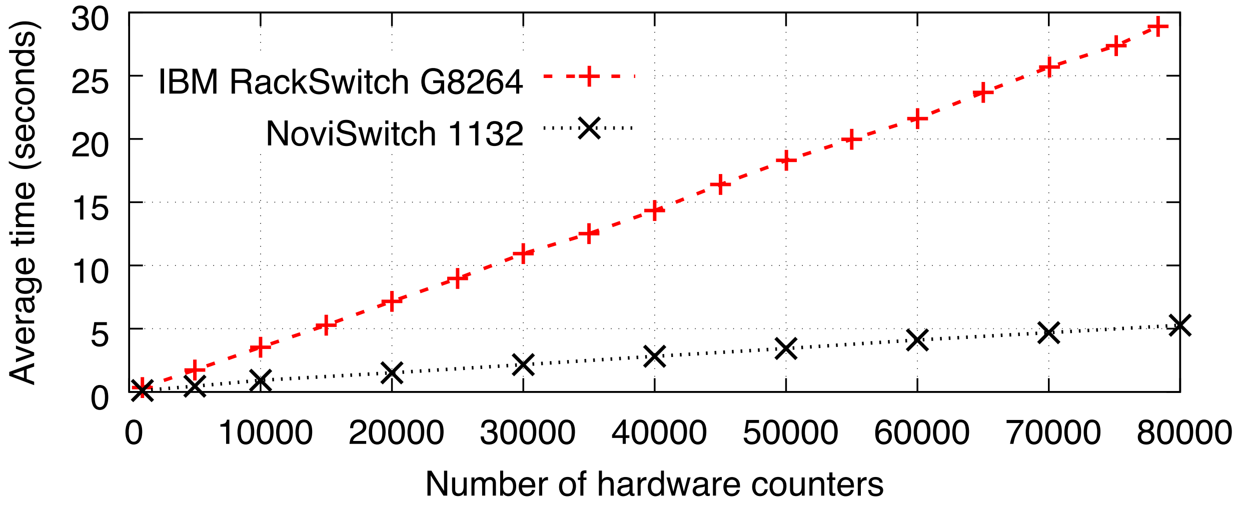
Focus on heavy hitter only.

Univmon [2]: assist the controller by exporting smart representation of aggregated statistics.

The actual detection is performed in the control plane.

Wait a minute. Is this a problem?

Retrieving a large number of counters from hardware is **time consuming**!!!



Note: probabilistic data structures (i.e., sketches) require large amount of counters to lower false positive ratio.

Can we leverage **dataplane programmability** to **assist** in the detection of those events?

Can we leverage **dataplane programmability** to **~~assist~~** in the detection of those events?

Can we leverage **dataplane programmability** to **enable in-network** detection of those events?

Because if you have in-network detection..

As soon as you detect you can take pre-defined actions.

Good for network reactiveness.

Because if you have in-network detection..

As soon as you detect you can take pre-defined actions.

Good for network reactiveness.

You can directly export the detection result to the control plane.

Control plane does not have to receive lot of data and understand what is going on.

Because if you have in-network detection..

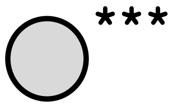
Elastic Trie in a nutshell

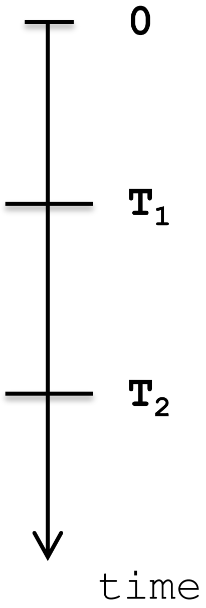
- Prefix tree that **grows** or **collapses:** focus on who account for a *large* share of the traffic.

- **Starting condition:** a single node corresponding with zero-length prefix *.

Elastic Trie in action

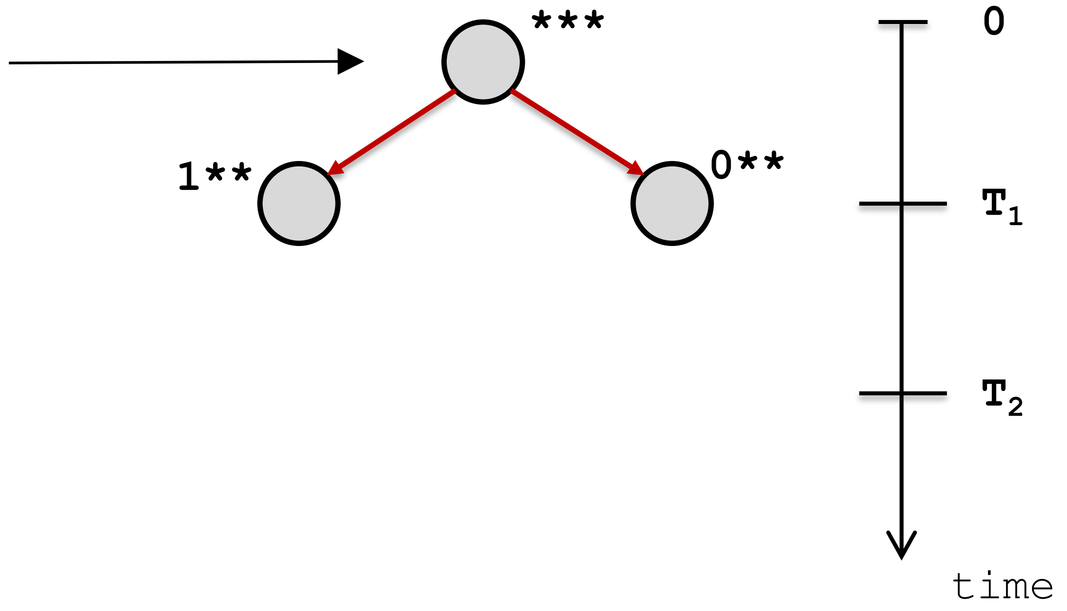
```
- counter-Left
- timestamp
- counter-Right
```





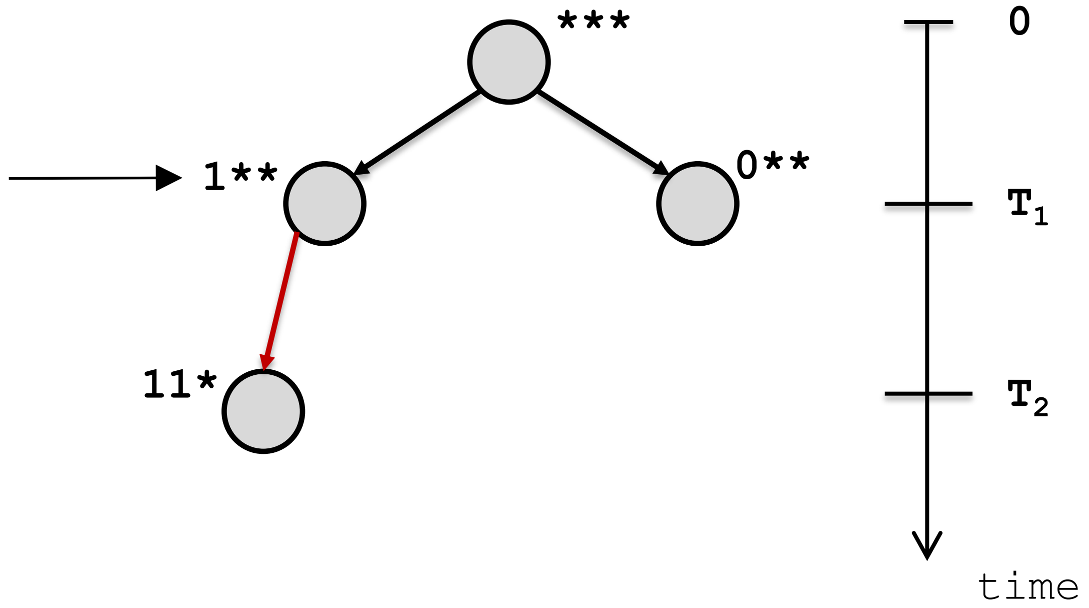
Elastic Trie in action

Both counterL and counterR exceed threshold in T_1

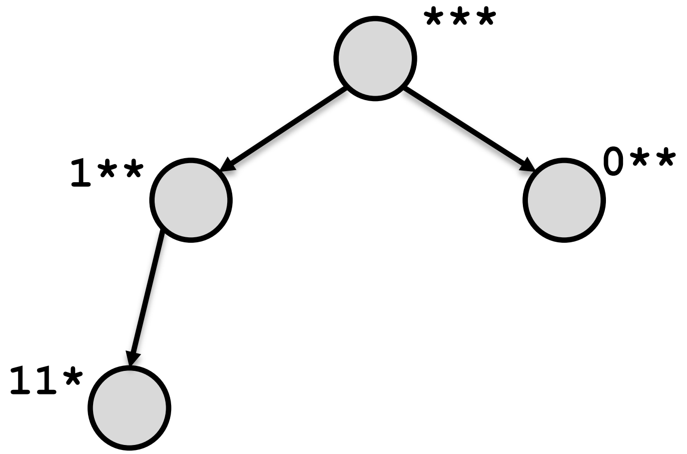


Elastic Trie in action

counterL exceeds threshold in T_2



Elastic Trie in action

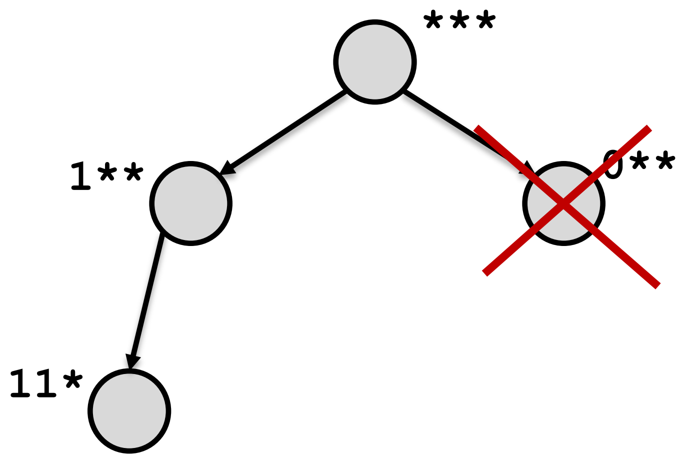


Packet timestamp

>>

Node timestamp

Elastic Trie in action

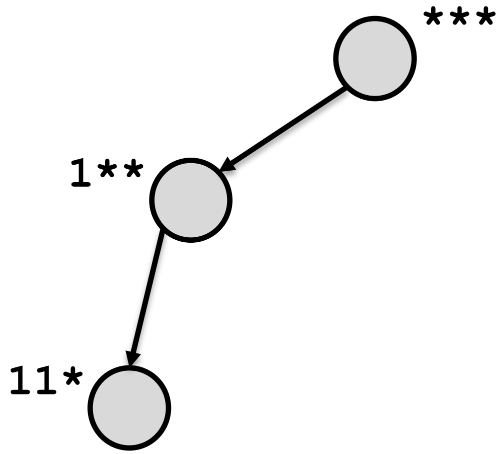


Packet timestamp

>>

Node timestamp

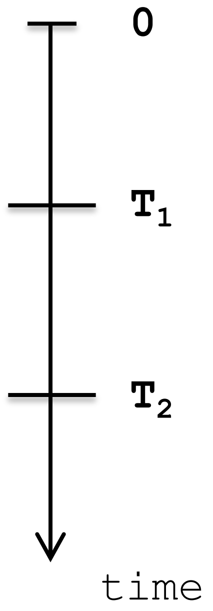
Elastic Trie in action



Elastic Trie implications

- The dataplane iteratively refine the responsible IP prefixes:

the controller can receive a **flexible granularity information**.



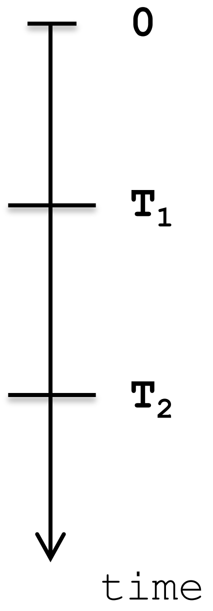
Elastic Trie implications

- The dataplane iteratively refine the responsible IP prefixes:

the controller can receive a **flexible granularity information**.

- Each prefix tree layer can have a **different timeout**:

trade-off between trie building process and memory consumption.



Elastic Trie implications

- The dataplane iteratively refine the responsible IP prefixes:

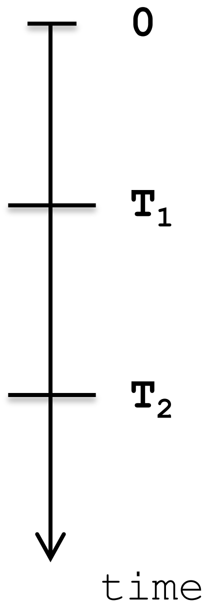
the controller can receive a **flexible granularity information**.

- Each prefix tree layer can have a **different timeout**:

trade-off between trie building process and memory consumption.

- By looking at the **growing rate** of the trie it is possible to:

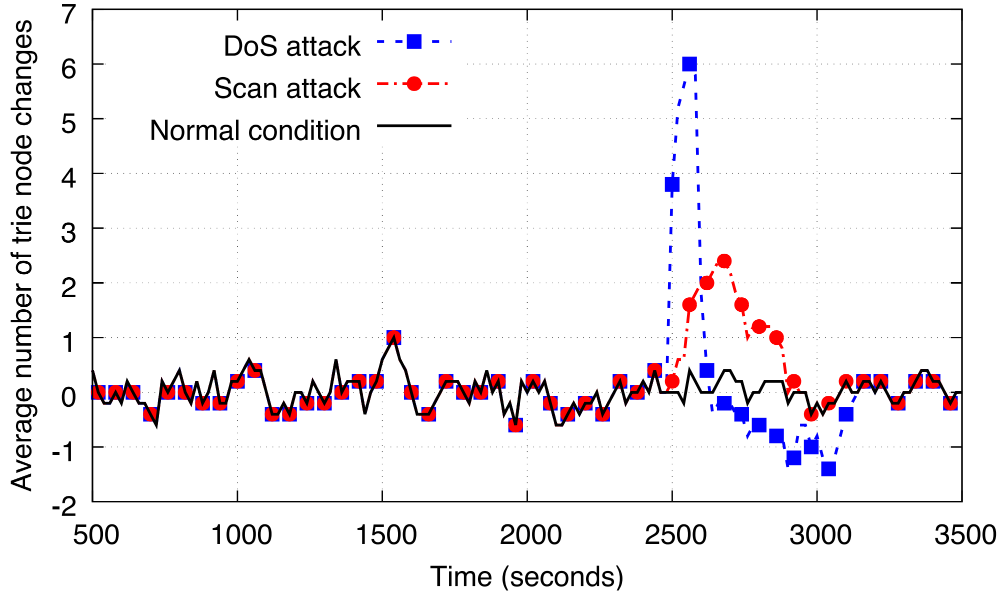
identify changes in the traffic patterns.



Elastic Trie in action

- We prototyped Elastic Trie in P4 and evaluated with packet traces from an ISP backbone and datacenter
- **Hierarchical Heavy Hitter:** 95% precision with 8KB of memory
- **Superspreader:** 95% precision with 80KB of memory

Elastic Trie in action



Changes can be spotted!!!

Conclusions

- Elastic Trie enables **in-network detection** of traffic aggregates
- **Push-based monitoring** approach
- Suitable for **HH**, **HHH**, **Superspreader** and **Change detection**.
- Low memory footprint!
- Suitable for match-action architectures