Los Alamos

NATIONAL LABORATORY

EST. 1943

NNSA

National Nuclear Security Administration

LA-UR-21-28051

Network Monitoring and Analytics with sFlow

Conner Whitfield
HPC-SYS CSP Team

August 12, 2021

Overview

- Production monitoring needs
- What is sFlow?
- sFlow virtual environment testing
- Initial deployment process
- Challenges and deployment changes
- Discoveries in the network
- Future work

Production Monitoring Needs

Needs:

- Real-time transparency into switches' network and system metrics.
- Monitoring approach that can be easily deployed on a large scale, across multiple switch brands.
- Monitoring software that can replace custom monitoring scripts.
- Monitoring software that integrates with existing monitoring tools.

Solution:

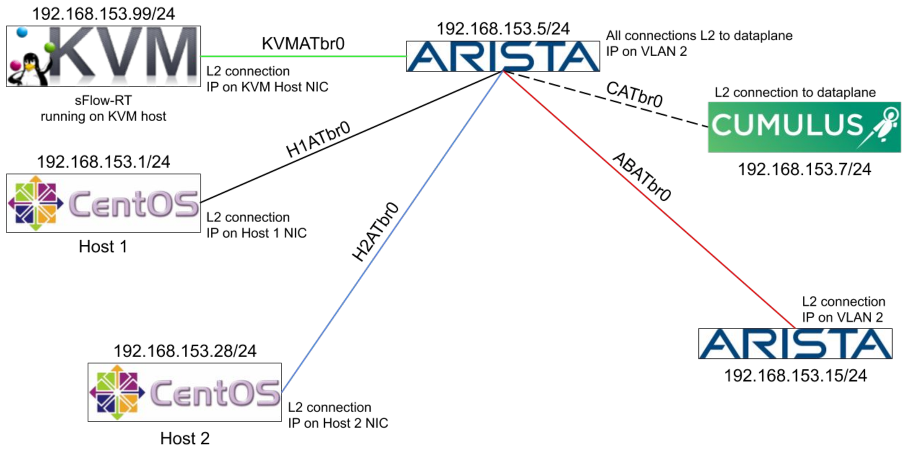
- Utilize sFlow, as it meets all of the above criteria.
    - Real-time monitoring of network and system performance.
    - Widely used standard, included on Arista and Cumulus switches.
    - Default feature set meets monitoring requirements, with possibility of creating additional custom metrics.
    - Compatible with Telegraf, Prometheus, and Splunk.

What is sFlow?

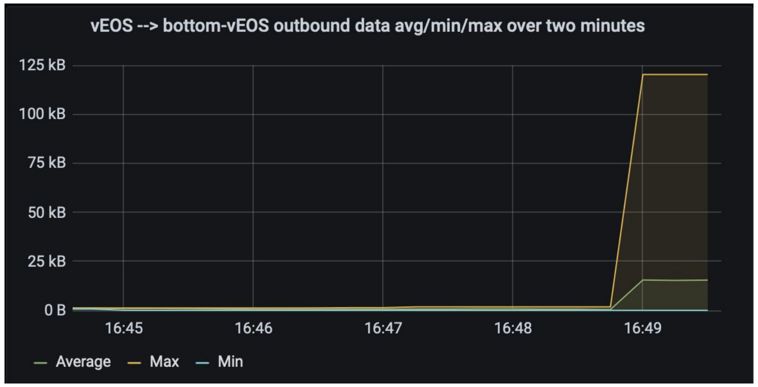
- Monitoring software that collects system and network data in real-time, for analysis in a metrics dashboard.
- Metrics examples:
  - CPU utilization
  - Memory utilization
  - Interface I/O rates
- sFlow generates datagrams by sampling packets and system data on switches, then sends datagrams to a central server.

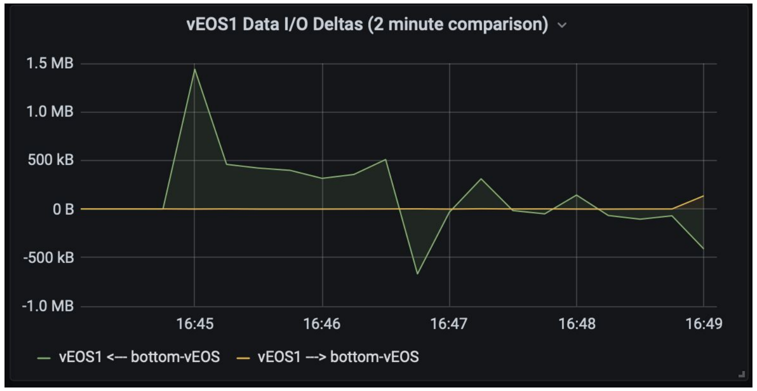
sFlow Virtual Environment Testing

- sFlow tested on a KVM-based virtual network.
- Demo dashboard built in Grafana to highlight key metrics.

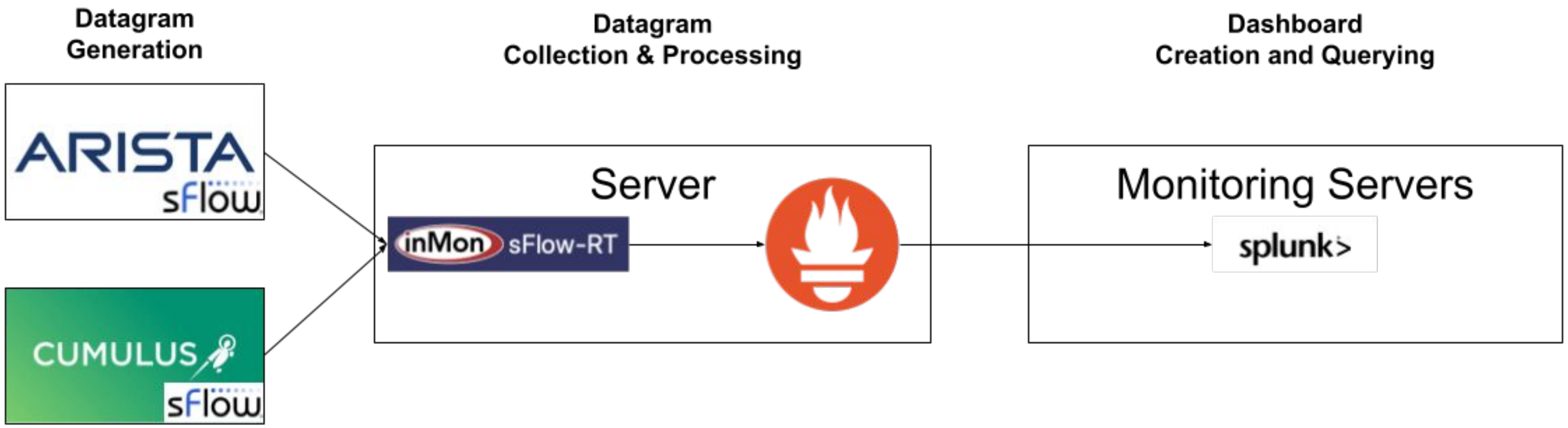


192.168.153.99/24
KVM
sFlow-RT running on KVM host
KVMATbr0
L2 connection IP on KVM Host NIC
192.168.153.5/24
ARISTA
All connections L2 to dataplane IP on VLAN 2
CATbr0
L2 connection to dataplane
CUMULUS
192.168.153.7/24
H1ATbr0
192.168.153.1/24
CentOS
Host 1
L2 connection IP on Host 1 NIC
ABATbr0
H2ATbr0
L2 connection IP on VLAN 2
ARISTA
192.168.153.15/24
192.168.153.28/24
CentOS
Host 2
L2 connection IP on Host 2 NIC



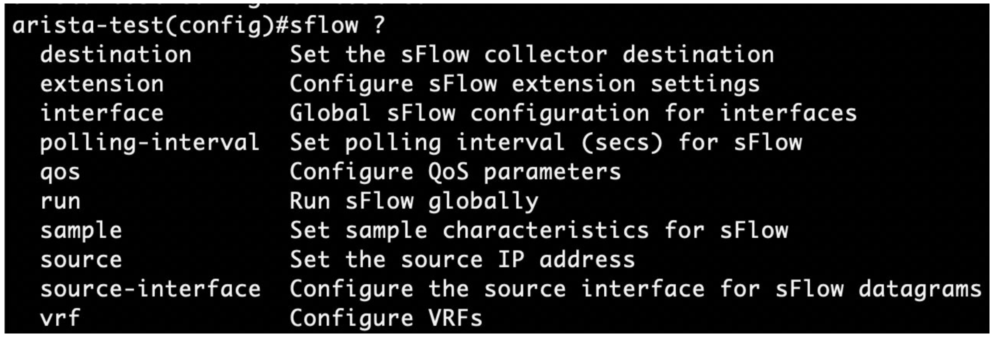


First Deployment - Overview



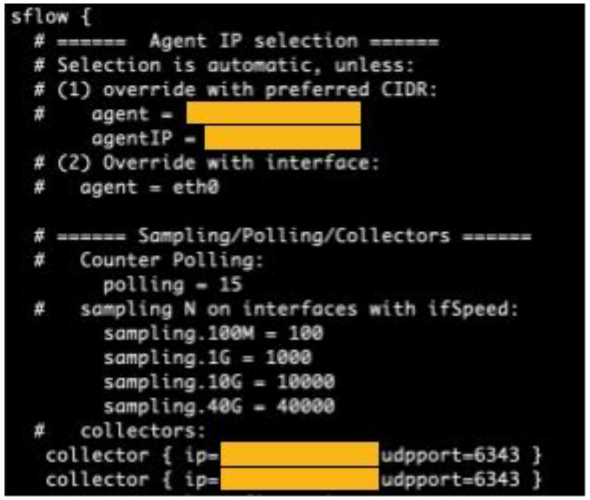
sFlow Deployment Process - Switches

- Restricts sFlow configuration to specific commands, limiting configuration possibilities.
- Datagram destinations, source interface, VRF configuration, sample rate, polling interval, per-interface enabling, and other details.



```
arista-test(config)#sflow ?
  destination        Set the sFlow collector destination
  extension          Configure sFlow extension settings
  interface          Global sFlow configuration for interfaces
  polling-interval   Set polling interval (secs) for sFlow
  qos                Configure QoS parameters
  run                Run sFlow globally
  sample             Set sample characteristics for sFlow
  source             Set the source IP address
  source-interface   Configure the source interface for sFlow datagrams
  vrf                Configure VRFs
```

- Allows full configuration of sFlow, via editable configuration files.
- Includes Arista's configuration capabilities, the ability to manually set source IP address, and more.



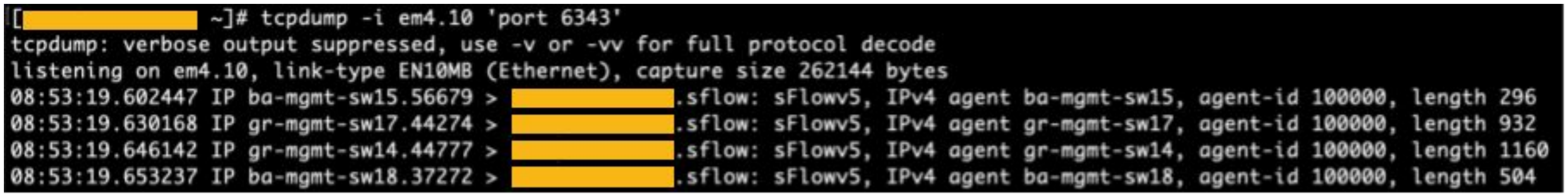
```
sflow {
  # ====== Agent IP selection ======
  # Selection is automatic, unless:
  # (1) override with preferred CIDR:
  #      agent = [redacted]
         agentIP = [redacted]
  # (2) Override with interface:
  #    agent = eth0

  # ====== Sampling/Polling/Collectors ======
  #    Counter Polling:
         polling = 15
  #    sampling N on interfaces with ifSpeed:
         sampling.100M = 100
         sampling.1G = 1000
         sampling.10G = 10000
         sampling.40G = 40000
  #    collectors:
    collector { ip=[redacted] udpport=6343 }
    collector { ip=[redacted] udpport=6343 }
```

First Deployment - Switch Configuration

- Switches on both platforms were configured with:
  - Source interface set to each switch's IP address in the same subnet as the central server.
  - Datagram destination set to the central server's IP address.
  - 15 second polling interval.
  - Interface packet sample rates set to default values.
- sFlow datagrams received on central server via port 6343.

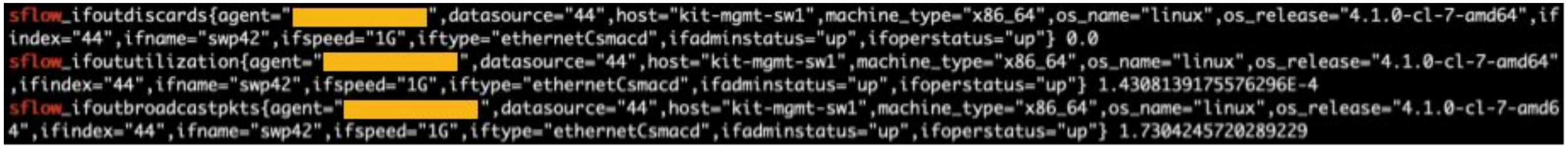
Interface speed	Packet sampling rate
100M	100
1G	1,000
10G	10,000
40G	40,000



```
[[redacted] ~]# tcpdump -i em4.10 'port 6343'
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on em4.10, link-type EN10MB (Ethernet), capture size 262144 bytes
08:53:19.602447 IP ba-mgmt-sw15.56679 > [redacted].sflow: sFlowv5, IPv4 agent ba-mgmt-sw15, agent-id 100000, length 296
08:53:19.630168 IP gr-mgmt-sw17.44274 > [redacted].sflow: sFlowv5, IPv4 agent gr-mgmt-sw17, agent-id 100000, length 932
08:53:19.646142 IP gr-mgmt-sw14.44777 > [redacted].sflow: sFlowv5, IPv4 agent gr-mgmt-sw14, agent-id 100000, length 1160
08:53:19.653237 IP ba-mgmt-sw18.37272 > [redacted].sflow: sFlowv5, IPv4 agent ba-mgmt-sw18, agent-id 100000, length 504
```

First Deployment - Server

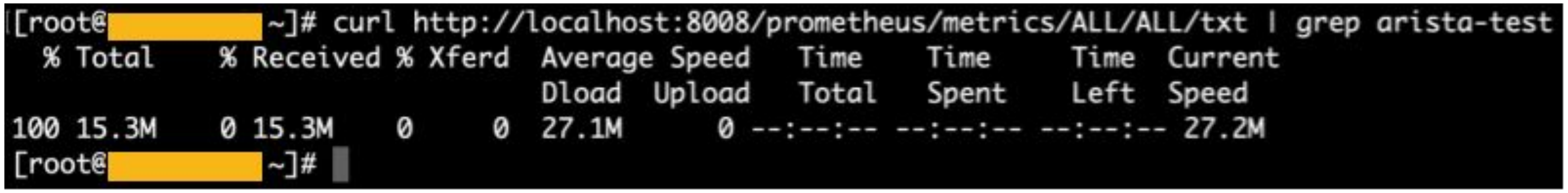
- Datagrams collected through sFlow-RT.
  - Required for querying and processing sFlow data.
  - Outputs Prometheus-formatted data.
  - Provides implementation of custom metrics.
  - Exports server’s system information data for querying in Splunk.
- Prometheus-formatted sFlow data fed into Splunk via port 9090 on central server.
- This first deployment faced two key challenges:
  - Poor dashboard readability due to lack of hostnames on Arista switches.
  - CPU and memory information not provided by Arista’s sFlow implementation.



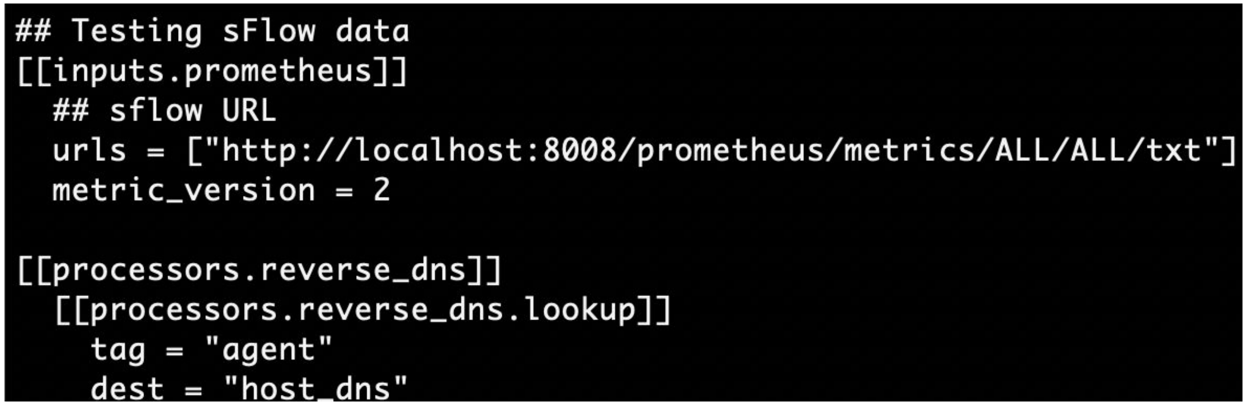
```
sflow_ifoutdiscards{agent="[redacted]",datasource="44",host="kit-mgmt-sw1",machine_type="x86_64",os_name="linux",os_release="4.1.0-cl-7-amd64",if
index="44",ifname="swp42",ifspeed="1G",iftype="ethernetCsmacd",ifadminstatus="up",ifoperstatus="up"} 0.0
sflow_ifoututilization{agent="[redacted]",datasource="44",host="kit-mgmt-sw1",machine_type="x86_64",os_name="linux",os_release="4.1.0-cl-7-amd64"
,ifindex="44",ifname="swp42",ifspeed="1G",iftype="ethernetCsmacd",ifadminstatus="up",ifoperstatus="up"} 1.4308139175576296E-4
sflow_ifoutbroadcastpkts{agent="[redacted]",datasource="44",host="kit-mgmt-sw1",machine_type="x86_64",os_name="linux",os_release="4.1.0-cl-7-amd6
4",ifindex="44",ifname="swp42",ifspeed="1G",iftype="ethernetCsmacd",ifadminstatus="up",ifoperstatus="up"} 1.7304245720289229
```

Challenge 1: Missing Hostnames in Arista sFlow Data

- Arista sFlow datagrams do not include the hostname as a field.
- Creates poor search readability and search simplicity.
- Solved by processing Prometheus-formatted sFlow data through Telegraf and performing a reverse DNS lookup on all data that does not have a hostname.
- Dashboard now more readable, and searches are consistent.



```
[root@[redacted] ~]# curl http://localhost:8008/prometheus/metrics/ALL/ALL/txt | grep arista-test
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 15.3M    0 15.3M    0     0  27.1M      0 --:--:-- --:--:-- --:--:-- 27.2M
[root@[redacted] ~]#
```

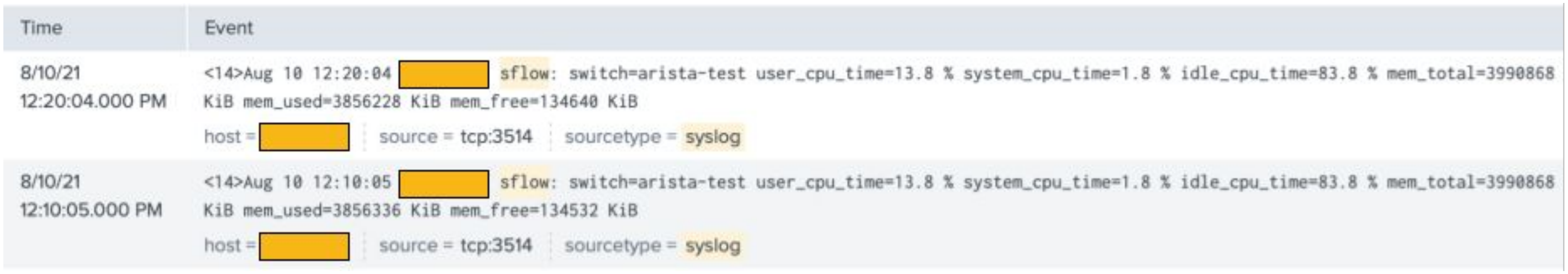


```
## Testing sFlow data
[[inputs.prometheus]]
  ## sflow URL
  urls = ["http://localhost:8008/prometheus/metrics/ALL/ALL/txt"]
  metric_version = 2

[[processors.reverse_dns]]
  [[processors.reverse_dns.lookup]]
    tag = "agent"
    dest = "host_dns"
```

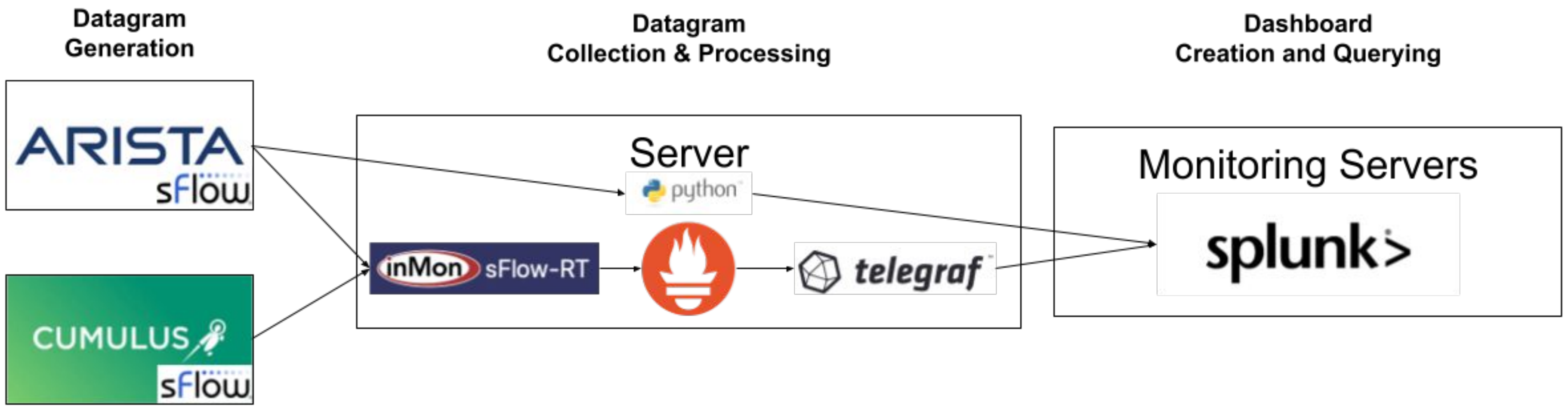
Challenge 2: Missing CPU and Memory Information on Arista sFlow Datagrams

- CPU and memory information not provided by Arista's sFlow implementation, while Cumulus switches provide this data by default.
- This violates production's needs and doesn't provide full insight into the switches.
- Solved by implementing a cron job that will SSH to Arista switches, run `top`, parse data via custom Python script, and send data to Splunk via syslog.
- All necessary data is now provided for Arista switches.

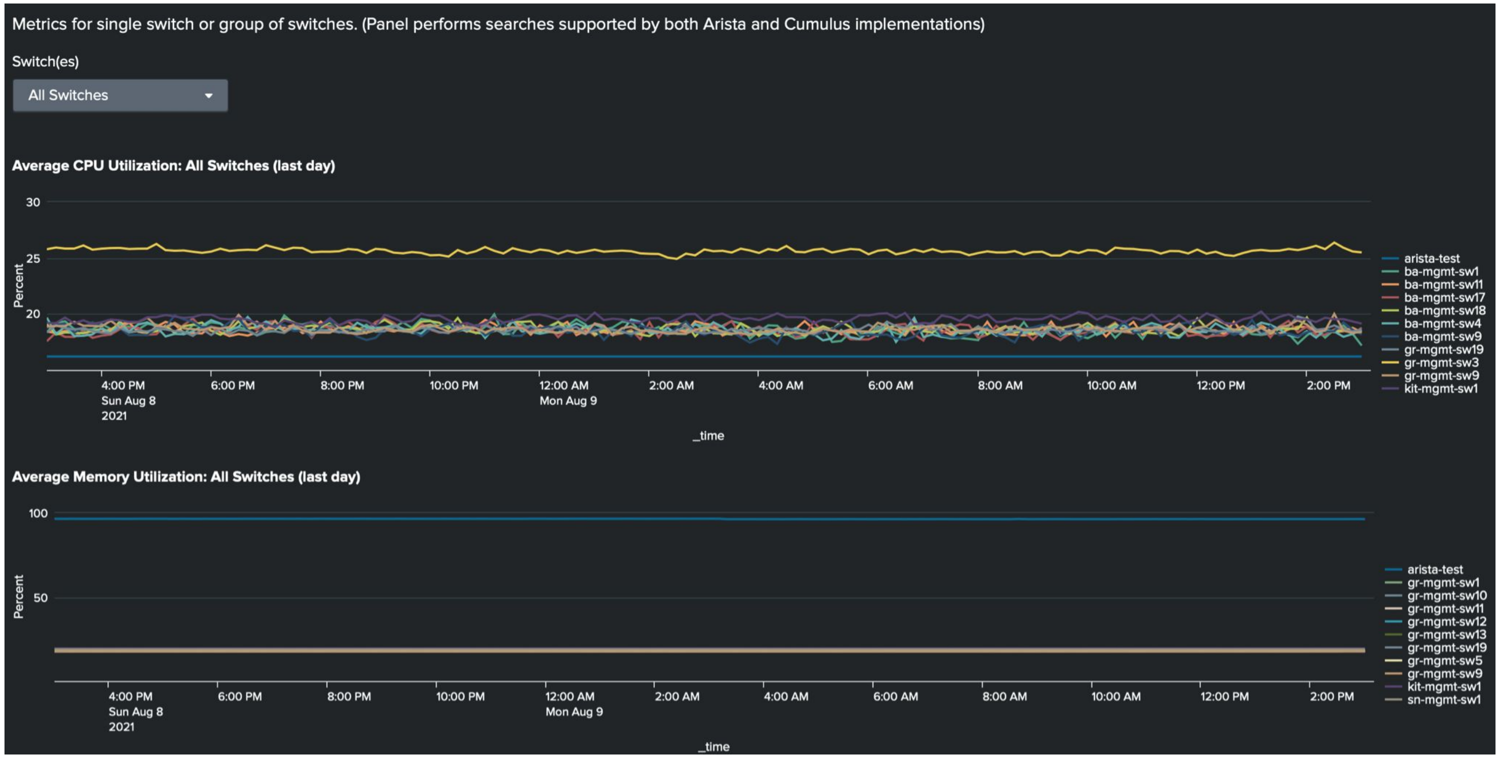


Time	Event
8/10/21 12:20:04.000 PM	`<14>Aug 10 12:20:04` [redacted] `sflow: switch=arista-test user_cpu_time=13.8 % system_cpu_time=1.8 % idle_cpu_time=83.8 % mem_total=3990868 KiB mem_used=3856228 KiB mem_free=134640 KiB` host = [redacted] | source = tcp:3514 | sourcetype = syslog
8/10/21 12:10:05.000 PM	`<14>Aug 10 12:10:05` [redacted] `sflow: switch=arista-test user_cpu_time=13.8 % system_cpu_time=1.8 % idle_cpu_time=83.8 % mem_total=3990868 KiB mem_used=3856336 KiB mem_free=134532 KiB` host = [redacted] | source = tcp:3514 | sourcetype = syslog

The Revised Deployment



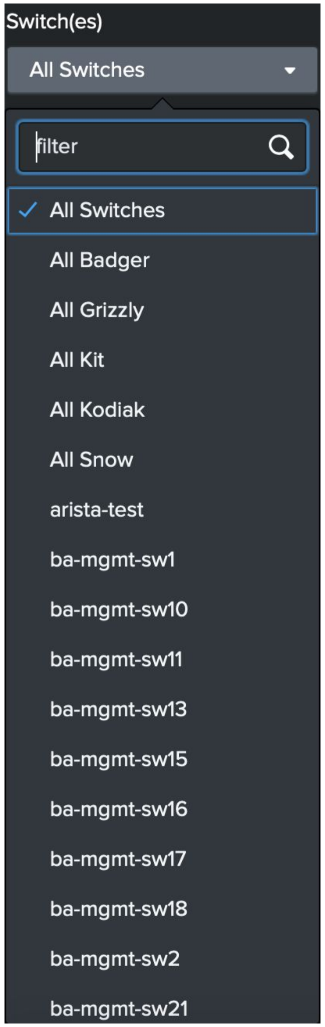
Splunk Dashboard - Preview 1


Metrics for single switch or group of switches. (Panel performs searches supported by both Arista and Cumulus implementations)

Switch(es): All Switches

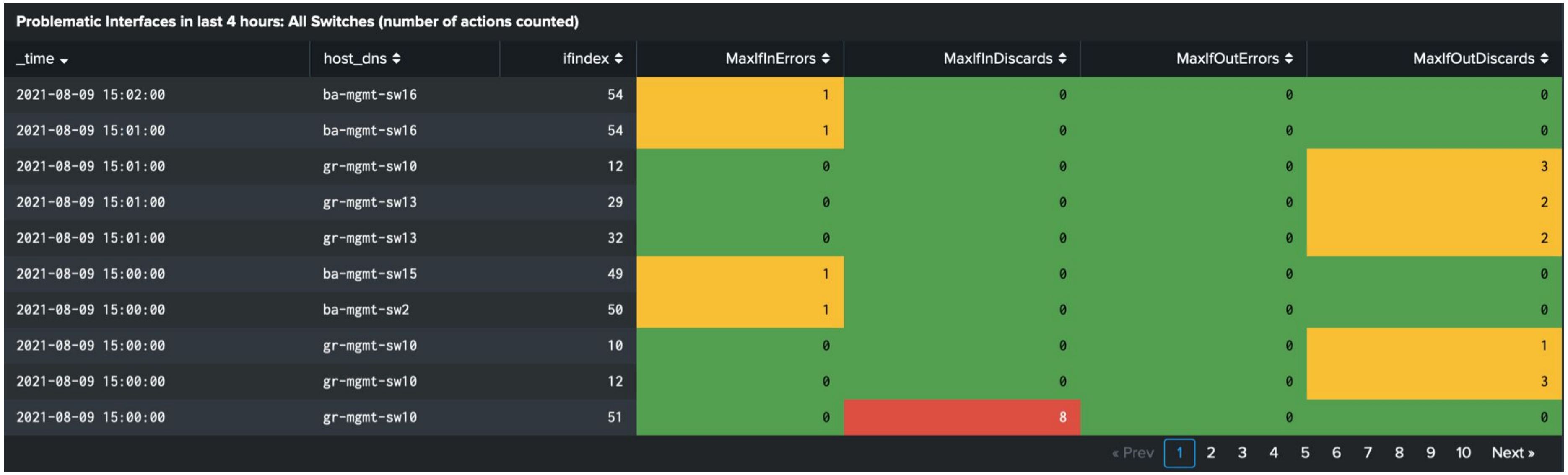
Average CPU Utilization: All Switches (last day)

Average Memory Utilization: All Switches (last day)



- Gr-mgmt-sw3 consistently has higher CPU utilization when compared to other Cumulus switches.
- One Arista switch consistently maintains 99% memory utilization. Discovered this is consistent with the specific model's performance, though only one Arista switch is in the current deployment.

Splunk Dashboard - Preview 2

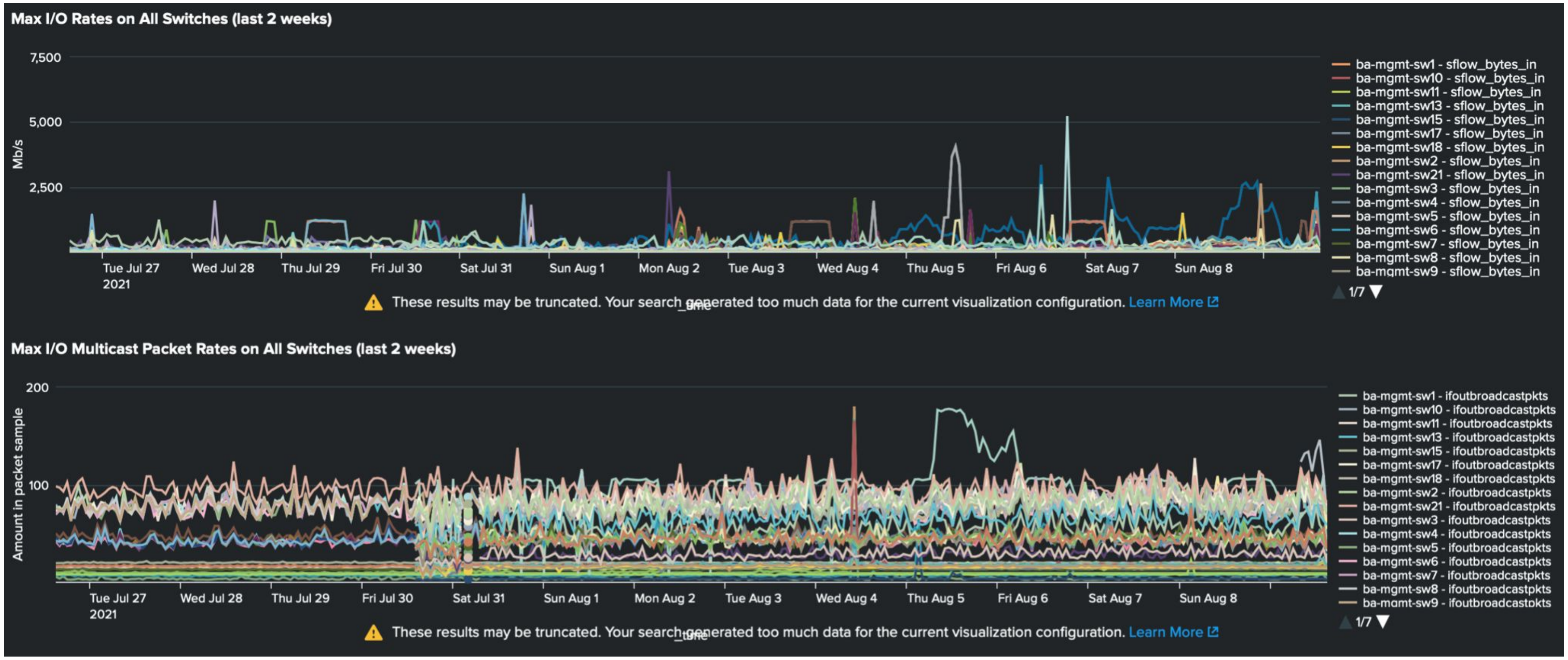


Problematic Interfaces in last 4 hours: All Switches (number of actions counted)

_time	host_dns	ifindex	MaxIfInErrors	MaxIfInDiscards	MaxIfOutErrors	MaxIfOutDiscards
2021-08-09 15:02:00	ba-mgmt-sw16	54	1	0	0	0
2021-08-09 15:01:00	ba-mgmt-sw16	54	1	0	0	0
2021-08-09 15:01:00	gr-mgmt-sw10	12	0	0	0	3
2021-08-09 15:01:00	gr-mgmt-sw13	29	0	0	0	2
2021-08-09 15:01:00	gr-mgmt-sw13	32	0	0	0	2
2021-08-09 15:00:00	ba-mgmt-sw15	49	1	0	0	0
2021-08-09 15:00:00	ba-mgmt-sw2	50	1	0	0	0
2021-08-09 15:00:00	gr-mgmt-sw10	10	0	0	0	1
2021-08-09 15:00:00	gr-mgmt-sw10	12	0	0	0	3
2021-08-09 15:00:00	gr-mgmt-sw10	51	0	8	0	0

```
| mstats max(telegraf.prometheus.sflow_ifinerrors) as MaxIfInErrors max(telegraf.prometheus.sflow_ifindiscards) as MaxIfInDiscards
  max(telegraf.prometheus.sflow_ifouterrors) as MaxIfOutErrors max(telegraf.prometheus.sflow_ifoutdiscards) as MaxIfOutDiscards WHERE index=telegraf_metrics host_dns=* BY
      host_dns, ifindex span=1m
| eval MaxIfInErrors = ceiling(MaxIfInErrors), MaxIfInDiscards = ceiling(MaxIfInDiscards), MaxIfOutErrors = ceiling(MaxIfOutErrors), MaxIfOutDiscards = ceiling
    (MaxIfOutDiscards)
| table _time, host_dns, ifindex, MaxIfInErrors, MaxIfInDiscards, MaxIfOutErrors, MaxIfOutDiscards
| where MaxIfInErrors!=0 OR MaxIfInDiscards!=0
  OR MaxIfOutErrors!=0 OR MaxIfOutDiscards!=0
```

Splunk Dashboard - Preview 3


Mb/s
7,500
5,000
2,500
Tue Jul 27 2021
Wed Jul 28
Thu Jul 29
Fri Jul 30
Sat Jul 31
Sun Aug 1
Mon Aug 2
Tue Aug 3
Wed Aug 4
Thu Aug 5
Fri Aug 6
Sat Aug 7
Sun Aug 8
_time
ba-mgmt-sw1 - sflow_bytes_in
ba-mgmt-sw10 - sflow_bytes_in
ba-mgmt-sw11 - sflow_bytes_in
ba-mgmt-sw13 - sflow_bytes_in
ba-mgmt-sw15 - sflow_bytes_in
ba-mgmt-sw17 - sflow_bytes_in
ba-mgmt-sw18 - sflow_bytes_in
ba-mgmt-sw2 - sflow_bytes_in
ba-mgmt-sw21 - sflow_bytes_in
ba-mgmt-sw3 - sflow_bytes_in
ba-mgmt-sw4 - sflow_bytes_in
ba-mgmt-sw5 - sflow_bytes_in
ba-mgmt-sw6 - sflow_bytes_in
ba-mgmt-sw7 - sflow_bytes_in
ba-mgmt-sw8 - sflow_bytes_in
ba-mgmt-sw9 - sflow_bytes_in
1/7
Amount in packet sample
200
100
Tue Jul 27 2021
Wed Jul 28
Thu Jul 29
Fri Jul 30
Sat Jul 31
Sun Aug 1
Mon Aug 2
Tue Aug 3
Wed Aug 4
Thu Aug 5
Fri Aug 6
Sat Aug 7
Sun Aug 8
_time
ba-mgmt-sw1 - ifoutbroadcastpkts
ba-mgmt-sw10 - ifoutbroadcastpkts
ba-mgmt-sw11 - ifoutbroadcastpkts
ba-mgmt-sw13 - ifoutbroadcastpkts
ba-mgmt-sw15 - ifoutbroadcastpkts
ba-mgmt-sw17 - ifoutbroadcastpkts
ba-mgmt-sw18 - ifoutbroadcastpkts
ba-mgmt-sw2 - ifoutbroadcastpkts
ba-mgmt-sw21 - ifoutbroadcastpkts
ba-mgmt-sw3 - ifoutbroadcastpkts
ba-mgmt-sw4 - ifoutbroadcastpkts
ba-mgmt-sw5 - ifoutbroadcastpkts
ba-mgmt-sw6 - ifoutbroadcastpkts
ba-mgmt-sw7 - ifoutbroadcastpkts
ba-mgmt-sw8 - ifoutbroadcastpkts
ba-mgmt-sw9 - ifoutbroadcastpkts
1/7

Max I/O Rates on All Switches (last 2 weeks)

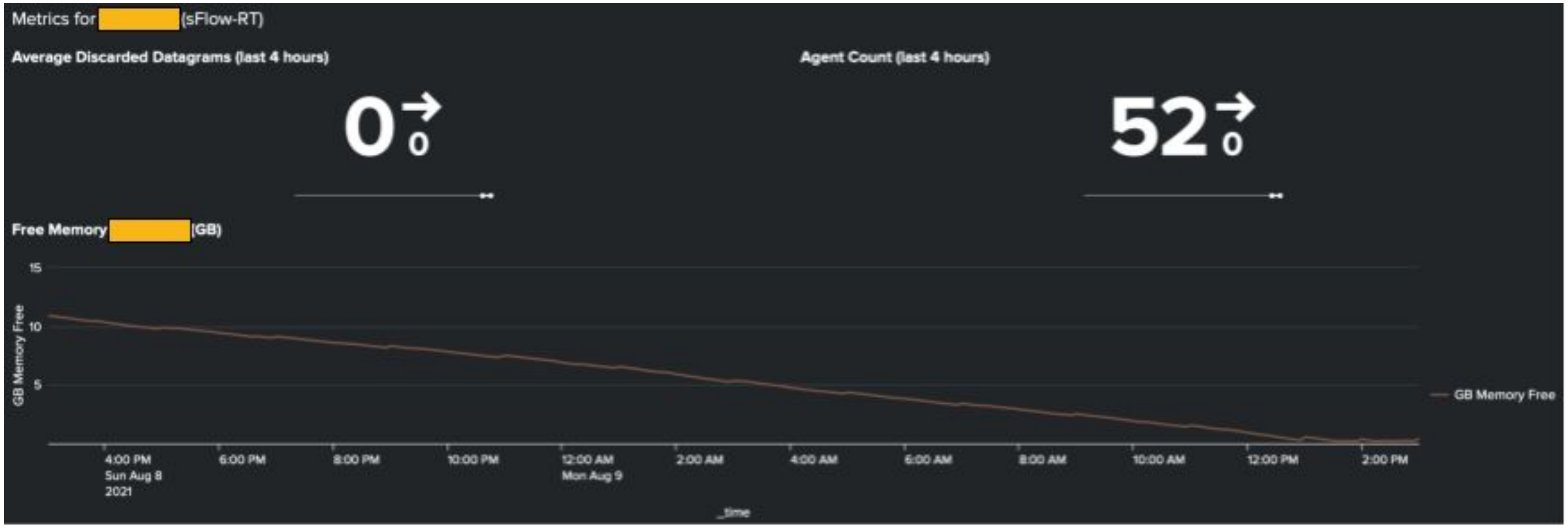
⚠ These results may be truncated. Your search generated too much data for the current visualization configuration. Learn More

Max I/O Multicast Packet Rates on All Switches (last 2 weeks)

⚠ These results may be truncated. Your search generated too much data for the current visualization configuration. Learn More

These searches replace custom monitoring scripts used in production.

Splunk Dashboard - Preview 4



No connections have been lost with sFlow agents, and datagrams are not being discarded. Server also has low free memory.

Future Work

- Implement dashboard alerts to trigger on data anomalies.
- Explore implementing custom metrics for additional data.
- Increase deployment scale beyond 52 switches.
- Implement sFlow on additional systems, projects, and environments.

HIGH PERFORMANCE COMPUTING

Acknowledgements and Sources

- Mentors:
  - Jesse Martinez
  - Brett Holman
- Special Thanks:
  - Thomas Areba
  - Dan Illescas
  - sFlow Community
  - Arista support team

- Sources:
  - Host sFlow, *Configuring Host sFlow for Linux via /etc/hsflowd.conf* https://sflow.net/host-sflow-linux-config.php
  - sFlow Blog, *Flow metrics with Prometheus and Grafana* https://blog.sflow.com/2019/10/flow-metrics-with-prometheus-and-grafana.html
  - sFlow-RT, *Metrics* https://sflow-rt.com/metrics.php

HIGH PERFORMANCE COMPUTING

Questions?

HIGH PERFORMANCE COMPUTING

Over 70 years at the forefront of supercomputing