INTRODUCING PANOPTES: A PYTHONIC NETWORK TELEMETRY PLATFORM

Varun Varma, Principal Software Engineer
Matt Hudgins, Senior Software Engineer

{vvarun, mhudgins} @ yahoo-inc.com

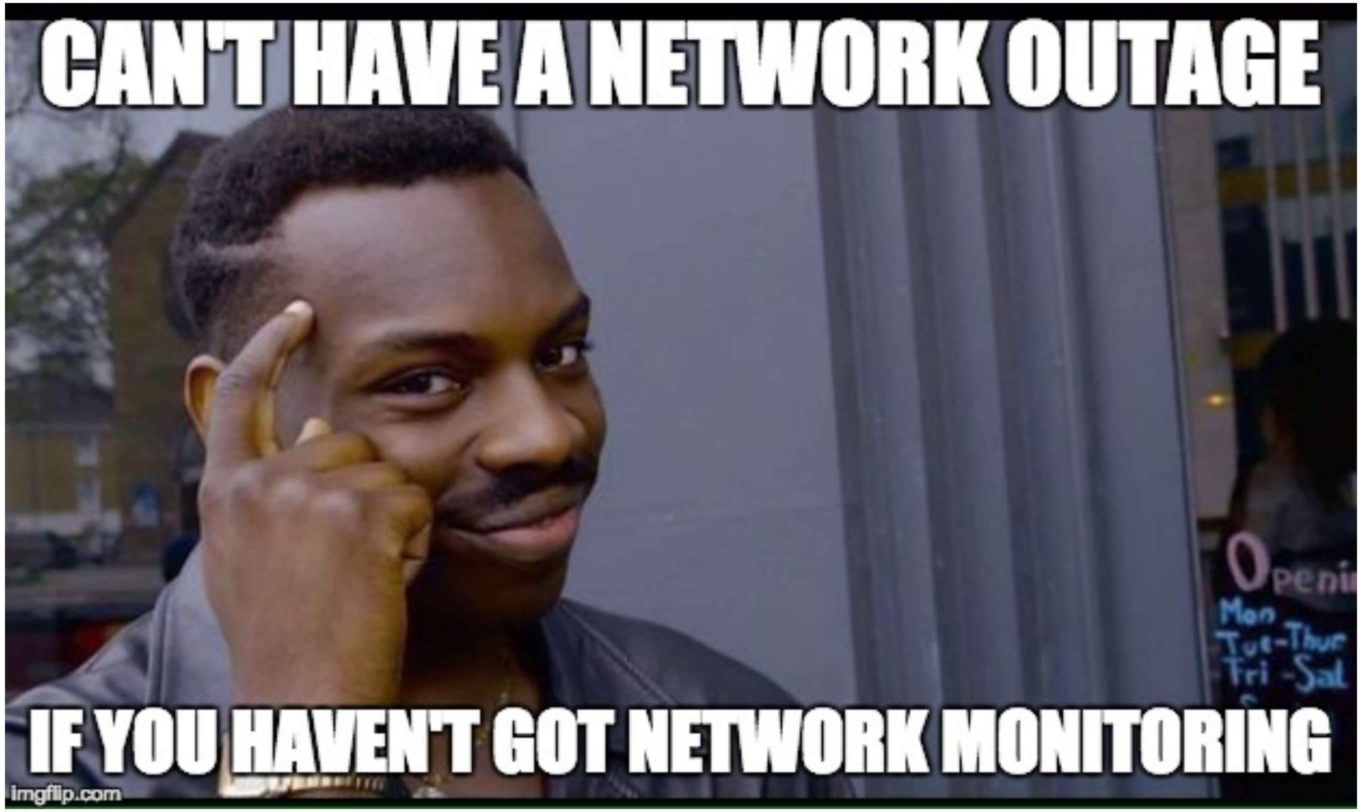
YAHOO!

WTF IS PANOPTES?

- Panoptes is our greenfield network telemetry platform that provides real time telemetry to Yahoo employees
- Yahoo’s production network consists of tens of thousands of multi-vendor network devices
- Easily accessible network telemetry enables powerful alerting, remediation and anomaly detection tools

IN THE BEGINNING

- Legacy Yahoo monitoring tools suffered from:
    - Overpolling
    - Data balkanization
    - SNMP dependence



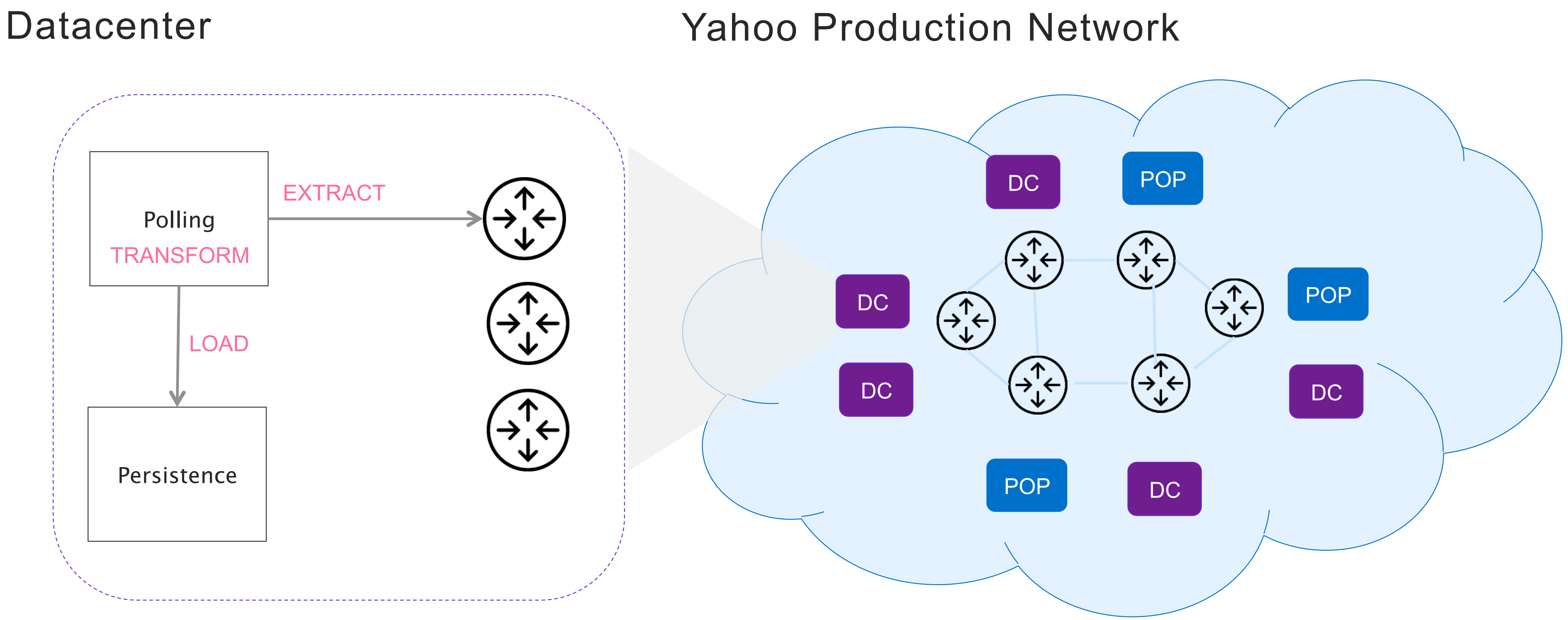
DESIGN GOALS

- **Extensible** – Minimize the effort required to poll new metrics or device types
- **Scalable** –Easily scale horizontally to meet new polling demands
- **Consumable** – Provide clean and understandable RESTful APIs for internal developers

ARCHITECTURE

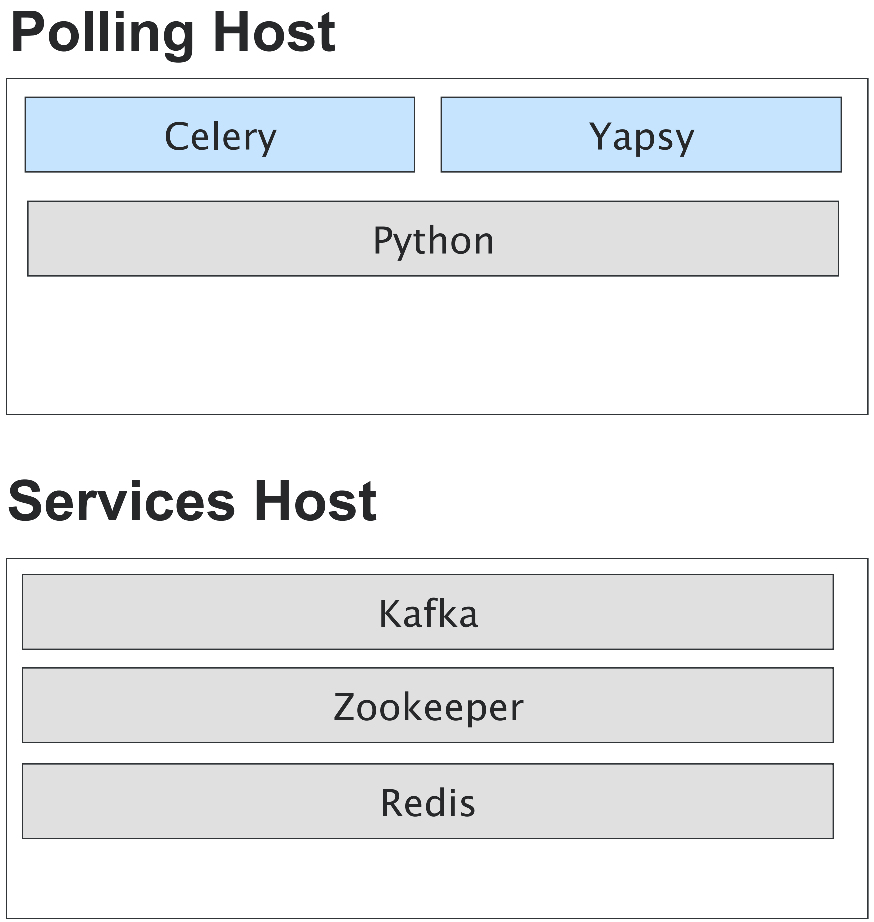
- Panoptes consists of highly available discovery, polling and persistence layers
- The platform’s primary abstractions are Python plugins and consumers
  - Plugin modules enumerate devices and poll telemetry
  - Consumer processes read polled data and load it into a configured data store

10,000 FOOT VIEW



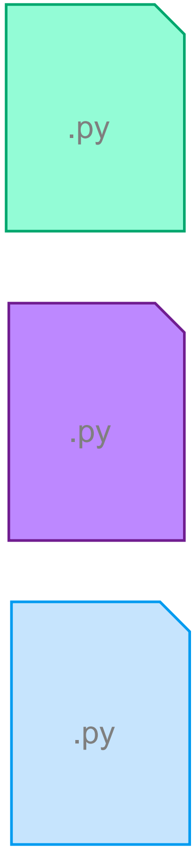
POLLING LAYER

- Panoptes polling plugins are Python modules that target specific device types and define what metrics to poll and how to poll them
- Worker hosts fetch tasks from Celery, an asynchronous task queue
- A Python process on the worker host executes the task and places the resultant Panoptes Metrics Group onto Kafka



POLLING SCHEDULING

1. **Device Discovery**
   - Polling hosts call internal services to enumerate devices
   - We cache discovered hosts for seven days to avoid service disruptions
2. **Polling Plugin Matching**
   - For each device discovered, try to find a matching polling plugin
3. **Polling Plugin Scheduling**
   - Place the polling plugin task on the queue for execution by the polling hosts
   - The polling host fetches a task to execute from the queue



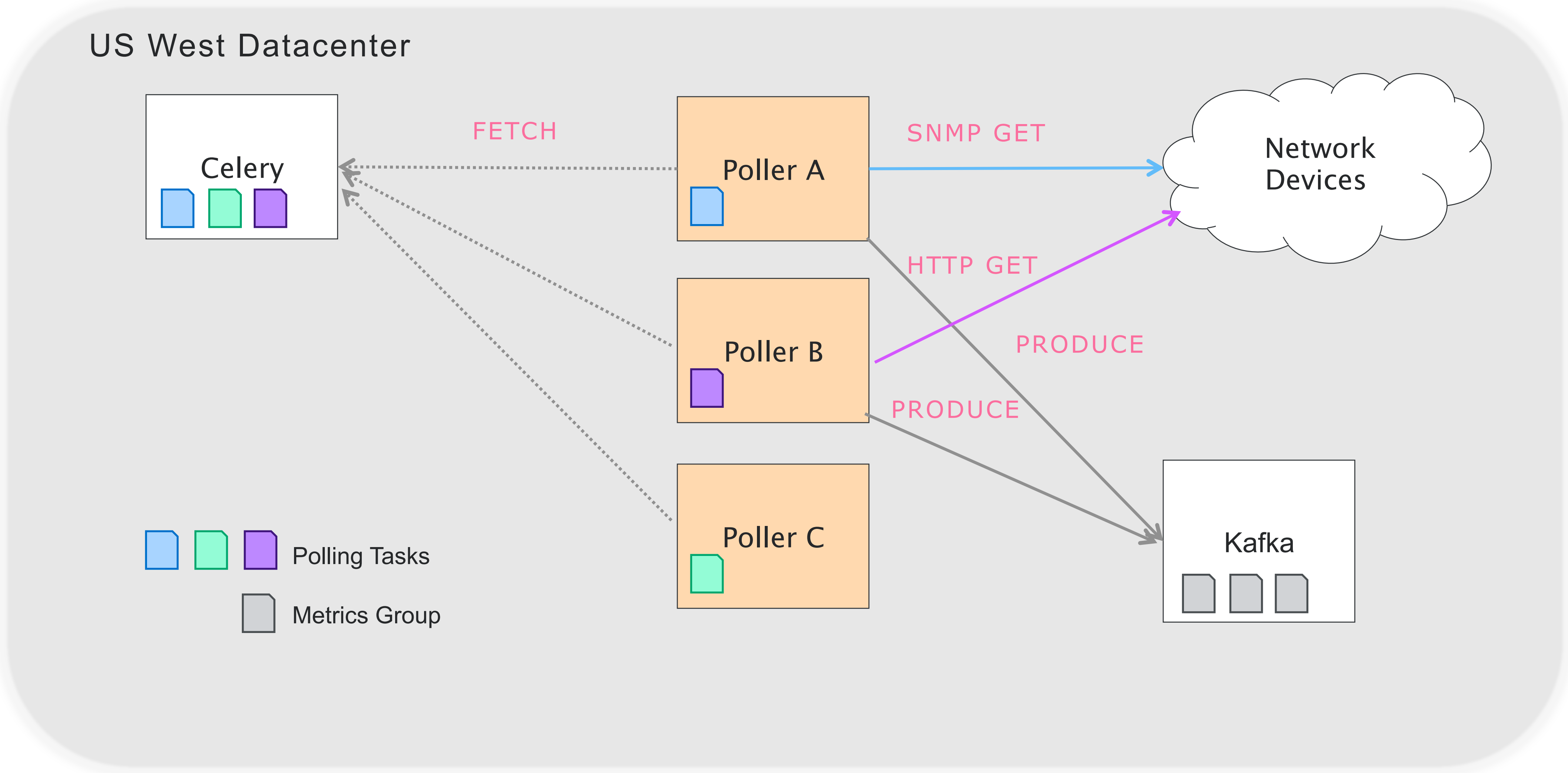
Make: Foo
Model: Pingmaster 1000
Method: SNMP

Make: Bar
Model: Big Ass Router
Method: SNMP

Make: Baz
Model: VIPinator
Method: REST API

Example polling plugins

POLLING PLUGIN EXECUTION



WHAT WE POLL

Element	Description	Example
Dimension	Dimensions are categories expressed as strings	bgp_adjacency_local_address
Dimension Value	Dimension values are explicit criteria	“1.1.1.1”
Metric Counter	Non-negative integers which monotonically increase until they wrap around (odometer)	interface_packets_sent
Metric Gauge	A point in time measurement that may increase or decrease (speedometer)	interface_packets_sent_rate

CONFIGURATION DRIVEN SNMP POLLING

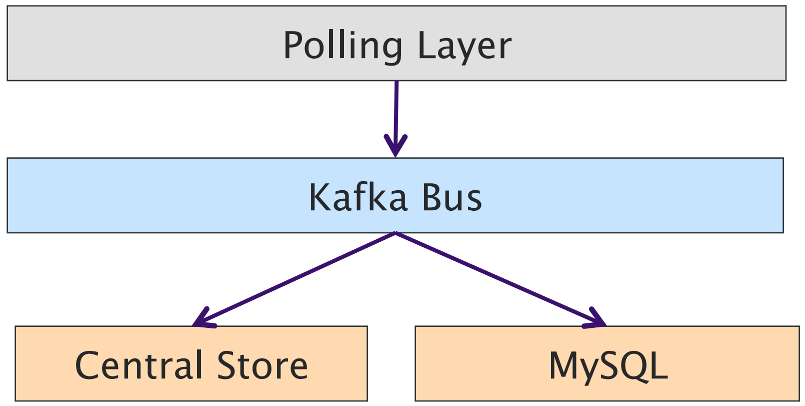
- Poll new metrics without having to write new functions
- An engineer specifies a Python dictionary with target OID(s) and how it maps to the resultant metrics group set:

```
{
    'oid': jnxBgpM2PeerEntry + '.7',
    'name': 'bgp_adjacency_local_address',
    'transform': 'ip',
    'type': 'dimension'
}
```

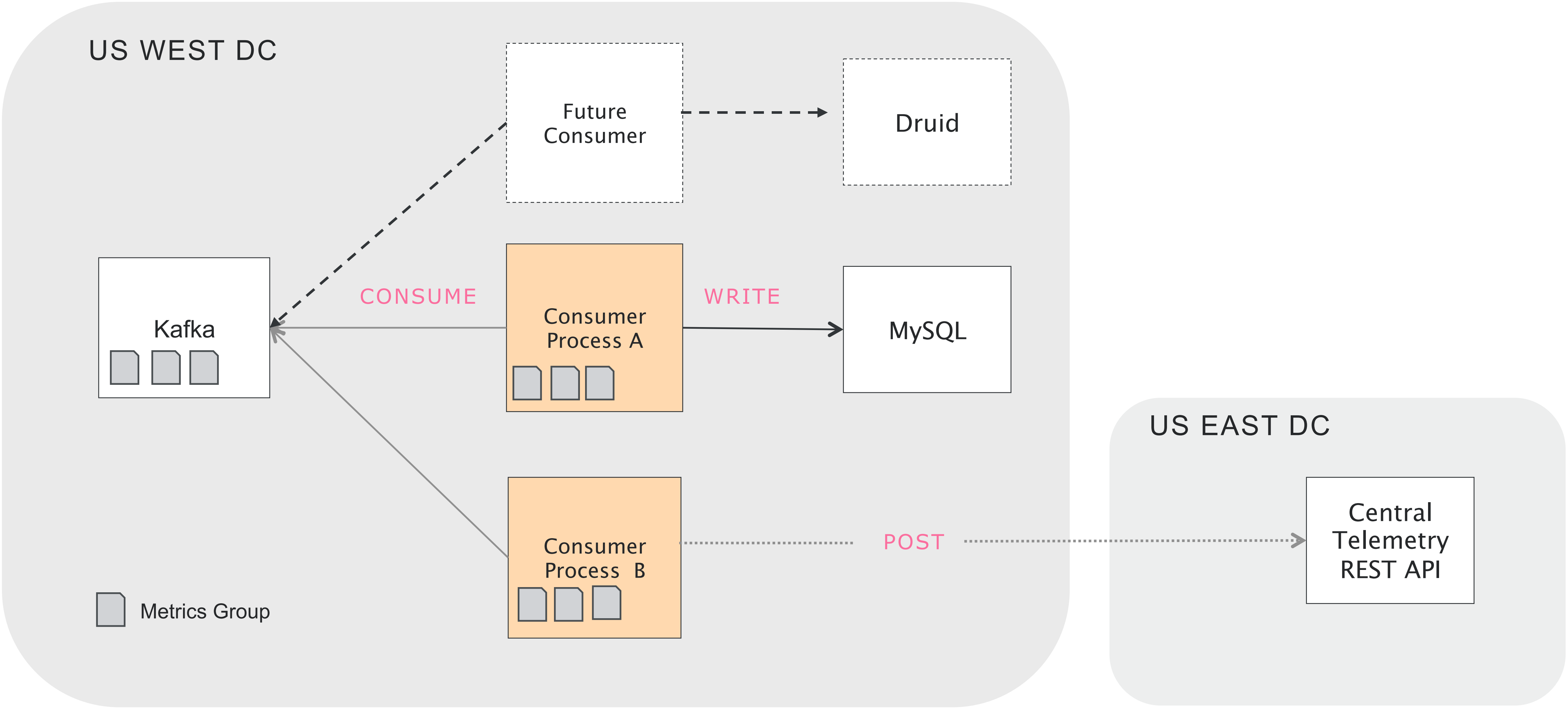
- A common library evaluates this data structure, issues the appropriate queries and emits a Panoptes metrics group

AFTER POLLING

- Kafka is the heart of our data distribution layer
- We do counter to gauge conversion and write back to Kafka
- A group of processes consume metrics from Kafka and writes the last point in time data to MySQL
- Another group of processes consume metrics and sends them to our centralized telemetry store



AFTER POLLING



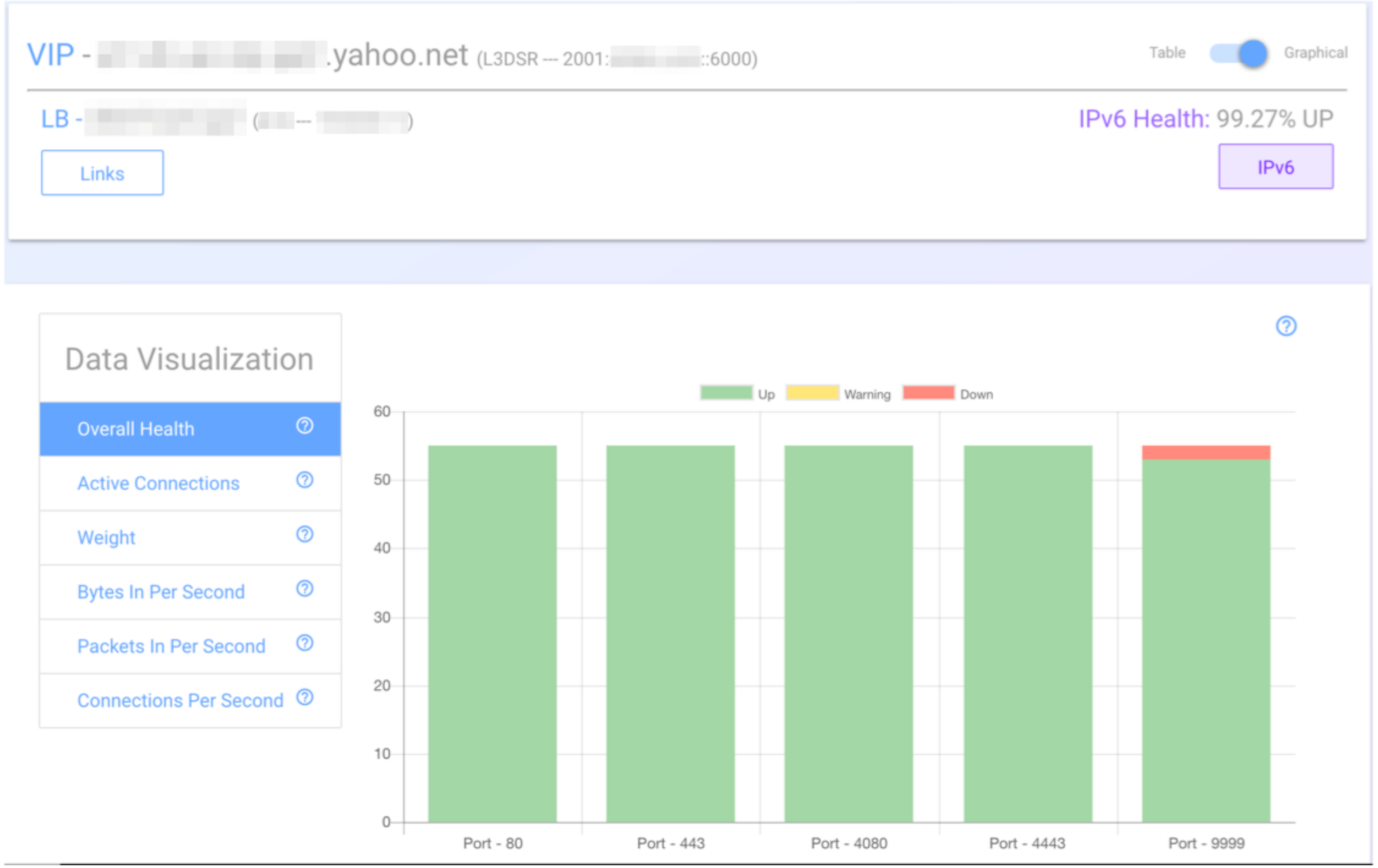
API EXAMPLE


```
{
  "members_metrics": [
    {
      "load_balancer_model": "[redacted]",
      "weight": 1,
      "site": "[redacted]",
      "vip": "[redacted]",
      "load_balancer_make": "[redacted]",
      "vip_property": "[redacted]",
      "max_connections": 100000,
      "bytes_in_gauge": 802742,
      "bytes_out_gauge": 0,
      "load_balancer_name": "[redacted]",
      "polling_interval": 60,
      "active_connections_gauge": 24307,
      "vip_port": 443,
      "status": 0,
      "pool_name": "[redacted]",
      "packets_out_gauge": 0,
      "timestamp": 1496772838,
      "real_port": 443,
      "vip_type": "l3dsr",
      "packets_in_gauge": 4221,
      "cache_age": 41,
      "ip_address": "[redacted]",
      "name": "[redacted]",
      "connections_per_second_gauge": 281,
      "total_connections_counter": 746440138,
```

LOAD BALANCER VIEWER

- Responsive Angular 2 application built from the in-colo MySQL telemetry data

- Used by support teams company wide to answer questions like:
  - What are the active connections on a given load balancer?
  - What is the overall health of the IPv4/IPv6 real?
  - What load balancers are in service for a given Yahoo! property?



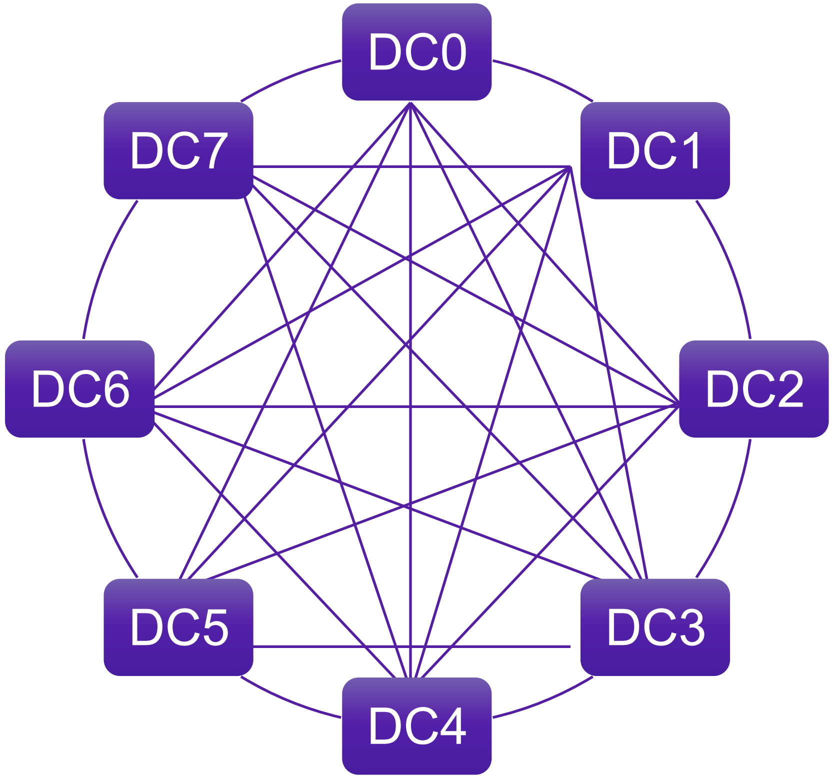
CENTRALIZED TELEMETRY SERVICE

- We push metrics to Yahoo's in-house time series database and alerting service (centralized telemetry)
- Custom dashboard service our user base is familiar with
- Economies of scale – no need to provision new hardware or software

Here we see control and data plane CPU statistics for a load balancer in one of our West Coast data centers.

FEDERATED API

- Due to availability concerns, each site has its own MySQL cluster
  - Telemetry data must be available during a network partition
  - Centralized telemetry store might not be reachable in all cases
- Each API endpoint acts as a tribe node
  - If a tribe node doesn't have the requested data, it returns a pointer to the node that does through a find API



CURRENT STATUS

- Deployed in all our production data centers across five continents
- Panoptes polls, processes and stores millions of metrics per minute from production load balancers and BGP speaking routers
- All Yahoo service owners use Panoptes-collected load balancer telemetry for troubleshooting and capacity planning

LESSONS LEARNED

- Python is fun to write, but painfully slow in some cases; luckily, C interactions are easy
- Creating a functional testbed requires a significant upfront investment
- RESTful APIs: if you build it, they will come

FUTURE

- Data availability is the prerequisite for more advanced use cases:
  - Anomaly detection
  - Machine learning
  - Auto-remediation
- Streaming telemetry
- Poll the rack switch layer – 10x increase in the number of polled devices
- This project wouldn’t exist without OSS: Python, Kafka, Linux…to name a few
  - Leadership mandate to open source Panoptes

SHOUT-OUTS

- We would like to thank some of our colleagues for their ideas, support, motivation and work:
  - Ian Flint
  - Sean Wade
  - Stormy Adams
  - Sutha Thangavel
  - Malcolm Flint
  - Jessica Tang
  - Vivek AM

QUESTIONS?