Anomaly Detection for Cloud Native Storage

Xing Yang, OpenSDS and Seiya Takei, Yahoo! JAPAN

Agenda

- Introduction to anomaly detection
  - Anomaly detection in storage performance
- Yahoo! JAPAN use cases
- Telemetry
  - Integrate with Prometheus and Grafana
  - Collect performance metrics from storage backends
  - Metrics drivers: LVM, Ceph, other storage systems…
- Anomaly detection
  - Detect anomalous data points based on metrics collected from Telemetry.
- Demo

What is Anomaly Detection

- Anomaly detection is a technique used to identify unusual patterns that do not conform to expected behavior, called outliers.
- Categories of anomalies:
  - Point anomalies
  - Contextual anomalies
  - Collective anomalies

Anomaly Detection Use Cases

- Intrusion detection
  - identifying strange patterns in network traffic that could signal a hack
- Medical health
  - spotting a malignant tumor in an MRI scan
- Fraud detection
  - credit card, cell phone, insurance claim fraud, etc.
- Fault detection
  - mechanical units, etc.
- Anomaly detection in storage
  - disk failure, etc.

Storage Performance Challenges

Bottlenecks

- Disk failure/Inaccessible disks
- Read/Write I/O errors
- Volume issues
- Port masking
- Configuration issues – Host, Storage subsystem, port, Interoperability
- Network congestion
- Workload configurations
- UPS battery failure
- Port protocol errors,
- Port congestion

Metrics

- I/O Rate R/W,
- Data Rate R/W,
- Response time R/W,
- Cache hit R/W,
- Data block size R/W,
- Porta data rate R/W,
- Port-local node queue time

Correlations

- CPU & Network Traffic
- CPU & Memory
- Port & Host counters
- IOPs, read rate, & CPU, memory

Source: Using Machine Learning for Intelligent Storage Performance Anomaly Detection

Yahoo! JAPAN's Environment

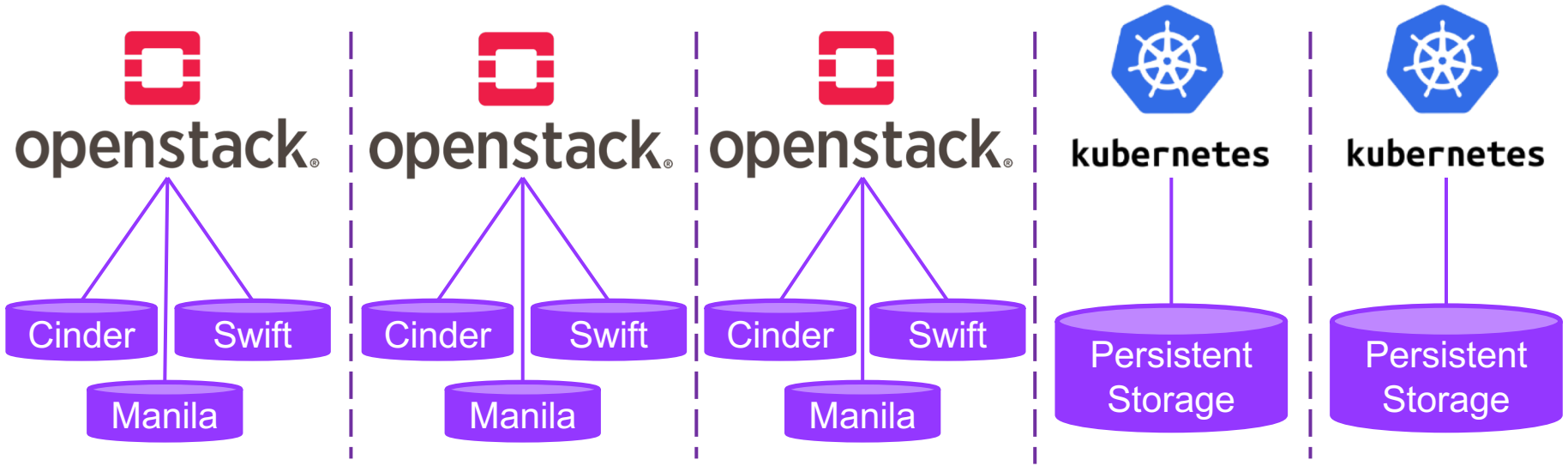
- Private Cloud
  - IaaS
    - 140,000 VMs
  - PaaS
    - 30,000 Containers
  - CaaS
    - 390 Kubernetes clusters

openstack®

kubernetes

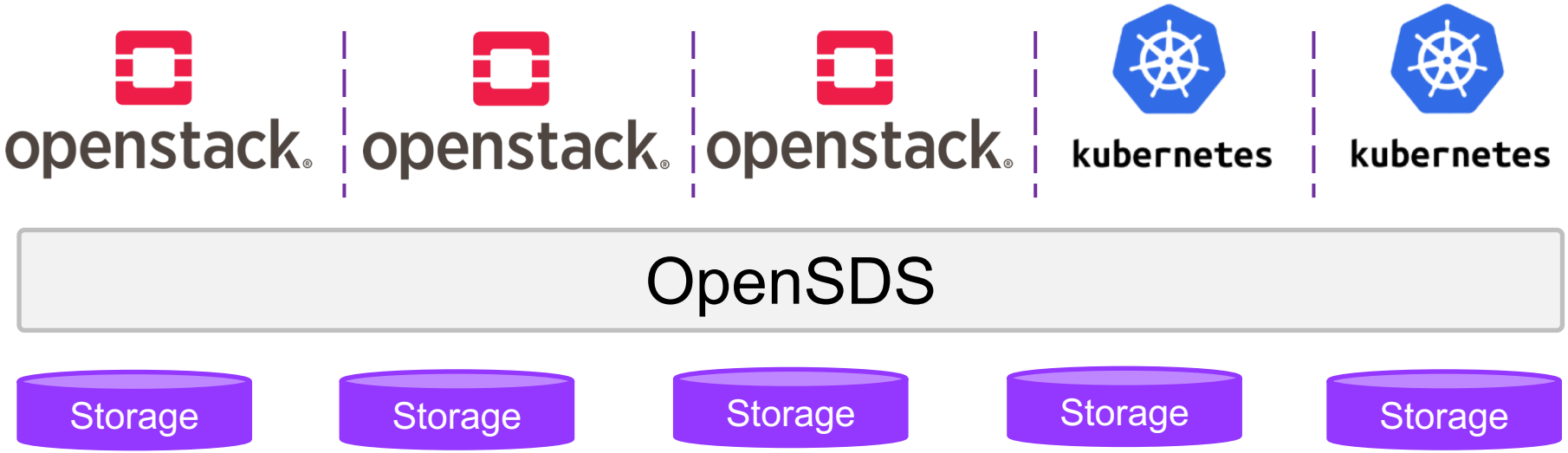
Storage for Private Cloud

- OpenStack/Kubernetes clusters each have storage
- There are many storages in Yahoo! JAPAN's environment



Future environment we hope

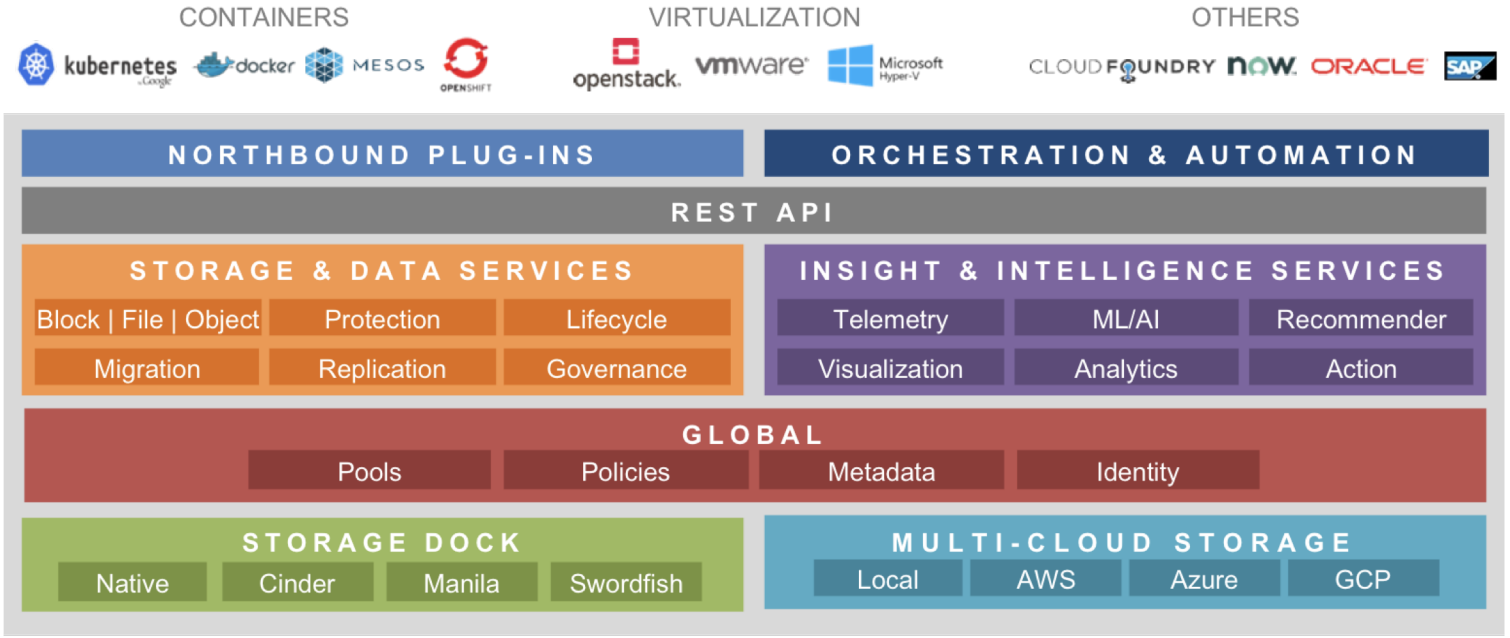
- We want to manage storages using OpenSDS
- If telemetry and anomaly detection can be managed by OpenSDS, we can manage many storages easier



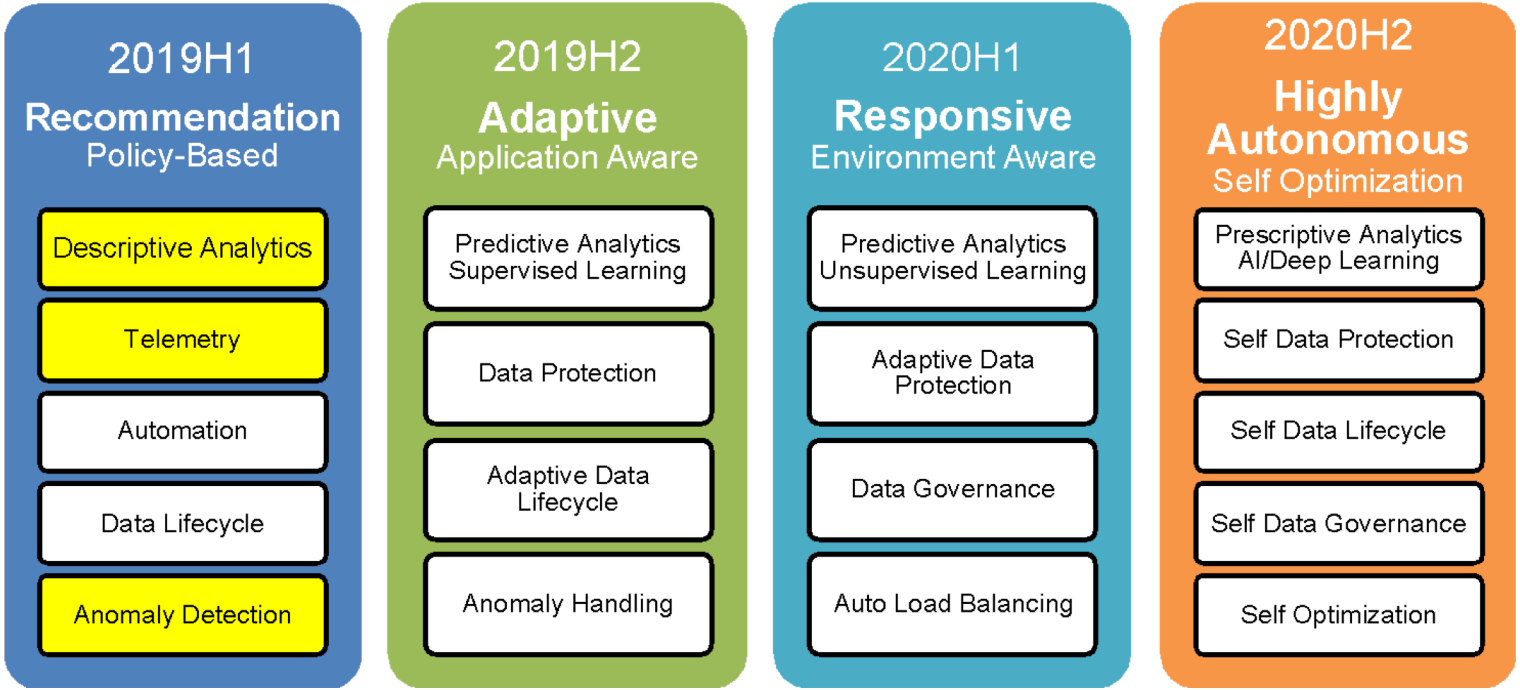
Software Defined Storage

- OpenStack/kubernetes backend storage
- Running SDS in Yahoo! JAPAN's environment
  - Ceph
    - Using in test environment
  - Quobyte
    - Started using it end of last year
- Difficult to operate
  - Health management of distributed system
    - server, network, ...
  - Telemetry and Anomaly Detection are very important

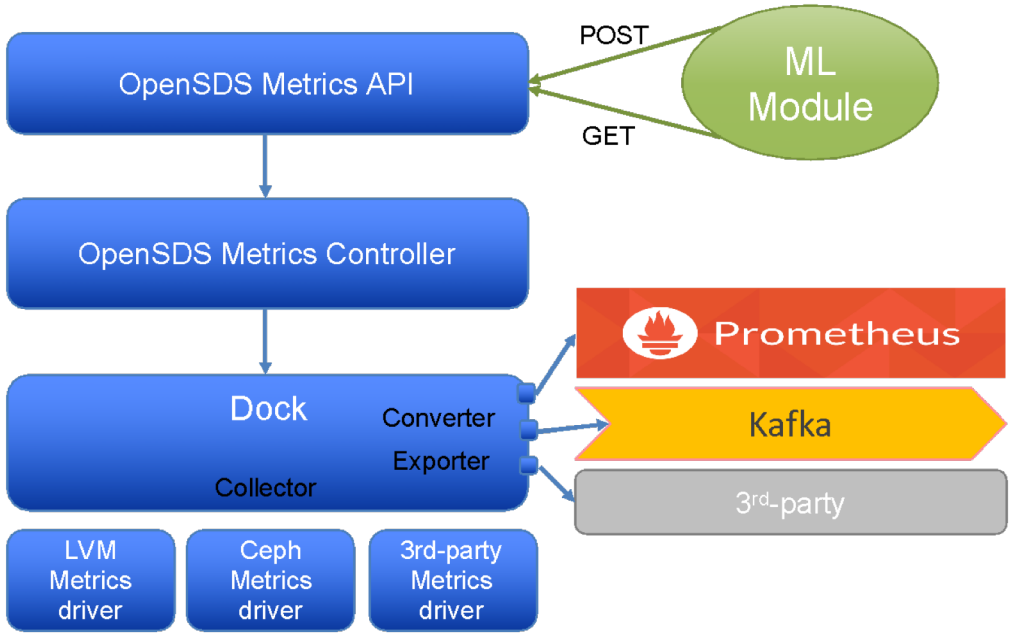
The Open Autonomous Data Platform



The Road To Autonomous Data Platform

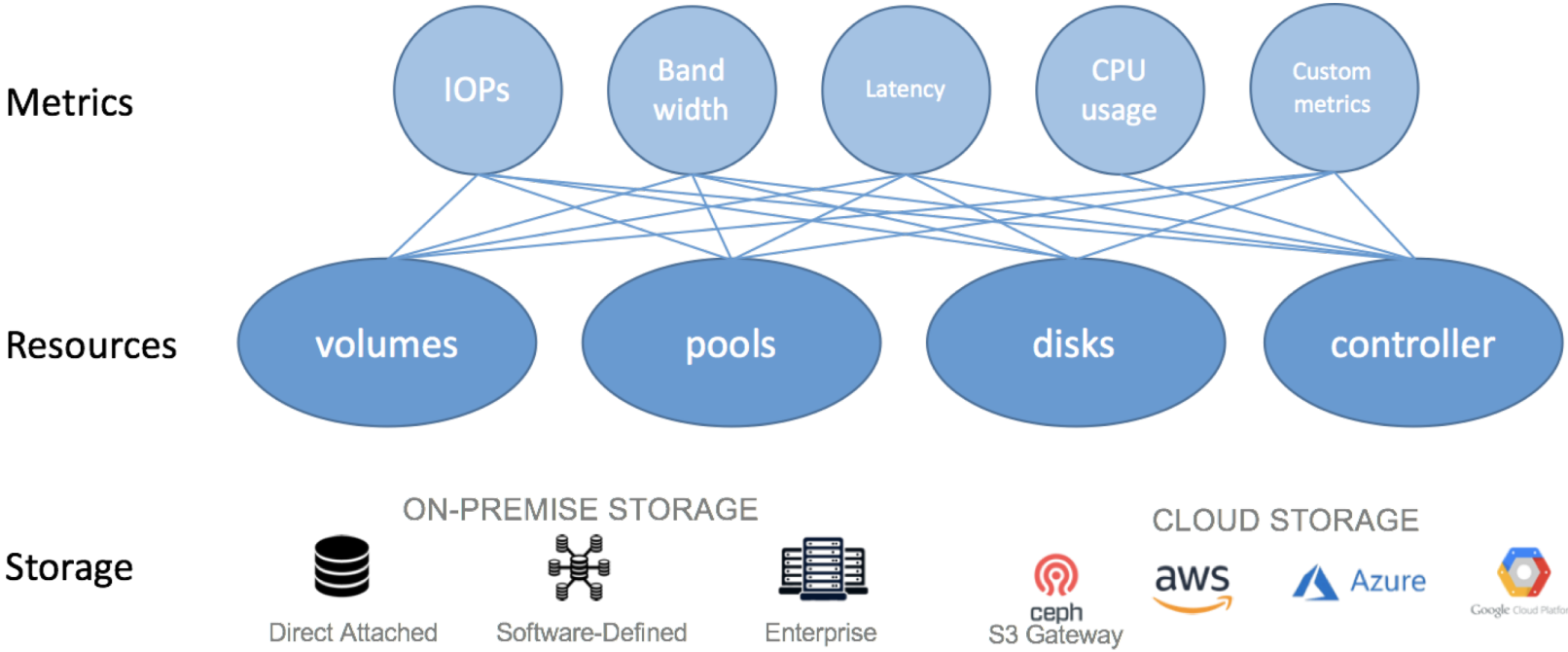


Telemetry

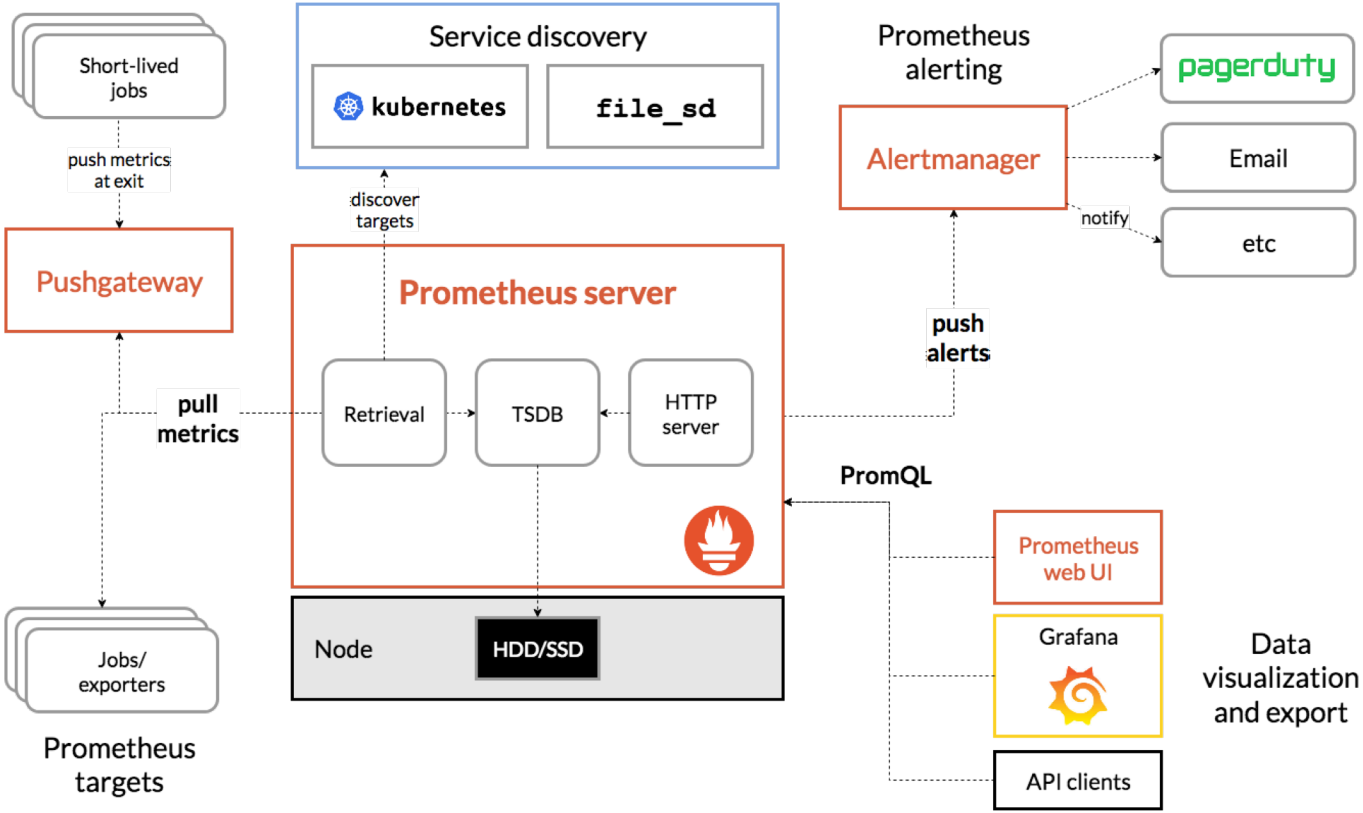


- ML module sends requests using Metrics API that generates data.
- Collector collects metrics from metrics drivers.
- Adapter includes a Converter that converts data to a proper format that can be understood by the receiving end, e.g., Prometheus, and an Exporter that sends(emits) the data to the intended destination.
- ML module receives data through Kafka. ML module also retrieves additional data using Metrics API which gets data from Prometheus.
- Collected metrics include IOPs, bandwidth, latency, average CPU usage, etc. for various resources such as storage controller, pools, volumes, disks, etc. For Ceph, an existing Prometheus Ceph exporter will be used. Prometheus Node exporter will also be used to collect node metrics.
- OpenSDS dashboard is integrated with Grafana to display metrics and Prometheus Alert Manager to show alerts.

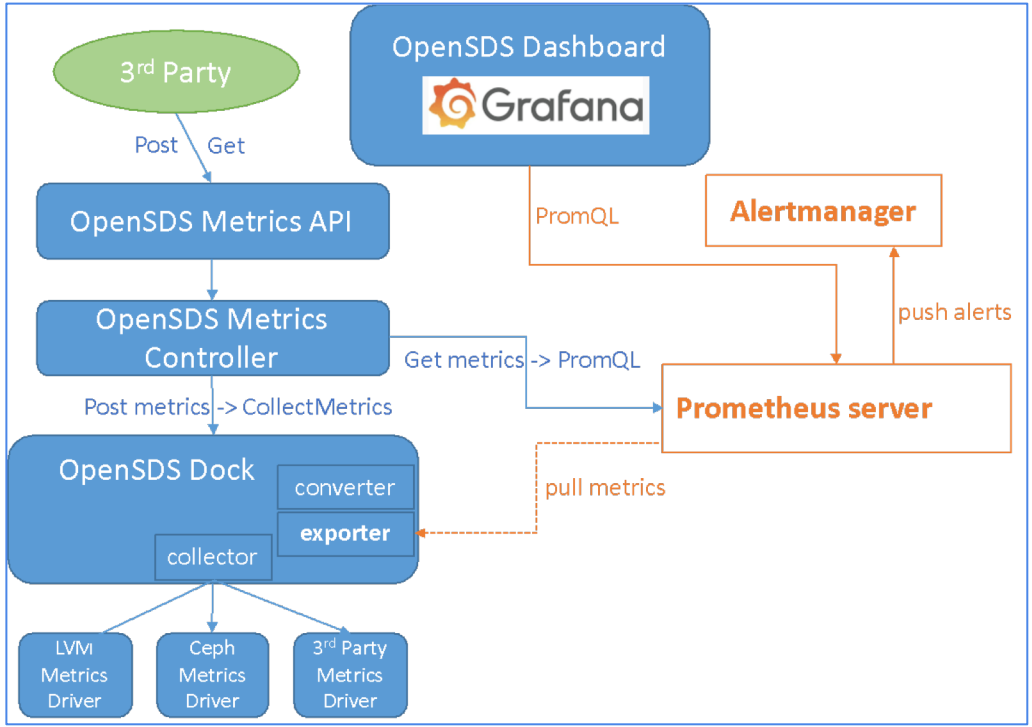
Collecting Storage Performance Metrics



Prometheus Architecture



Emit Metrics to Prometheus



- Post request will be sent to the Metrics driver to collect metrics
- Get request will be re-routed to Prometheus server using PromQL
- Metrics will be saved in Prometheus database.

Metrics interface

func CollectMetrics() ([]*model.MetricSpec, error)

```
type CollectMetricSpec struct {
        *BaseModel

        DriverType string
}


type GetMetricSpec struct {
        *BaseModel

        InstanceId string
        MetricName string
        StartTime string
        EndTime string
}
```

```
type MetricSpec struct {
        InstanceID string
        InstanceName string
        Job string
        Labels map[string]string
        Component string
        Name string
        Unit string
        AggrType string
        MetricValues []Metric
}

type Metric struct {
        Timestamp int64
        Value     float64
}
```

Collect LVM Metrics

- Tools:
  - lvmsar (LVM system activity reporter)
  - iostat

- resource: volume
- metrics:
  - iops (tps)
  - read_throughput (kb/s)
  - write_throughput (kb/s)
  - response_time (ms)
  - service_time (ms)
  - utilization_percentage (%)

- resource: disk
- metrics:
  - iops (tps)
  - read_throughput (kb/s)
  - write_throughput (kb/s)
  - response_time (ms)
  - service_time (ms)
  - utilization_percentage (%)

Collect Ceph Metrics

- Use existing Ceph exporter in Prometheus

https://prometheus.io/docs/instrumenting/exporters/

- RabbitMQ Management Plugin exporter

Storage

- Ceph exporter
- Ceph RADOSGW exporter
- Gluster exporter
- Hadoop HDFS FSImage exporter
- Lustre exporter
- ScaleIO exporter

HTTP

- Apache exporter
- HAProxy exporter (official)
- Nginx metric library
- Nginx VTS exporter

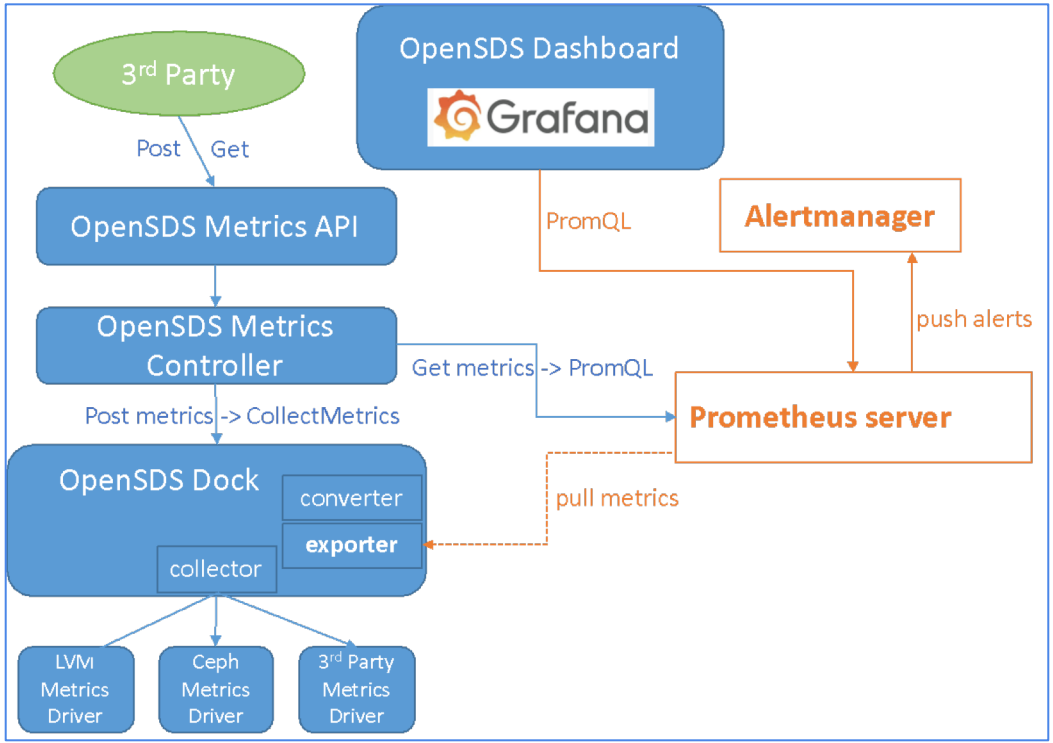
- resource: pool
- metrics:
  - pool_used_bytes
  - pool_available_bytes
  - pool_objects_total
  - pool_dirty_objects_total
  - pool_read_total
  - pool_read_bytes_total
  - pool_write_total
  - pool_write_bytes_total

- resource: cluster
- metrics:
  - ……

LVM Metrics in Grafana

LVM Metrics in Grafana

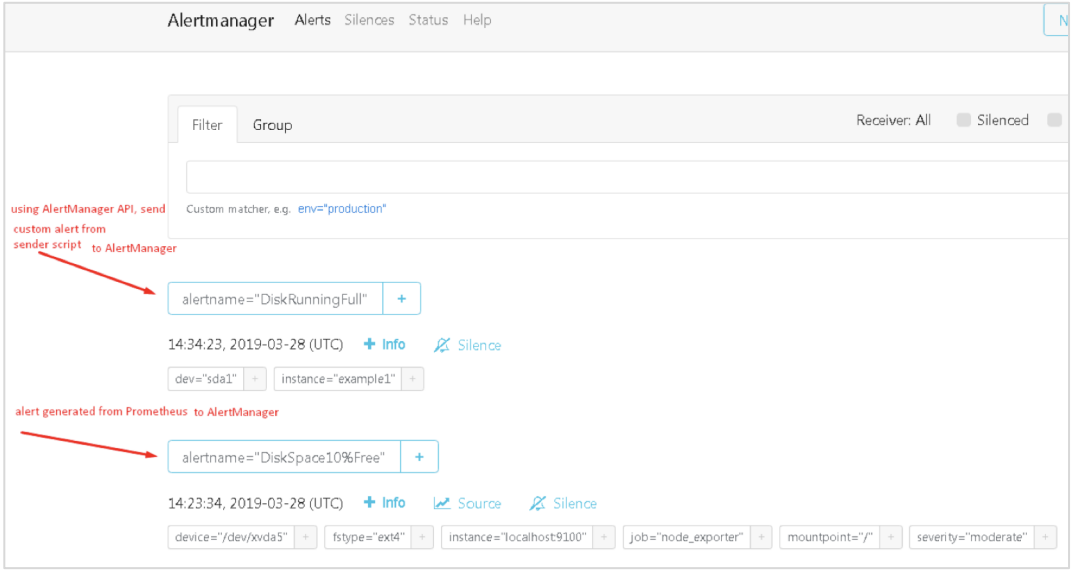
Emit Metrics to Prometheus (recap)



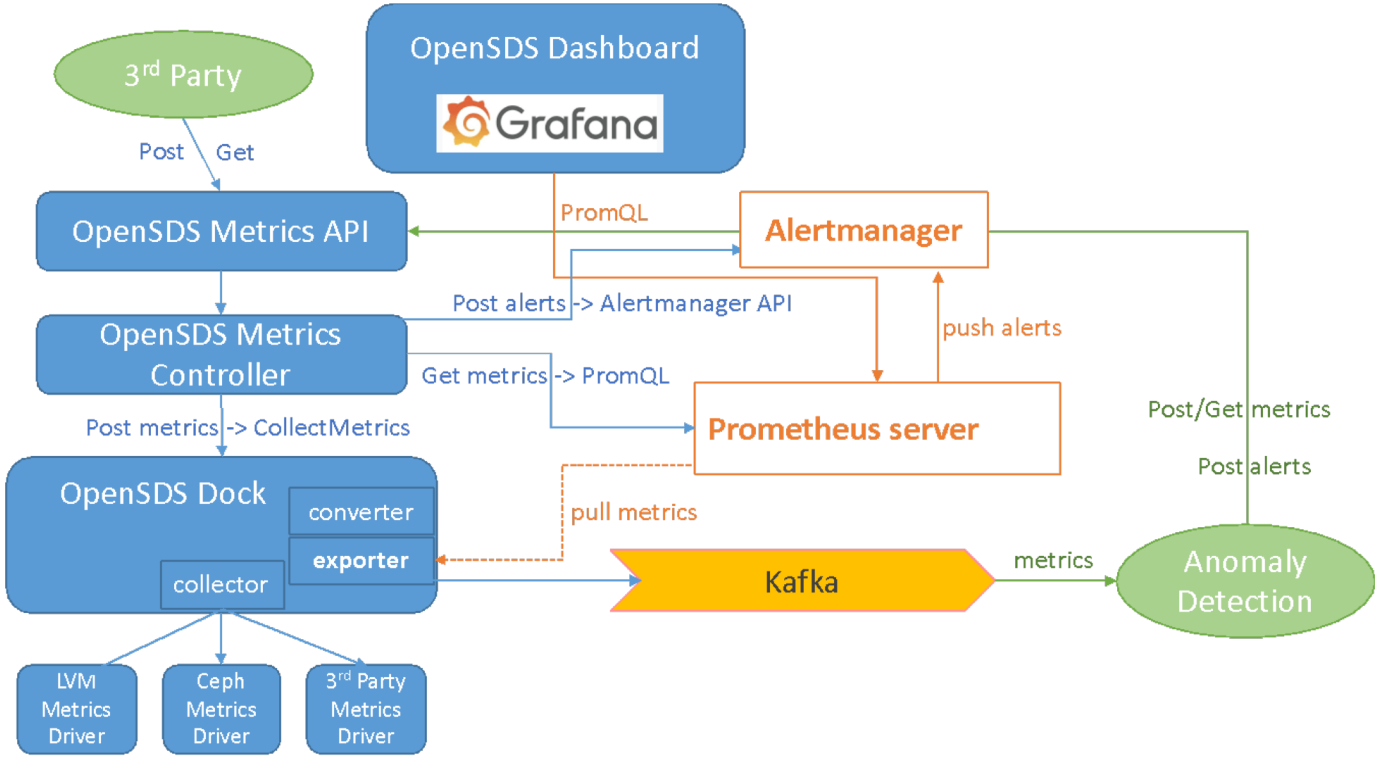
- Post request will be sent to the Metrics driver to collect metrics
- Get request will be re-routed to Prometheus server using PromQL
- Metrics will be saved in Prometheus database.

Prometheus Alert Manager

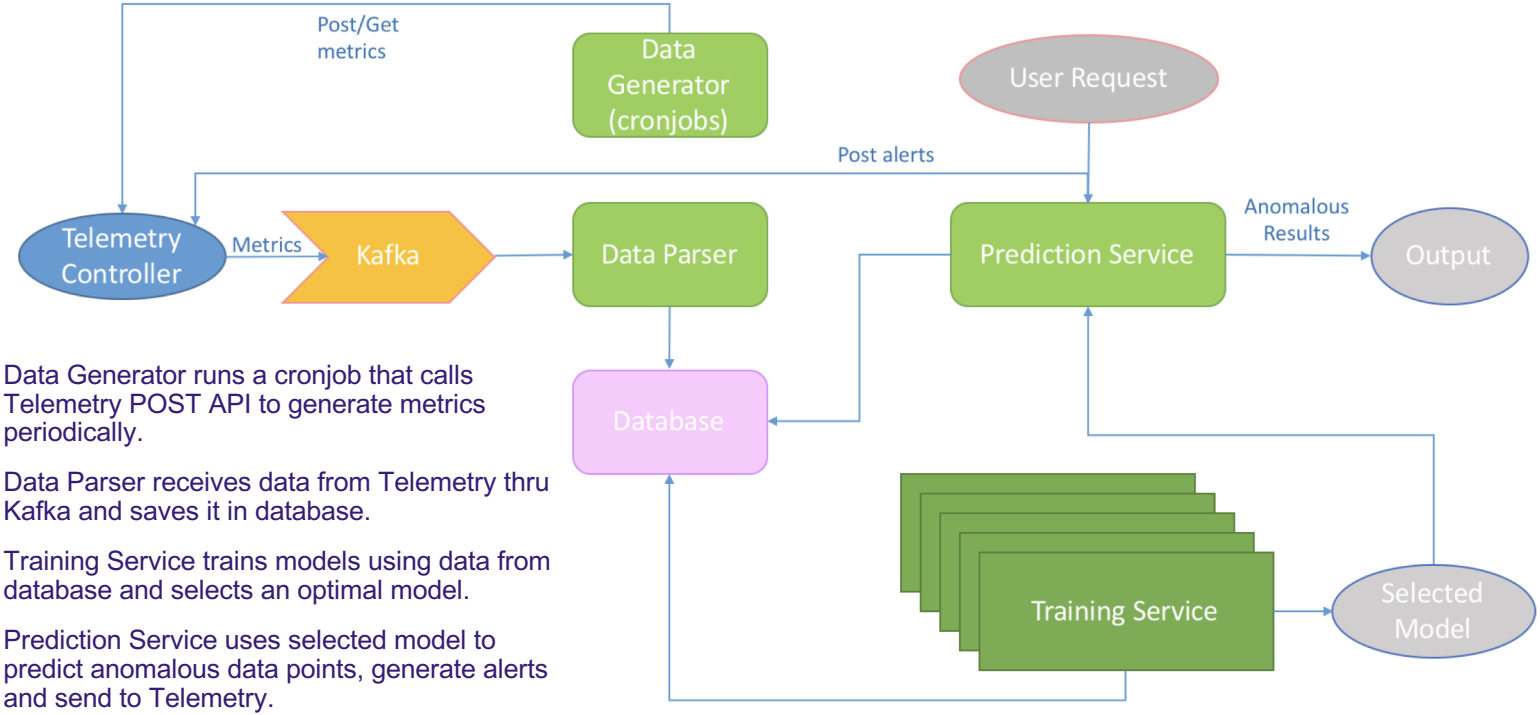
- The alert rules are configured in Prometheus, and on the thresholds being crossed, Prometheus will raise an alert to the Alertmanager. Alerts can be defined on raw metrics and derived metrics.
- The Anomaly Detection module detects an anomaly, raises a custom alert to Alertmanager using the REST API interface of Alertmanager



Send Metrics through Kafka



Anomaly Detection Architecture



- Data Generator runs a cronjob that calls Telemetry POST API to generate metrics periodically.
- Data Parser receives data from Telemetry thru Kafka and saves it in database.
- Training Service trains models using data from database and selects an optimal model.
- Prediction Service uses selected model to predict anomalous data points, generate alerts and send to Telemetry.

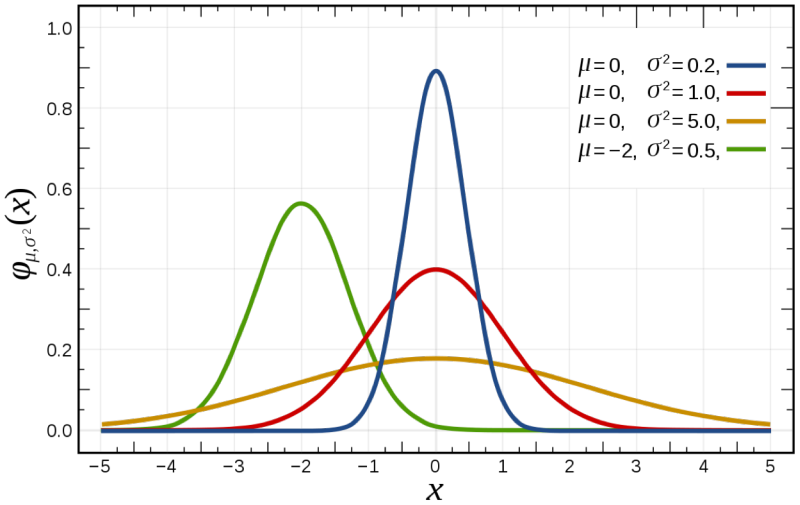
Anomaly Detection Algorithms

- Classification based
  - A classifier that can distinguish between normal and anomalous classes can be learned in the given feature space.
- Nearest neighbor based
  - Normal data instances occur in dense neighborhoods, while anomalies occur far from their closest neighbors.
- **Clustering based**
  - Normal data instances belong to a cluster in the data, while anomalies either do not belong to any cluster.
- Information theoretic
  - Anomalies in data induce irregularities in the information content of the data set.
- Spectral
  - Data can be embedded into a lower dimensional subspace in which normal instances and anomalies appear significantly different.
- Statistical models
  - **Gaussian model**
  - Regression model

Reference: Anomaly Detection : A Survey

Gaussian Model

- The Gaussian model is a statistical model that assumes the pattern of the dataset follows the gaussian distribution.
- A threshold needs to be specified to differentiate between normal and abnormal data points.



Source: Normal Distribution

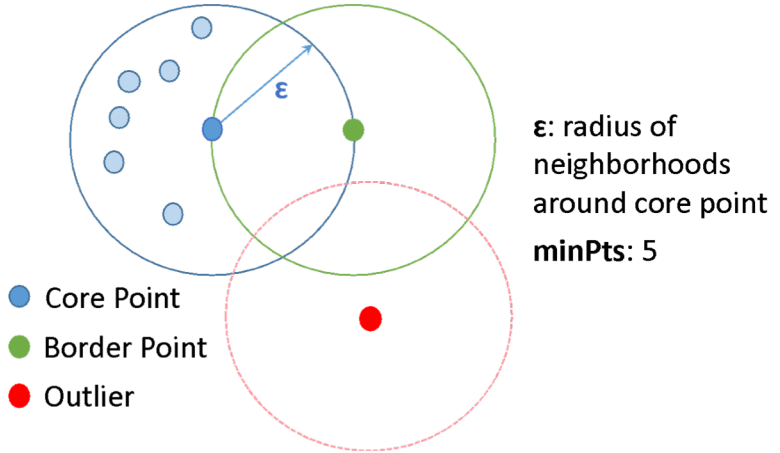
Anomaly detection

Anomaly detection using the multivariate Gaussian distribution

by Andrew Ng

DBSCAN Clustering

- DBSCAN refers to Density-Based Spatial Clustering Applications with Noise. Clustering is used to group similar data instances into clusters. DBSCAN is a clustering model designed to discover clusters of arbitrary shape based on density.
- The algorithm has two input parameters **ε** and **minPts**.

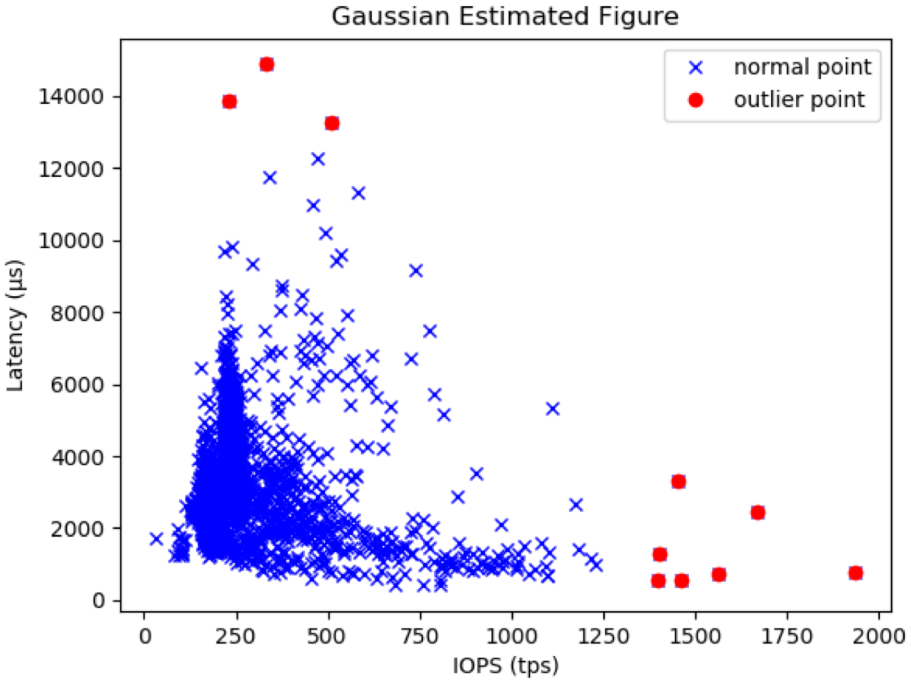


DBSCAN categories the data points into three categories: Core Points, Border Points, and Outlier.

Source: Density Based Clustering

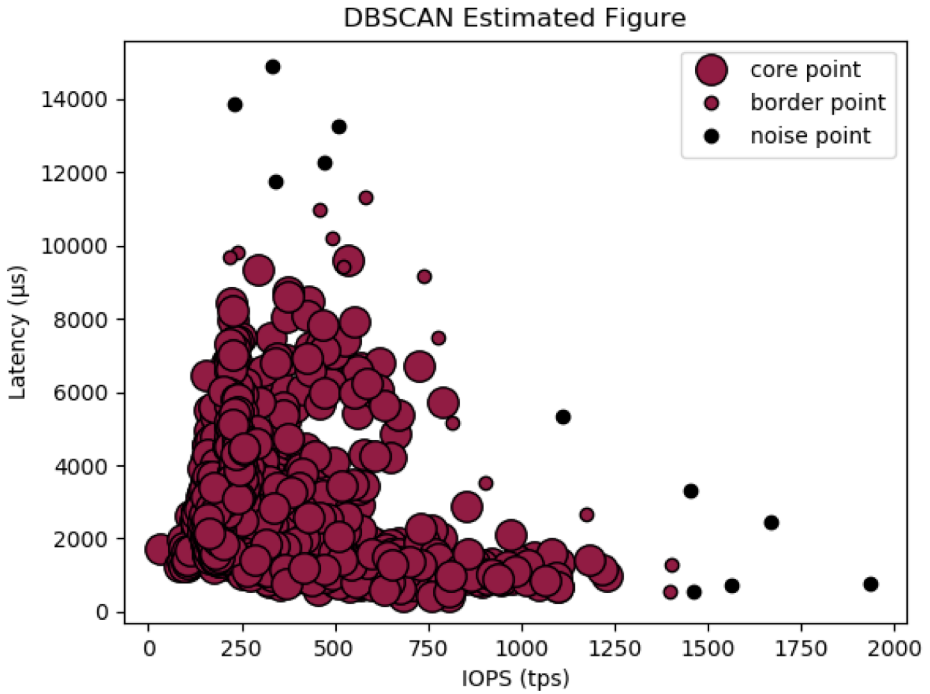
Gaussian Model Graph – LVM

- Volume level metrics
  - IOPs
  - Latency
- Data generation: used dd for random writes and bonnie++ for heavy workloads
- Data processing: 6000 data points collected in 2 days, removing zeros.
- mean: [259.76 3105.44]
- covariance: -16620.57
- epsilon: -41.80
- f1_score: 0.8



DBSCAN Graph – LVM

- Volume level metrics
  - IOPs
  - Latency
- Data generation: used dd for random writes and bonnie++ for heavy workloads
- Data processing: 6000 data points collected in 2 days, removing zeros
- epsilon: 1.40
- minPts: 11
- adjusted_rand_score: 0.69



POC Setup

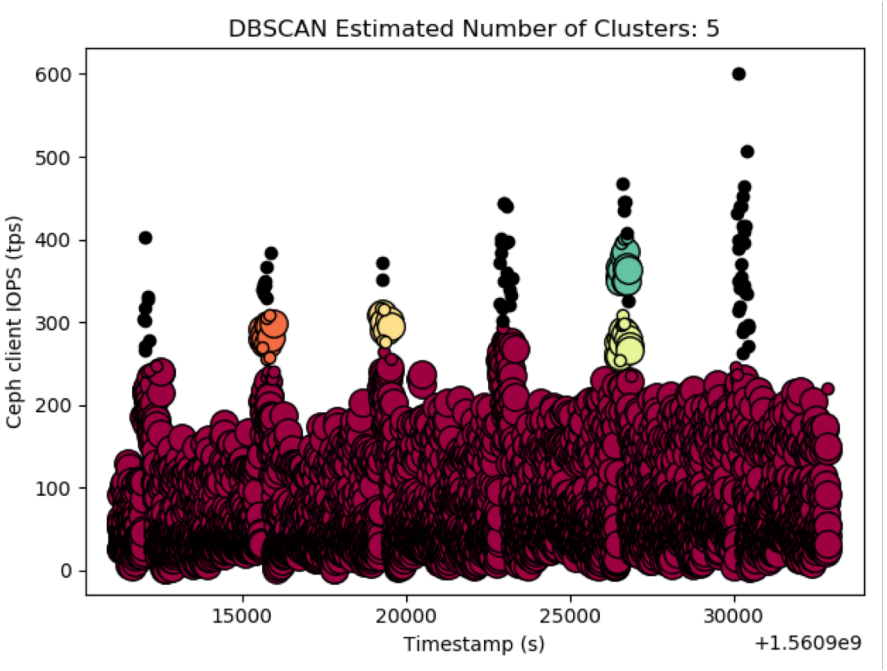
- MongoDB running on a volume provisioned by OpenSDS CSI plugin in Kubernetes environment with Ceph backend
- Collect metrics from Ceph, node-exporter, and MongoDB

```
……
volumeClaimTemplates:
  - metadata:
      name: mongodb-persistent-storage-claim
      annotations:
        volume.beta.kubernetes.io/storage-class: "csi-sc-opensdsplugin"
    spec:
      accessModes: [ "ReadWriteOnce" ]
      resources:
        requests:
          storage: 1Gi
…….
```

```
……
apiVersion: apps/v1beta1
kind: StatefulSet
metadata:
  name: mongod
spec:
  serviceName: mongodb-service
  replicas: 3
  template:
    metadata:
      labels:
        role: mongo
        environment: test
        replicaset: MainRepSet
        name: mongo
    spec:
……
```

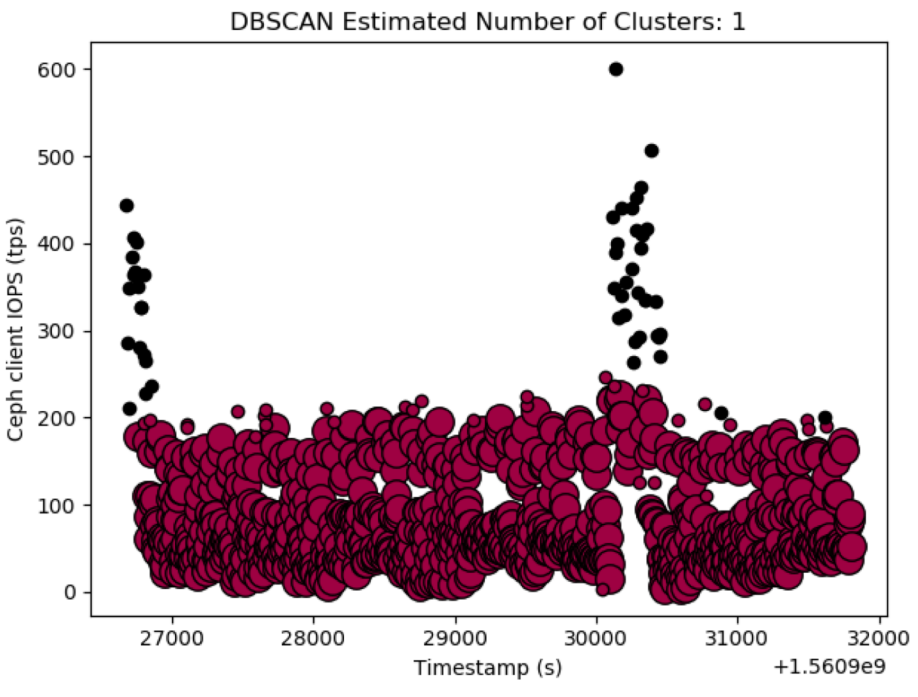
DBSCAN Graph - Ceph

- Ceph Client IOPs
- Data generation: MongoDB tools Workload Driver and MongoDB Multithreaded Performance Test Tool were used to generate MongoDB workloads (insert/delete/update/find)
- Data processing: 6000 data points collected in 6 hours, removing zeros
- epsilon: 0.30
- minPts: 10
- adjusted_rand_score: 0.77



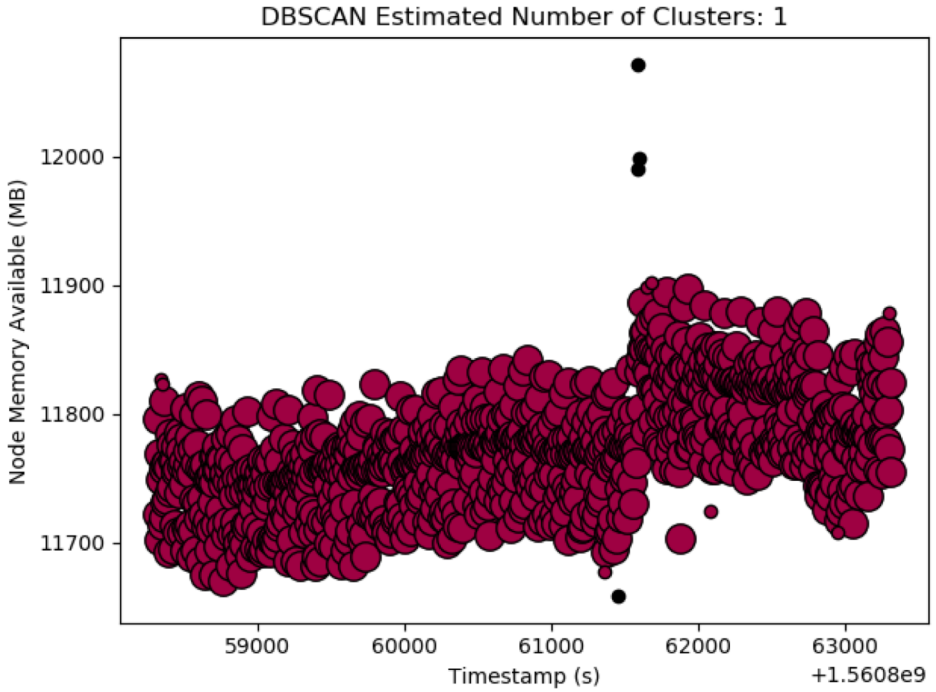
DBSCAN Graph - Ceph

- Ceph Client IOPs
- Data generation: MongoDB tools Workload Driver and MongoDB Multithreaded Performance Test Tool were used to generate MongoDB workloads (insert/delete/update/find)
- Data processing: 1000 data points collected in more than 1 hour, removing zeros
- epsilon: 0.30
- minPts: 10
- adjusted_rand_score: 0.84



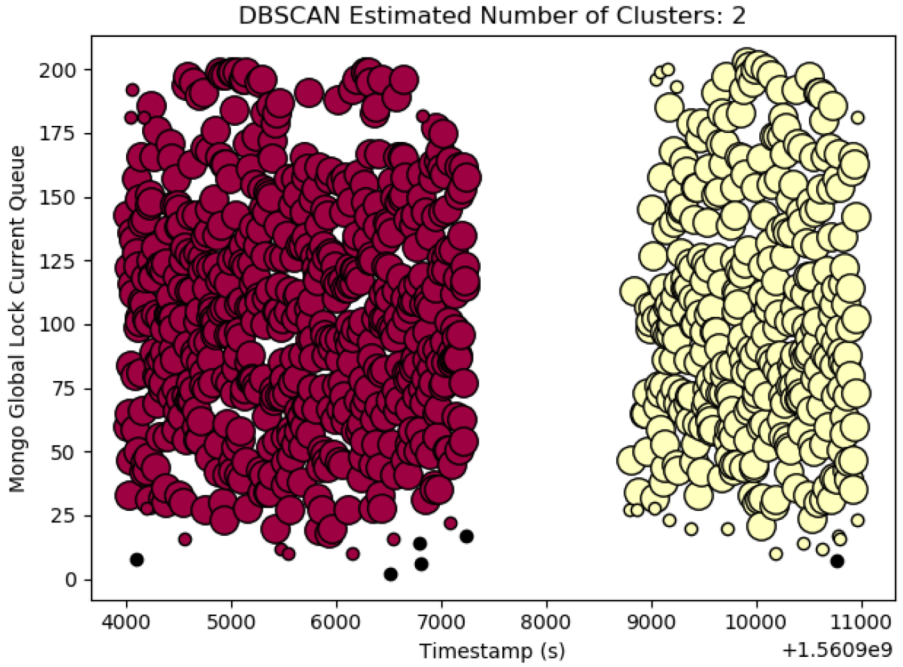
DBSCAN Graph - Node Exporter

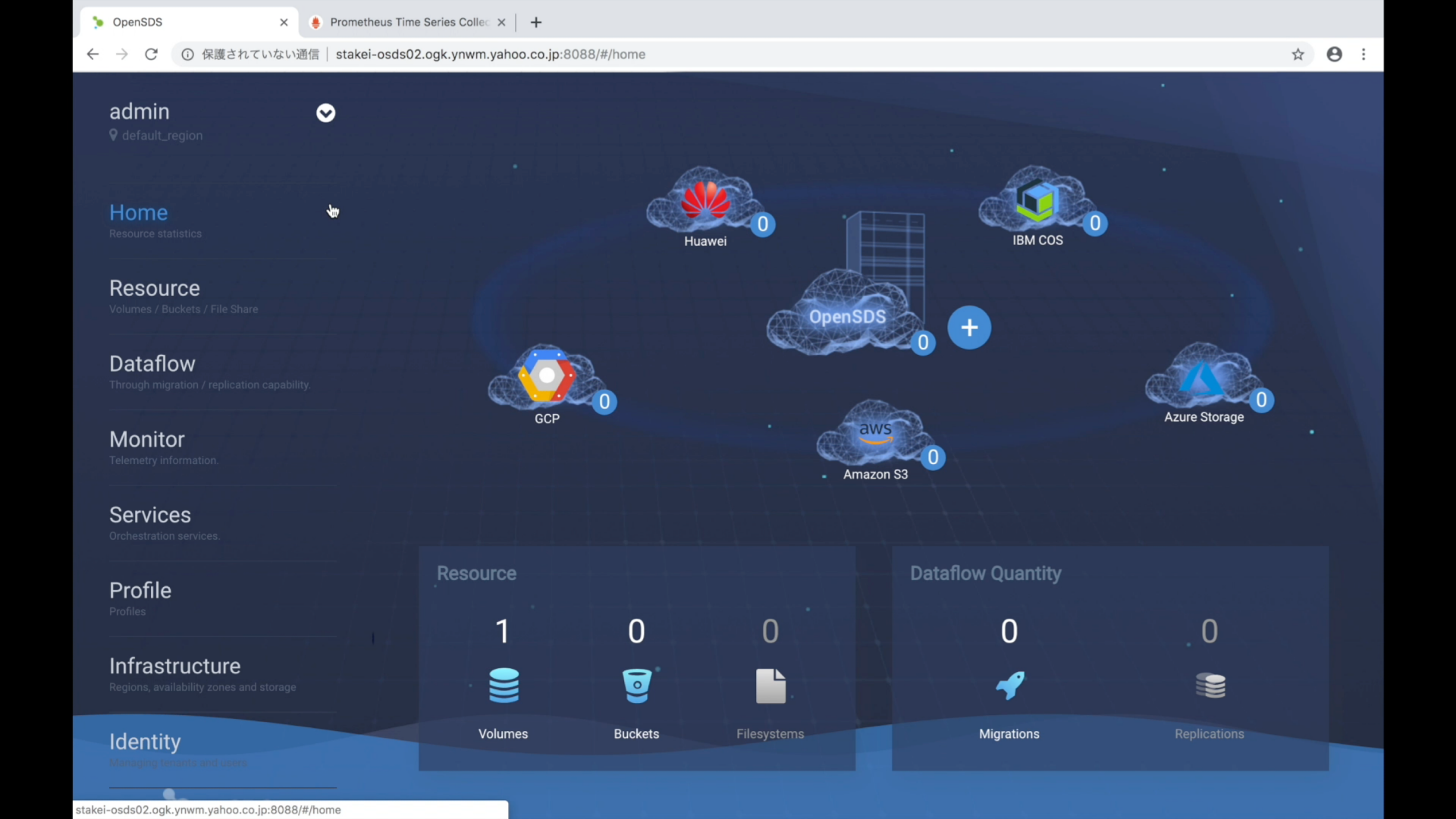
- Node Exporter
  - Node Memory Available (MB)
- Data generation: MongoDB tools Workload Driver and MongoDB Multithreaded Performance Test Tool were used to generate MongoDB workloads (insert/delete/update/find)
- Data processing: 1000 data points collected in more than 1 hour, removing zeros
- epsilon: 0.50
- minPts: 10
- adjusted_rand_score: 0.84



DBSCAN Graph - MongoDB

- MongoDB Global Lock Current Queue
- Data generation: MongoDB tools Workload Driver and MongoDB Multithreaded Performance Test Tool were used to generate MongoDB workloads (insert/delete/update/find)
- Data processing: 1000 data points collected in more than 1 hour, removing zeros
  - Gap between 2 clusters indicates a period with no data generated
- epsilon: 0.30
- minPts: 10
- adjusted_rand_score: 0.64

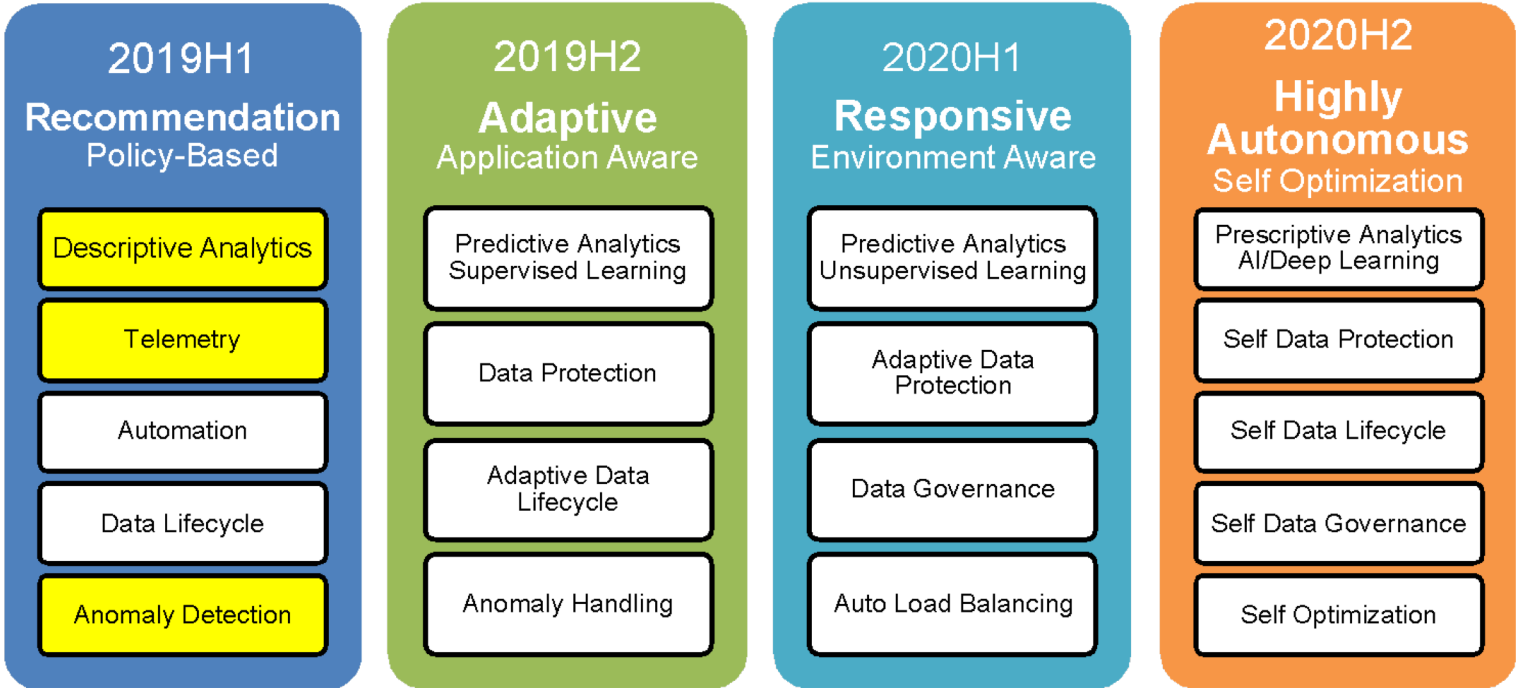



OpenSDS
Prometheus Time Series Colle
保護されていない通信 | stakei-osds02.ogk.ynwm.yahoo.co.jp:8088/#/home
admin
default_region
Home
Resource statistics
Resource
Volumes / Buckets / File Share
Dataflow
Through migration / replication capability.
Monitor
Telemetry information.
Services
Orchestration services.
Profile
Profiles
Infrastructure
Regions, availability zones and storage
Identity
Managing tenants and users
Huawei
0
IBM COS
0
OpenSDS
0
GCP
0
Azure Storage
0
aws
Amazon S3
0
Resource
1
Volumes
0
Buckets
0
Filesystems
Dataflow Quantity
0
Migrations
0
Replications
stakei-osds02.ogk.ynwm.yahoo.co.jp:8088/#/home

What's Next

- Collect more data ……
- Correlate storage metrics, node-exporter metrics, with performance issues in applications running on storage provisioned by OpenSDS in Kubernetes environment
- Other algorithms to consider
  - Random forest
  - ARIMA - AutoRegressive Integrated Moving Average
  - …...
- Continue the journey towards self-driving storage ...

The Road To Autonomous Data Platform



THANK YOU

OpenSDS

https://www.opensds.io

https://github.com/opensds

info@opensds.io

@opensds_io

opensds.slack.com

FIND OUT MORE

www.opensds.io

JOIN AS MEMBER

Accepting New Members

Vendors And End Users Welcome

OpenSDS


KubeCon
CloudNativeCon
OPEN SOURCE SUMMIT
China 2019