VERA C. RUBIN OBSERVATORY

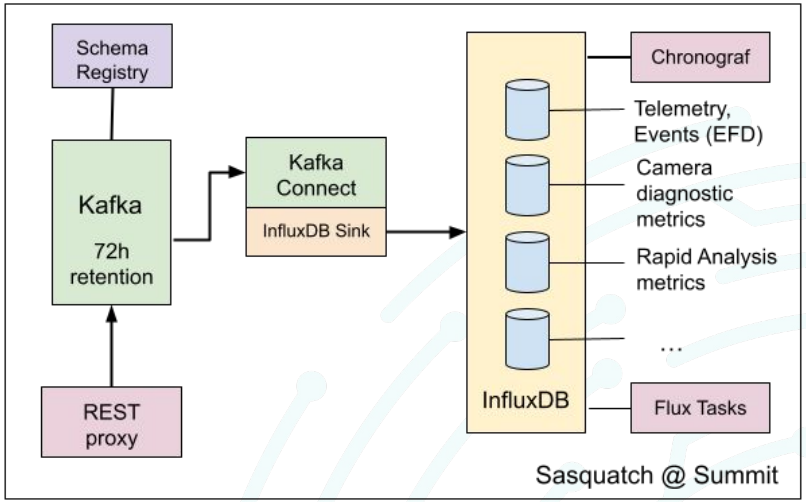
Sasquatch update

March 13 2023

Sasquatch

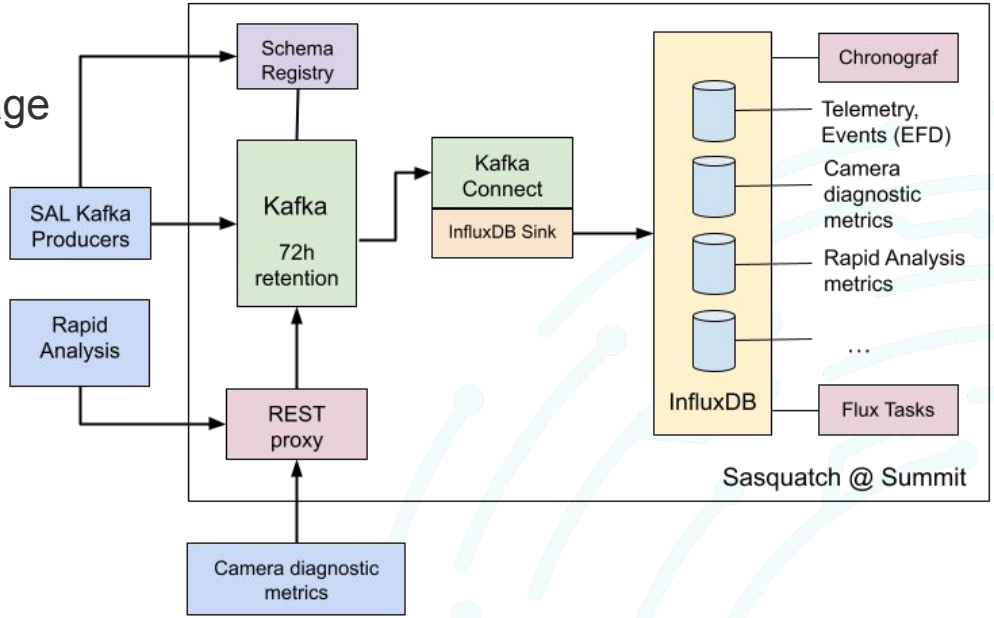
sqr-068.lsst.io

- An unified service for recording telemetry data and metrics
- Efficient storage and query of time-stamped data (InfluxDB)
- Scalability, fault tolerance, data replication (Kafka)
- Data exploration and visualization (Chronograf)
- Deployed via Phalanx at the Summit, TTS, BTS and USDF environments



Sending data to Sasquatch

- Messages in Avro format
  - It's fast
  - It has an extensible schema language
- CSCs use the SAL Kafka producers
  - With the replacement of DDS by Kafka CSCs will write directly to Kafka
- Sasquatch REST proxy (new!)
  - Replaces the SQuaSH API
  - Examples:
    i. Rapid Analysis/Prompt Processing metrics
    ii. Camera diagnostic metrics
    iii. External data



Sasquatch REST proxy

- Create a kafka topic for the metric “foo”

```
sasquatch_rest_proxy_url = "https://usdf-rsp.slac.stanford.edu/sasquatch-rest-proxy"

topic_config = {
        "topic_name": "foo",
        "partitions_count": 1,
        "replication_factor": 3
    }

headers = {"content-type": "application/json"}

requests.post(f"{sasquatch_rest_proxy_url}/v3/clusters/{cluster_id}/topics", json=topic_config, headers=headers)
```

- The metric “foo” becomes an endpoint in the REST proxy API

```
metric_url = "https://usdf-rsp.slac.stanford.edu/sasquatch-rest-proxy/foo"
```

Sasquatch REST proxy

- Send a message for metric “foo” with an embedded schema
- Built-in integration with the Schema Registry

```
data = {
    "value_schema": "{\"type\": \"record\", \"name\": \"foo\", \"fields\": [{\"name\": \"bar\", \"type\": \"float\"}]}",
    "records": [
        {
            "value": {
                "bar": 1.0
            }
        }
    ]
}

headers = {"content-type": "application/vnd.kafka.avro.v2+json"}

r = requests.post("https://usdf-rsp.slac.stanford.edu/sasquatch-rest-proxy/topics/foo", json=data, headers=headers)
```

Analysis metrics

DM-37898

- Proposal: create a butler data store for Sasquatch
  - `butler.put('objectTableCore_metrics')`
- Metric records: metadata + metric measurements

```
record = {
    'id': 'eedb4b95-162f-4ce9-93e9-a4615bc03887',
    'run': 'LSSTCam-imSim/runs/ci_imsim/20220926T170001Z',
    'timestamp': '20220926T170001Z',
    'dataset_type': 'objectTableCore_metrics',
    'skymap': 'discrete/ci_imsim/4k',
    'tract': 0,
    'u_medianSky': -14.256834375755528,
    'u_meanSky': -23.56414638713267,
    'u_stdevSky': 300.82351725258707,
    'u_sigmaMADSky': 315.6434611660024,
}
```

Metadata from the DatasetRef

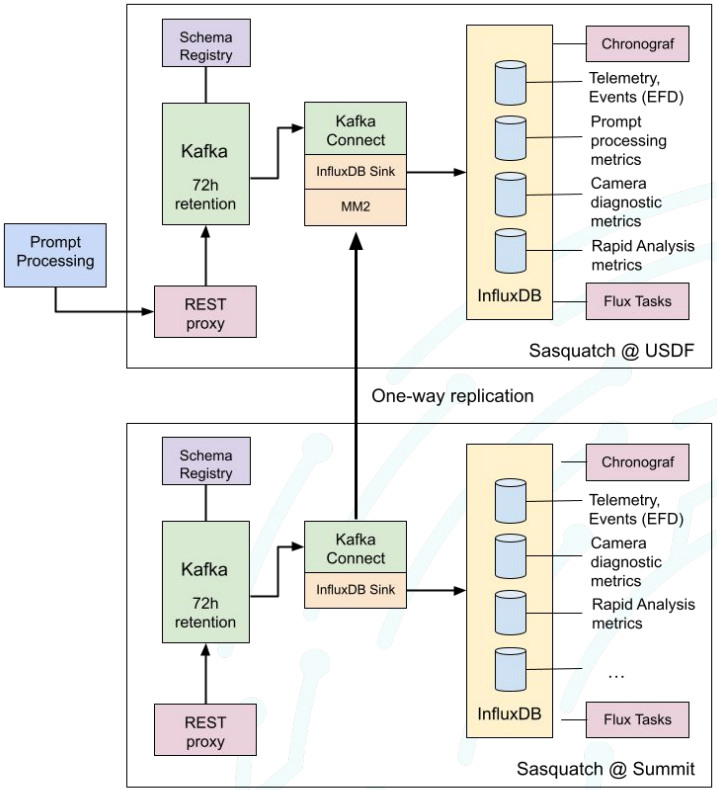
Metric measurements

- Auto generate the Avro schemas

Data replication

sqr-050.lsst.io

- Summit ⟶ USDF
  - It's finally working over the LHN!
  - ~6.5MB/s (or ~500GB over 24h)
  - Restoring past data manually (ongoing)
  - Schema Registry integrity at USDF
- BTS ⟶ USDF dev
- TTS ⟶ USDF dev (coming soon!)
- Two-way replication (coming soon!)
  - E.g. Prompt processing metrics at USDF replicated to the Summit



Migration to InfluxDB 2.x

- The InfluxDB Sink connector does not support InfluxDB 2.x
- Alternative: Telegraf Kafka consumer
  - Upstream an Avro parser plugin (thanks Adam!)
  - Running on TTS and performing 50-100x faster (VMS data!)
- UI improvements
  - New visualization types (Histogram, Heatmaps, Scatter plots)
  - Labels to group dashboards
  - New data Explore UI
- InfluxDB IOx OSS will add native SQL support
- Kapacitor replaced by InfluxDB tasks
- New InfluxDB OSS 2.x Python client