SEI's Experiences with Deploying Stream Technologies

Brad Powell

John Stogoski

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Document Markings

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM20-1025

Introduction of Team

Who we are:

We are a team of engineers focused on developing, testing, and evaluating technologies and design patterns.

Reside within Security Automation, a CERT Division directorate, with a focus on cybersecurity use cases.

Operate a hybrid lab environment including a dedicated data center and cloud services.

Develop trend reports to examine emerging technologies and shifts in the open source community and commercial marketplace.

Scope of efforts:

Manage environment including logging, monitoring, and analyzing operational data.

Develop prototypes integrating various technologies to assess capabilities and design implications.

Use traffic generator and other simulation techniques to evaluate functionality and scalability.

Background on Streaming Applications

Characteristics

- Data produced from a variety of sources containing “small” record sizes on a “frequent” basis
- Near real-time expectations
- Data needs to be routed to multiple destinations or potential for new destinations over time

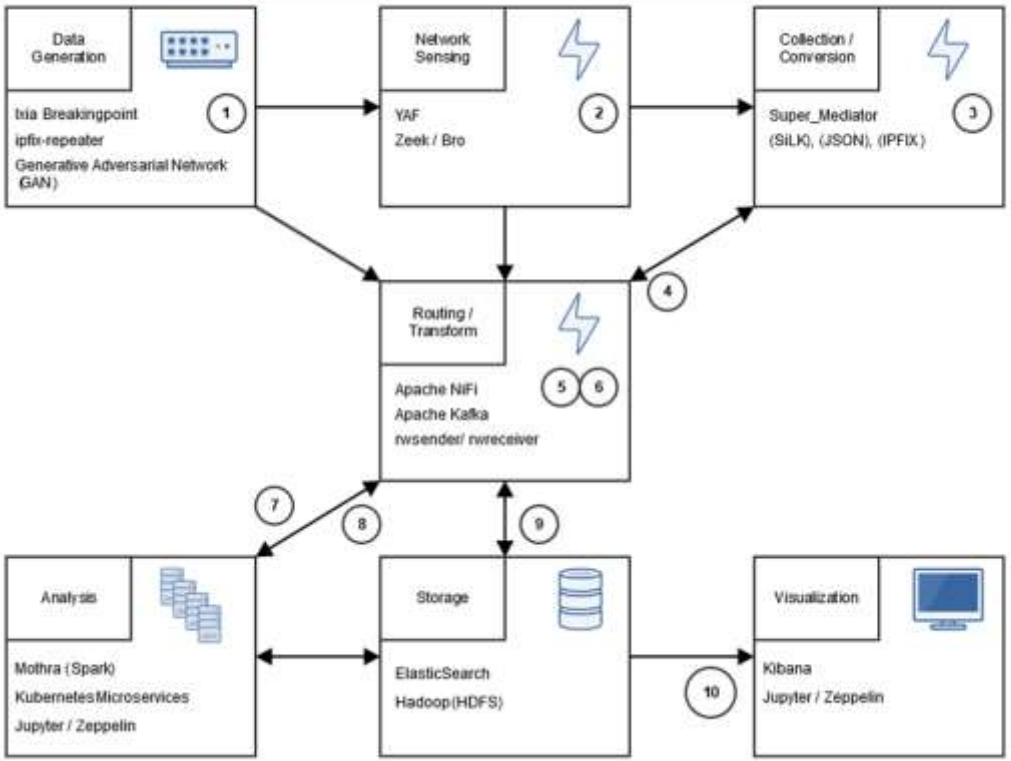
Use Cases

- Alerting (thresholds, watchlists, etc.)
- Data transformation
- Enrichment (adding meta-data)
- Inline analysis (statistics, sliding windows, abnormalities, etc.)
- Situational awareness dashboards

Technologies

- Modular microservice architectures
- Data center – cloud services (AWS)
- Automated Testing and Deployment
- Containers – Kubernetes, Docker
- Data Movement - Logstash, Kafka, Apache NiFi
- Destinations (HDFS, Elasticsearch, S3, …)
- Visualization (Kibana, Graphana, Tableau, Power BI)
- Application Programming Interfaces (APIs)
- Progression of capability (Redis -> Storm -> Kafka -> Apache NiFi -> Serverless)

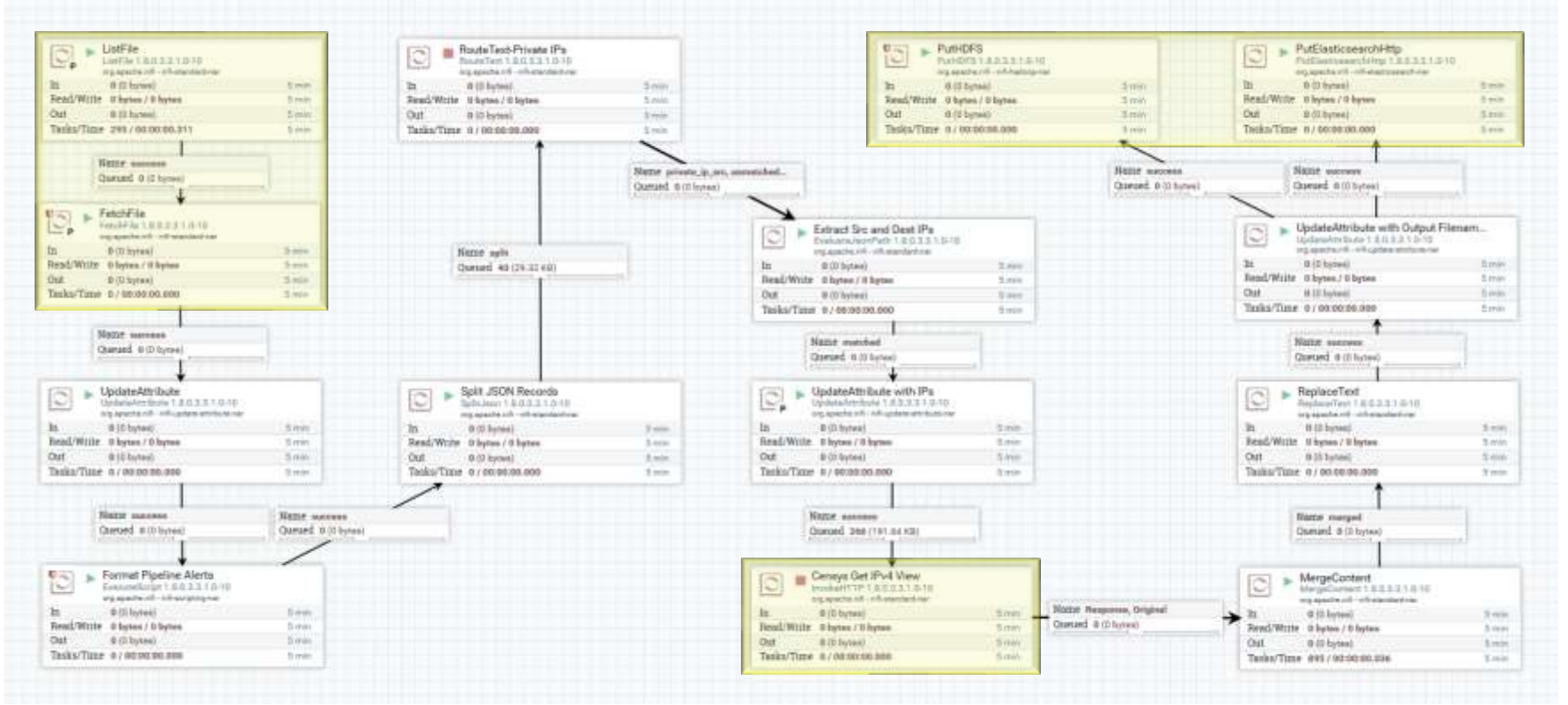
System Architecture Example



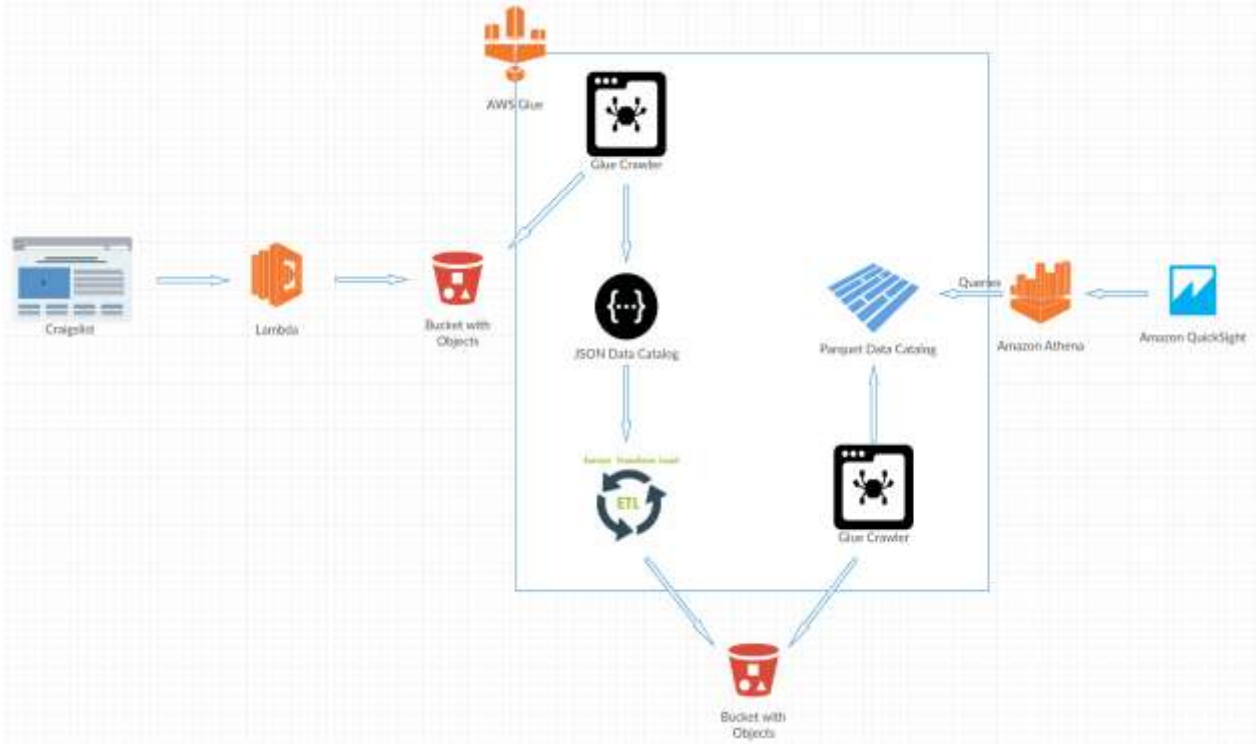
DNS Resolver Example:

1. The Ixia traffic generator is used to produce simulated network traffic.
2. Ixia is connected to a server running YAF which captures NetFlow and produces an IPFIX output.
3. Super_Mediator picks up the files and converts IPFIX to JSON output.
4. Apache NiFi is watching the output directory and fetches new incoming files.
5. Apache NiFi filters out DNS traffic where silkAppLabel = 53.
6. Apache NiFi writes a JSON message for each DNS flow to a Kafka topic.
7. A Python script running as a microservice on Kubernetes is subscribed to the Kafka topic and processes records.
8. The microservice writes Bad Resolvers to a new Kafka topic.
9. Apache NiFi is watching the bad resolvers kafka topic and routes the JSON records to ElasticSearch.
10. A Kibana dashboard connected to ElasticSearch visualizes the results in real time for analysts.

Apache NiFi Streaming Data Enrichment Example



Emerging Cloud & Serverless Opportunities



Example Technologies

- Lambda (Serverless)
- Containers (Docker / Kubernetes)
- AWS Glue / AWS Data Pipeline
- Messaging / Kinesis
- Azure / GCP
- Infrastructure as Code Terraform, Ansible, CloudFormation, Step Functions

https://medium.com/i-like-big-data-and-i-cannot-lie/serverless-data-engineering-aws-glue-lambda-athena-quicksight-de3ef177884f

Design Considerations

- Training and organization structure
- Capital vs. expense costs
- Data movement (cloud challenges eg. snowball, network latency)
- Metrics and monitoring
- Available resources (eg. on-premise vs. cloud, commercial vs. gov, scale out vs. up)
- Timeliness (avoid delays and complex processing, utilize parallel processing)
- Data types (eg. binary, text, json, parquet)
- Testing (eg. simulating data at scale, using automation)
- Microservices enable modularity and collaboration
- CI/CD and containers speed up deployment
- Using cloud efficiencies on-premise