Lessons Learned Building and Operating a Serverless Data Pipeline

Will Norman
ALRT #15630 on Vertica: vertica node down

738-89
ALRT #15630 on Vertica: vertica node down
Reply 4: Ack, 6: Resolv
Introduction

- Will Norman - Director of Engineering @ Intent
  - FinTech and AdTech background

- Intent
  - Data Science company for commerce sites
  - Primary application is an ad network for travel sites
MOD Squad

- MOD owns data
- 4 Engineers
- 1 Product Manager
What we’ll be covering

- What is Serverless?
- Intent Data Platform
- Lessons Learned
What is Serverless?

- More about managed services than lack servers
- Not just FaaS
- Scale on demand / pay for only what you use
- Empowers developers to own their platform
Intent Data Platform [Old World]

- Active MQ
- Log Processors
  - Java applications
  - Kept state locally
  - Cron scheduled tasks to roll files to S3
  - Ran on dedicated EC2 instances
- S3
Intent Data Platform [New World]

- Kinesis
- Lambda
- Kinesis Firehose
- SNS
- AWS Batch
- S3
Data Consumers

- Streaming Data Consumers
- Spark Jobs / Aggregations -> Redshift
- Snowflake Loader -> Snowflake
- Parqour -> Athena
  - EMR based jobs that convert AVRO -> Parquet
Worth the move?

- Fewer production issues
- Separation of concerns
- Horizontally scalable
- Removed a lot of undifferentiated heavy lifting
Lessons Learned

1. Total Cost of Ownership
2. Think about data formats upfront
3. Design for Failure
4. Design for Scalability
5. Not NoOps just DiffOps
6. Build Components
7. CI / CD Strategies
8. Leverage the Community
Total Cost of Ownership

- On demand costs
- Hidden Costs / Tag All The Things!
- Enterprise Support
- Value of being able to focus on core business problems
Think about data formats up front

- What does the ecosystem support?
- Schema vs Schemaless (eg AVRO vs JSON)
- Data validation & Data evolution
- Data at rest vs data in flight
- JSON / CSV / AVRO / Parquet?
Schema Registry

```
record DataWrapper {
    string dataType;
    long schemaFingerprint;
    bytes data;
}
```

- Publish Schema in JSON format to S3
- Consumers lookup schemas, and calculate fingerprints
Design for failure

- System Guarantees?
- Idempotency
- Over process (data lookbacks)
- Dead Letter Queues
Design for Scalability

- Decouple from non-scalable systems
- Don’t run lambdas in VPC if you can help it
- Partition data at rest
- Shard events based on GUID / random id if ordering isn’t necessary
- Think about fan out patterns
Not NoOps, just DiffOps

- Application problem or service problem
- Platform Limits
- Logs
- Metrics
- Dashboards
- Alerts
Some things remain the same
HAVE YOU TRIED TURNING IT OFF AND ON AGAIN?
Build Components

- Help to reason about different parts of the system
- Make it easy to do the right thing
- Easier to extend
- Infrastructure as Code
module "conversion_event_processor" {
    source = "../modules/event_processor"
data_type = "conversion"
data_source = "ad_server"
processor_lambda_handler = "com.intentmedia.data.stream.ConversionLambda::handler"
environment = "${var.environment}"
firehose_lambda_handler = "com.intentmedia.data.stream.ConversionFirehose::handler"
processor_lambda_reserved_concurrent_executions = 3
firehose_lambda_reserved_concurrent_executions = 2
}"
CI / CD

- Step backwards from being able to run stack locally
- Unit tests for business logic
- Integration Tests / End to End tests to ensure that everything is working as expected
- Use different AWS accounts to segregate staging and production
Leverage the Community

- **Slack**
  - Serverless Forum
  - og-aws

- **Blogs**
  - Symphonia [https://www.symphonia.io/](https://www.symphonia.io/)
  - Charity Majors [https://charity.wtf/](https://charity.wtf/)
  - Jeremy Daly [https://www.jeremydaly.com/](https://www.jeremydaly.com/)

- **Twitter**

- **Meetup Events / Conferences**
Questions?

Will Norman
will.norman@intent.com

We’re hiring!