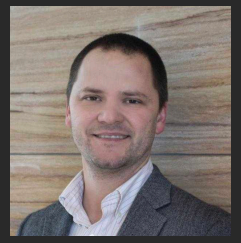
Elastic Cloud Enterprise in Azure

A Devon Energy Story

NYSE: DVN
devonenergy.com

Who Are We?



Paul PC

- Alphabet soup: MBA, GSE, GREM, GCIA, GCIH, GSEC, GPEN, GPYC, CISSP
- Security Architect / Team Lead for Devon Energy, an Independent E&P headquartered in OKC
- Fluent in Romanian, English, Python
- Loves the cloud almost as much as his Ducati

Prakhar S

- Educational Kaizen : Bachelor of Engineering (CSE), RHCE, SAS® Certified
- Works for Accenture Technology, collaborating with Devon Energy currently
- Close to nine years of experience, aspires to contribute in cutting edge tech

Agenda / History

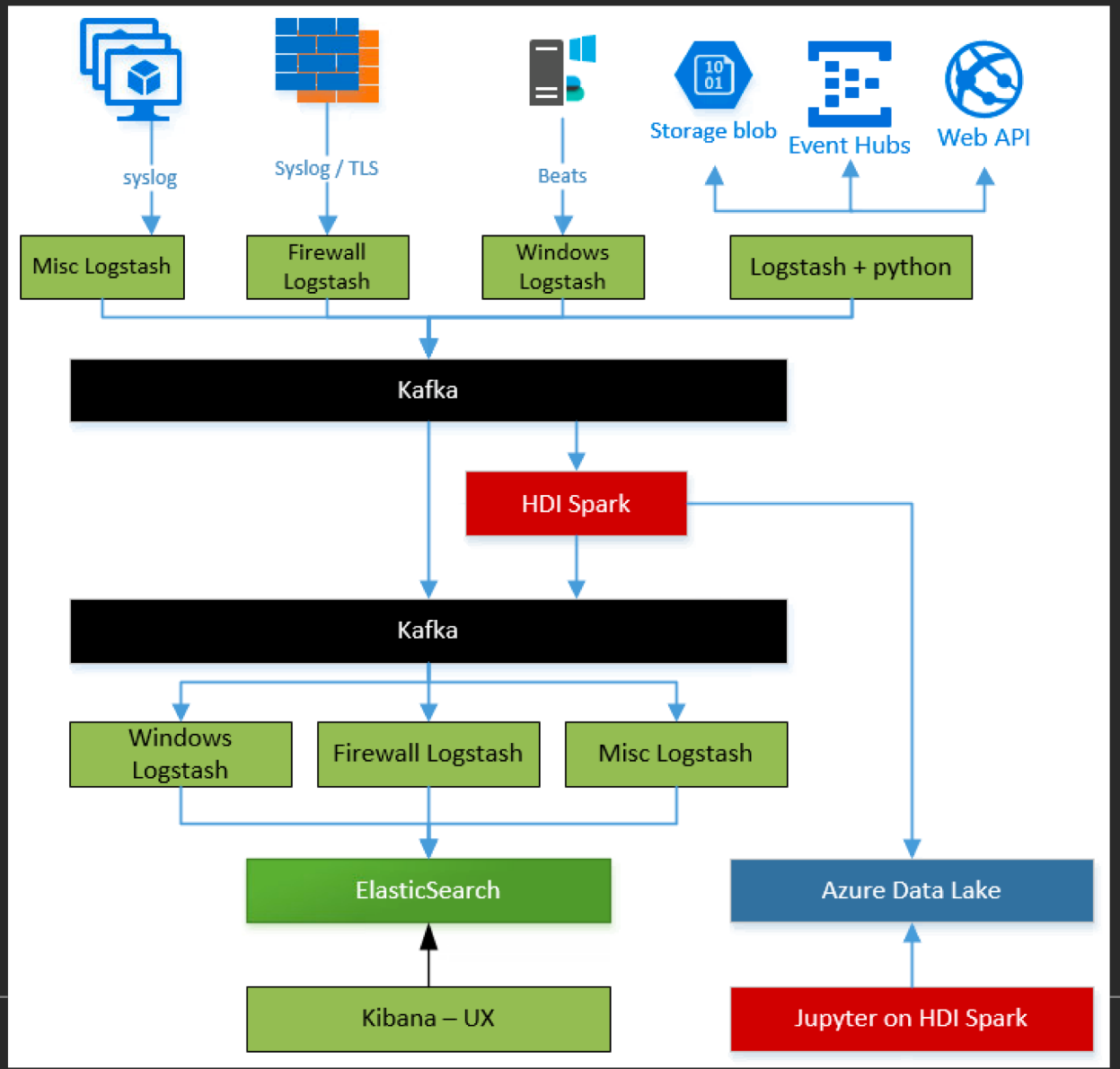
2013: ES 1.7 and Moloch

2015: Production-grade Servers

2017: Elastic Search in Security Analytics Platform

2019: SIEM replacement

2017: Analytics Platform v1.0 – Working Experiments



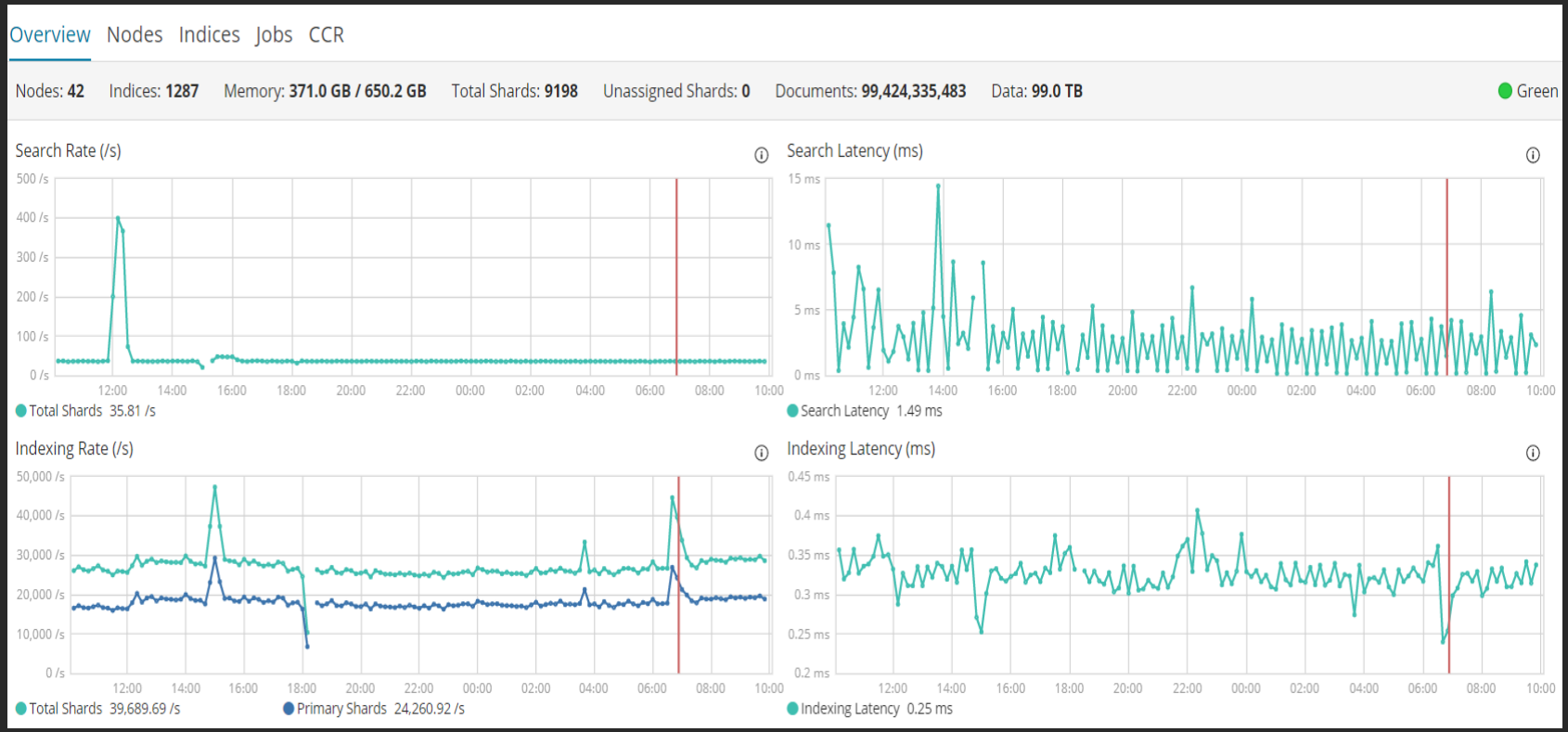
Axioms:

- Collaboration with Data Analytics team
- Only Cloud components
- Use as much azure 1st party as possible
- Be able to ingest 500GB/day

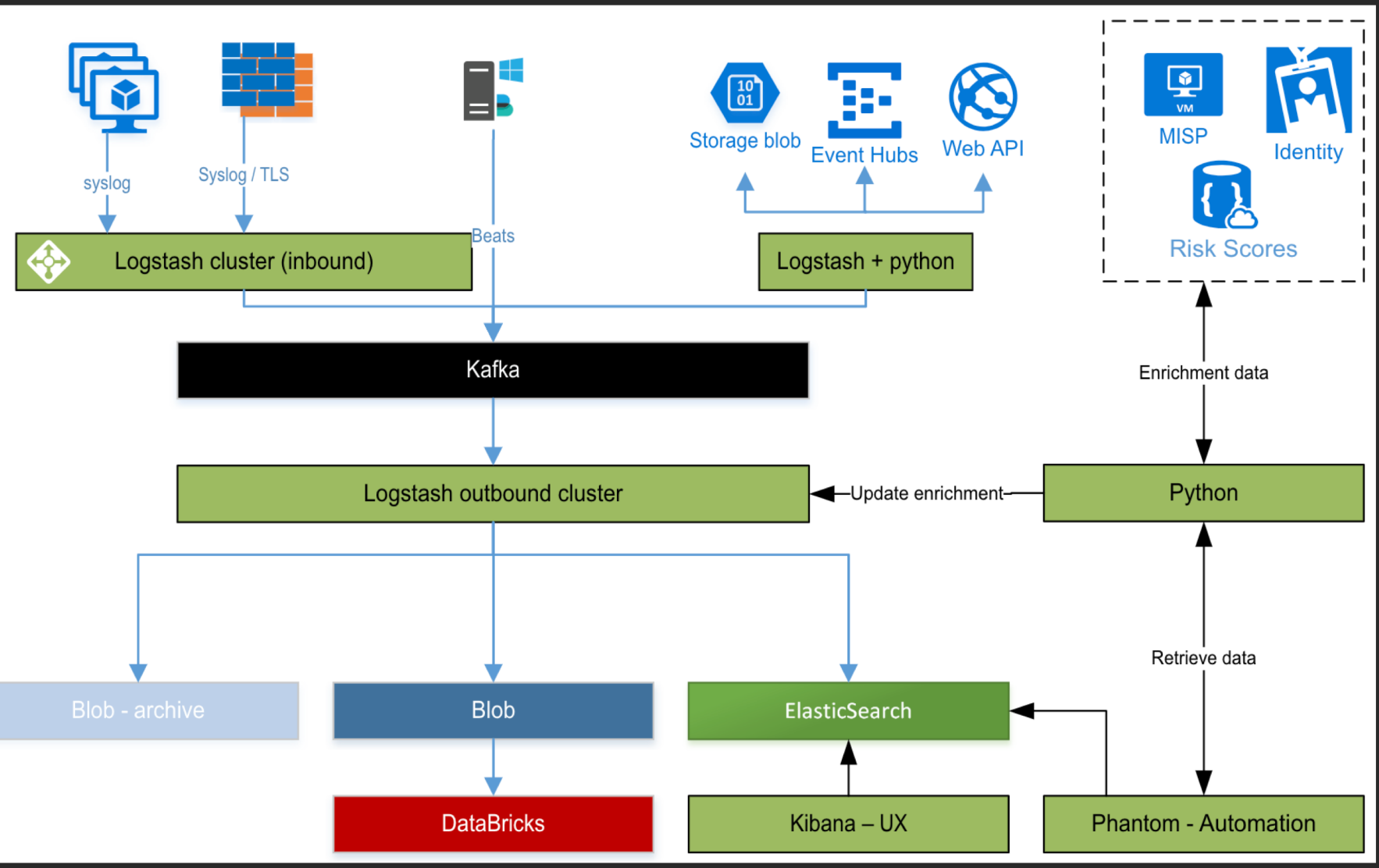
Scale

Approximately

- 100 billion documents
- 100 TB of Indexed data
- 1.5 – 2.0 TB DAILY
- 1300 indices
- 10000 shards
- Ingestion rate approximately 20-25k EPS



2019: Analytics Platform v3.0 – Production SIEM++



Enrichment moved to Logstash

DYI Retention to BLOB

First shard optimizations efforts

Automation is our best L1 HoD

More data, more user stories

- Network
- AKS
- Accounting
- SAP security

Logstash - The Core of Our FrankenSIEM

- Traditional SIEM problems:
  - Retention
  - Parsing - legacy logs are terrible
  - WEF is great, until it isn't
- Just enough normalization
  - Enrichment > Correlation
  - Atomic indicators
  - Identity information
- Cluster > purpose collectors
  - Logstash nodes are cattle, not pets
  - Puppet was good, K8s is great
  - Throughput vs. node count

```
if [username] {
    translate {
        id => "user_enrich_uname"
        field => "username"
        dictionary_path => "/enrichment/users.yml"
        destination => "user"
        add_tag => "user_enrichment_success"
    }
}
```

Why ECE

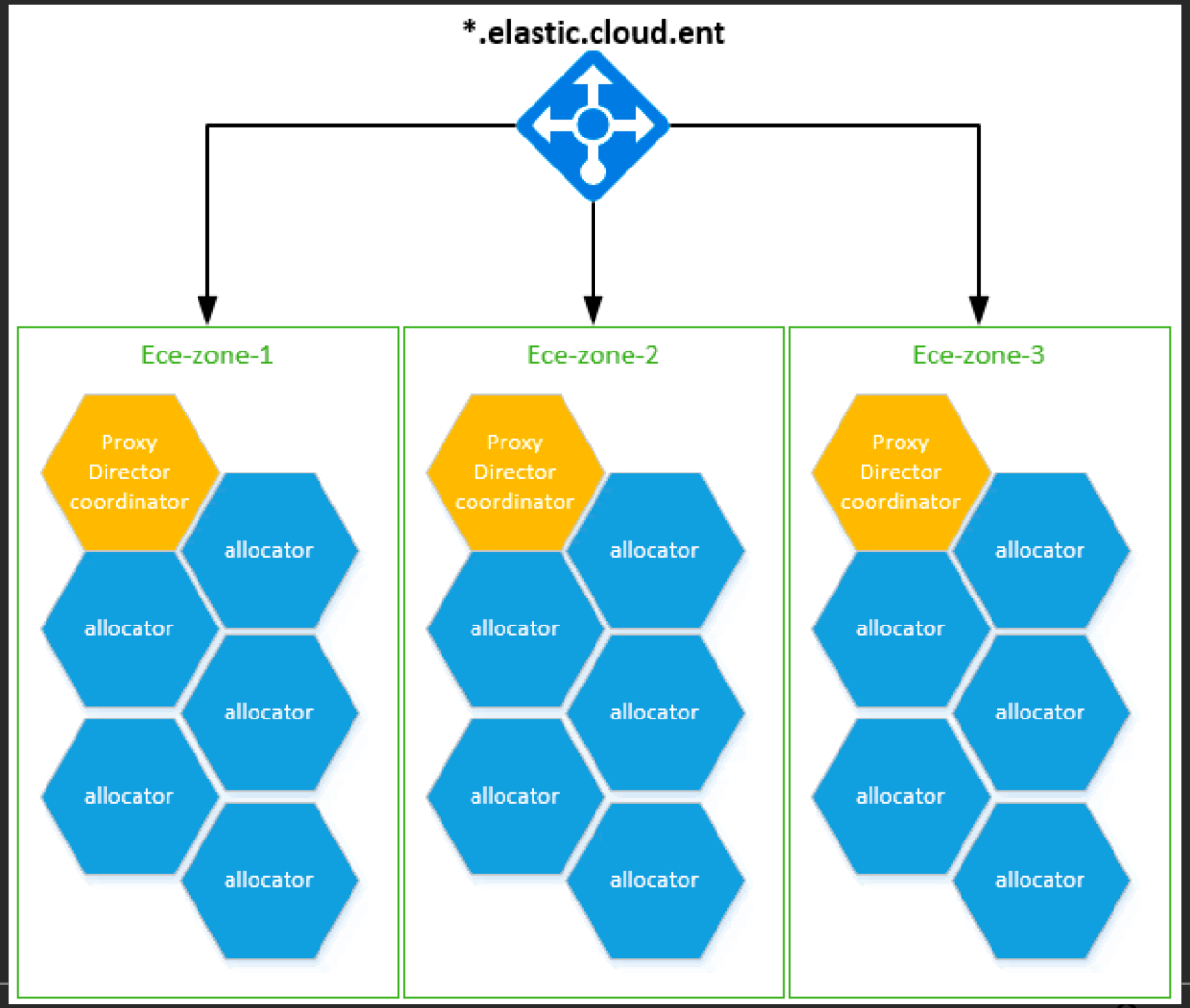
Intent : Security should not be managing it

What's cool about ECE

- It saves Time : Streamline scaling, securing, upgrading the stack
- Great way to enhance utility and value was to grant other teams access to their logs
- Centralized management of logstash pipelines and cluster user settings is cool
- Inbuilt monitoring gives a great insight

It was very different than traditional ES

- Troubleshooting is a bit different
- Elasticsearch.yml is not to be trifled with
- Initially, had to rely a lot more on support



Major Gotchas

Indexing throughput

- Use dedicated Master Nodes
- Index refresh interval refresh_interval : 30s (even -1 for initial load)
- Tune Indexing buffer size

OOMs

- Give JVM 50% of available memory
- Prevent JVM resizing : minHeap=maxHeap

Cautiously budget your cache

- Limit and monitor Field Data cache : it never goes away
- Using circuit breaker can save the day

Disk Sizing

- Monitor number of replica for your use case
- Experiment with Sharding : Larger shards means better indexing rate but needs can vary; more shards come with cost

Appraise your precedence, go *Quid-pro-quo*

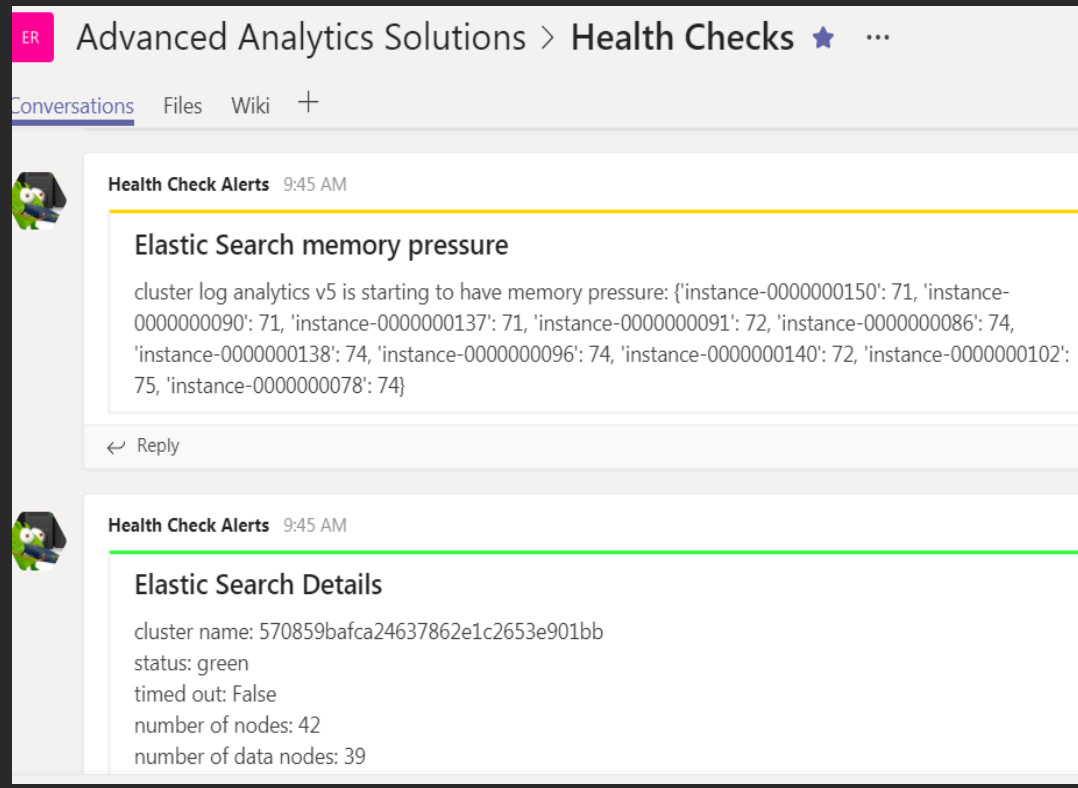
- Memory Intensive queries vs near-real-time data vs long-term retention

What We Learned – Operational Glitches

- It hurt being the first Azure customers for ECE
  - Docker ignorance didn't help
  - ECE 1.1.x supported specific version of docker with specific version of kernel
  - Updates tanked the cluster at-least twice – multi zone helps

- **Problems** at our scale:
  - Migrating shards between containers takes days
  - In-place container upgrades:
    advanced configuration will save your day/week
  - Moving to dedicated master nodes – election problems
  - Unexpected socket hang ups, difficult to diagnose

- Monitoring:
  - Independent Monitoring cluster
  - Created our own health check scripts



What's Next For Us

Chugging the Docker and Kubernetes Kool-Aid:

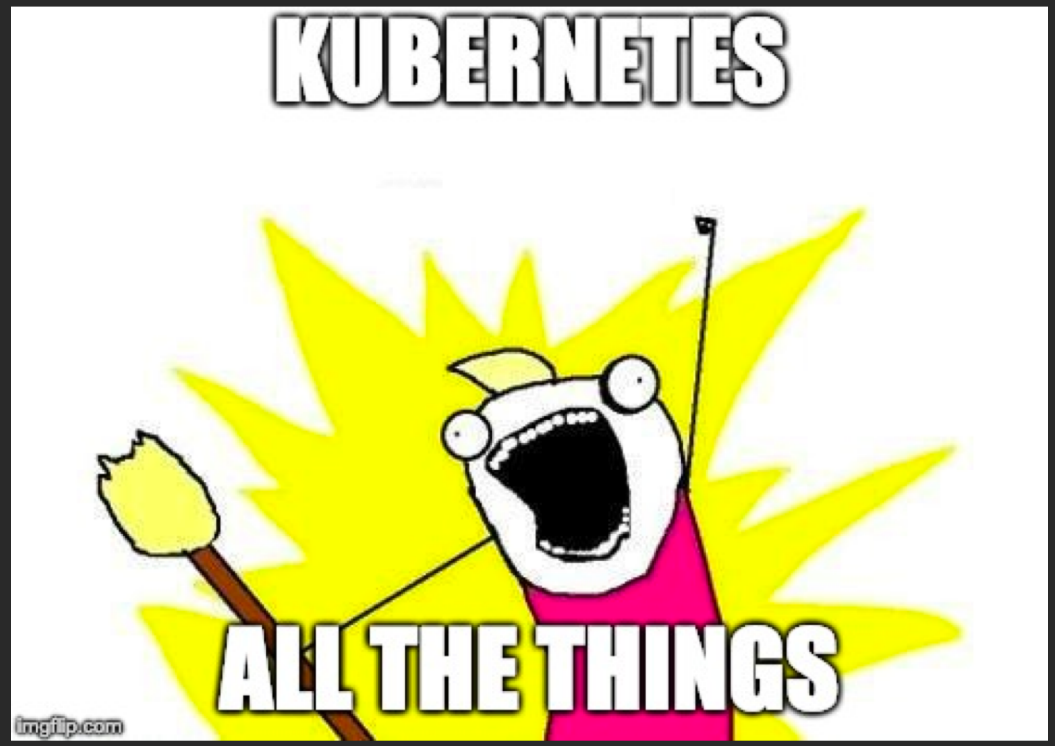
- Curator
- Enrichment data collection
- Logstash + autoscaling
- Health checks

Dedicated nodes:

- ML Nodes
- Search Only Nodes
- Ingest nodes

Separate security and non-security to their own spaces

SAML / OpenIDc with MFA



Come call BS at Happy Hour

Contact info:
@p4ulpc – paul.poputa-clean@dvn.com
Prakhar.sengar@dvn.com

Code:
https://github.com/paulpc/elasticon2019

devon