
AWS
re:Invent

ANT303

Modernize log analytics with Amazon Elasticsearch Service

Seth Carroll
Principal DevOps Engineer
Capital One

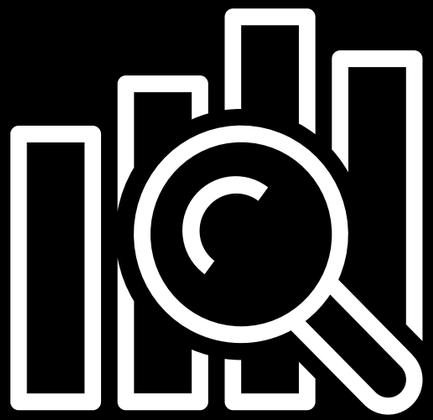
Kumar Anand
Lead Site Reliability Engineer
Capital One

Kevin Fallis
Principal Analytics Specialist Solutions Architect
AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Elasticsearch Service



Amazon Elasticsearch Service (Amazon ES) is a fully managed service that makes it easy to deploy, manage, and run Elasticsearch cost effectively with industry-leading reliability, scalability, and security

Amazon ES: Capabilities and benefits



- Fully managed



- Direct access to the Elasticsearch APIs



- Cost effective, no licensing fees for features



- Designed for zero downtime; scale without interruption



- Secure and compliant

Important Amazon ES features for modernizing log analytics



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

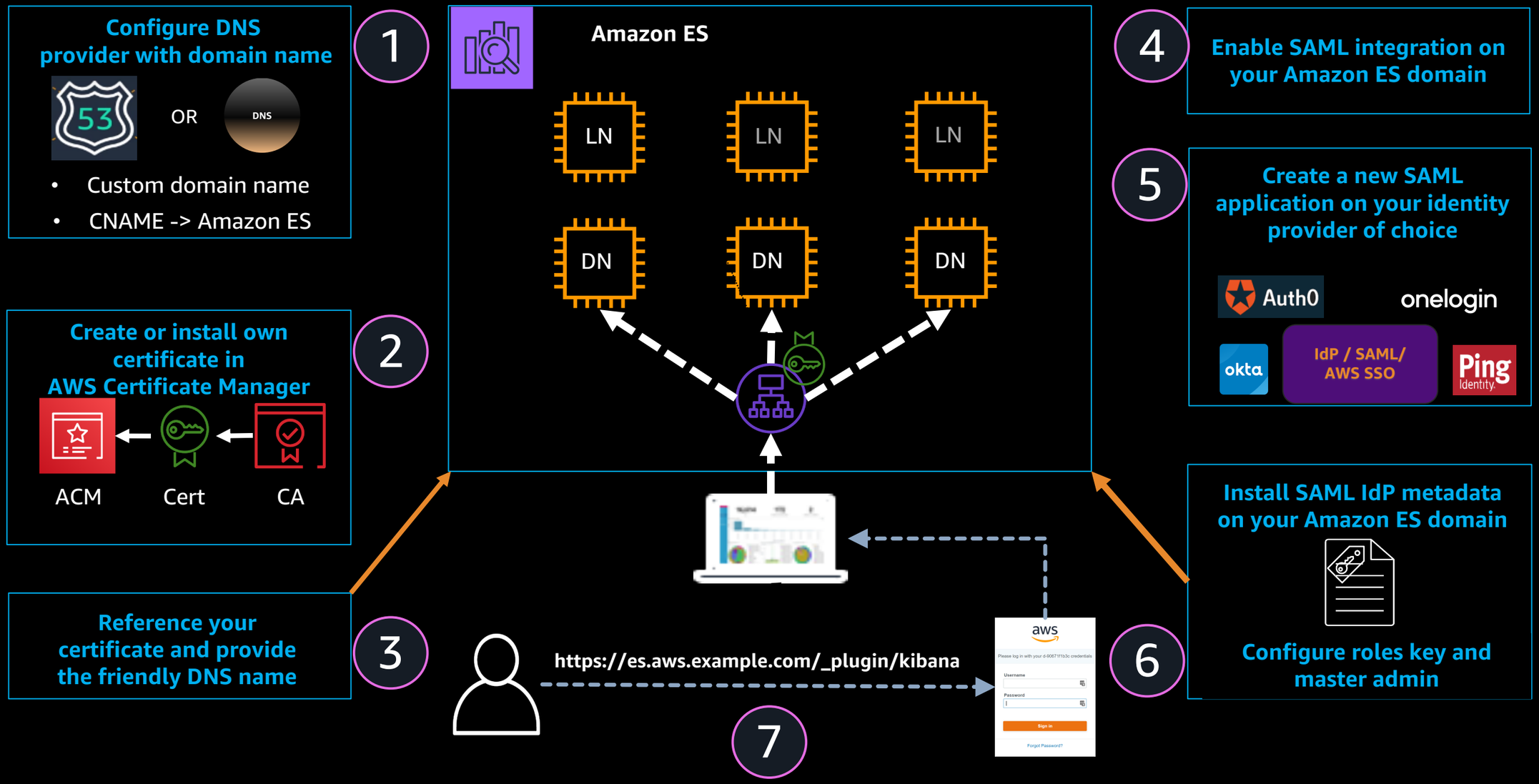
Cross-cluster search for Amazon ES

INCREASE SCALABILITY, EFFICIENCY & AVAILABILITY, BY SEPARATING DISTINCT WORKLOADS



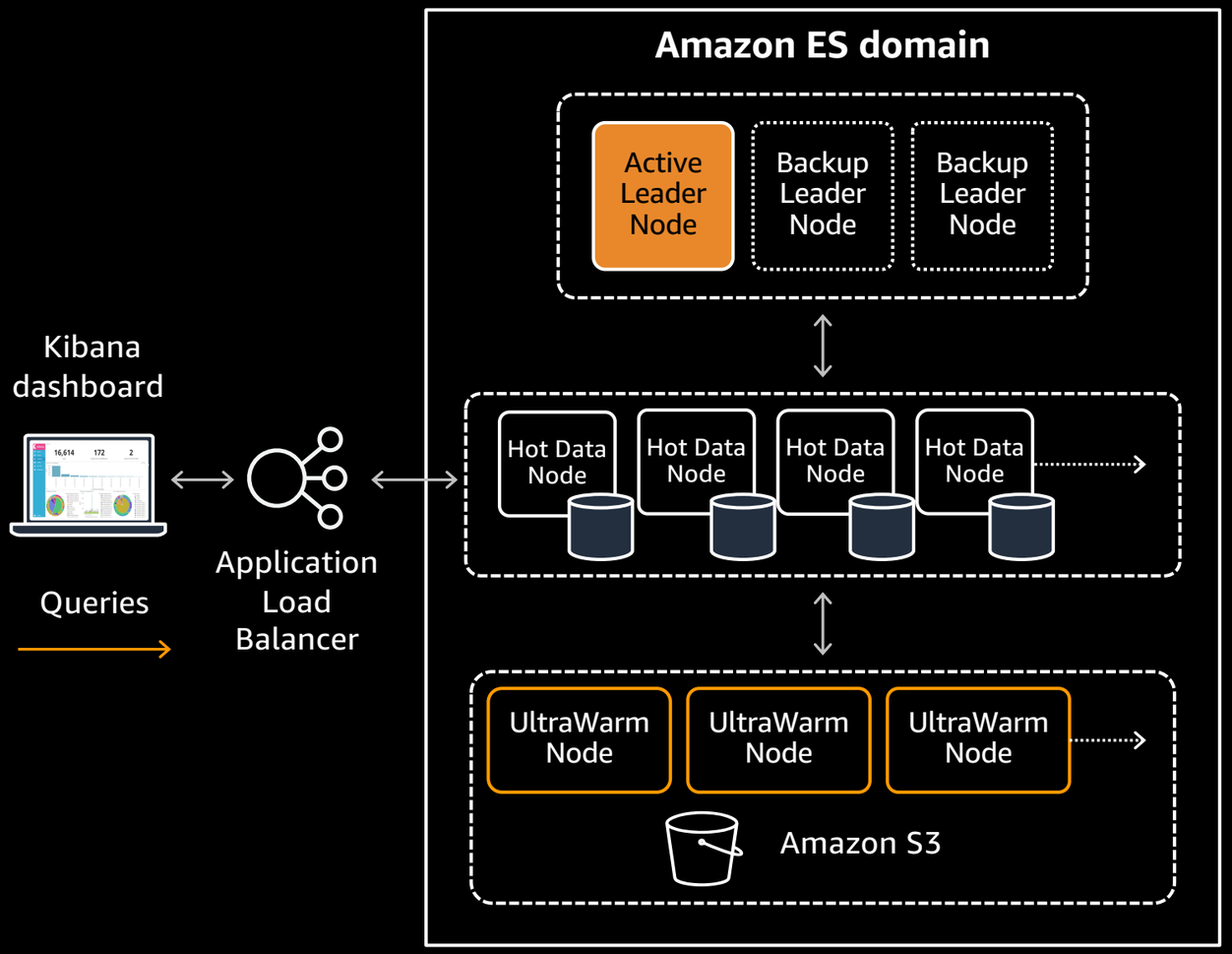
- Single Kibana interface to search across all included domains
- Tune domain resources for specific workloads
- Isolate failures to specific workloads

Custom domain names and SAML integration



UltraWarm for Amazon ES

A NEW, LOW-COST STORAGE TIER FOR AMAZON ES



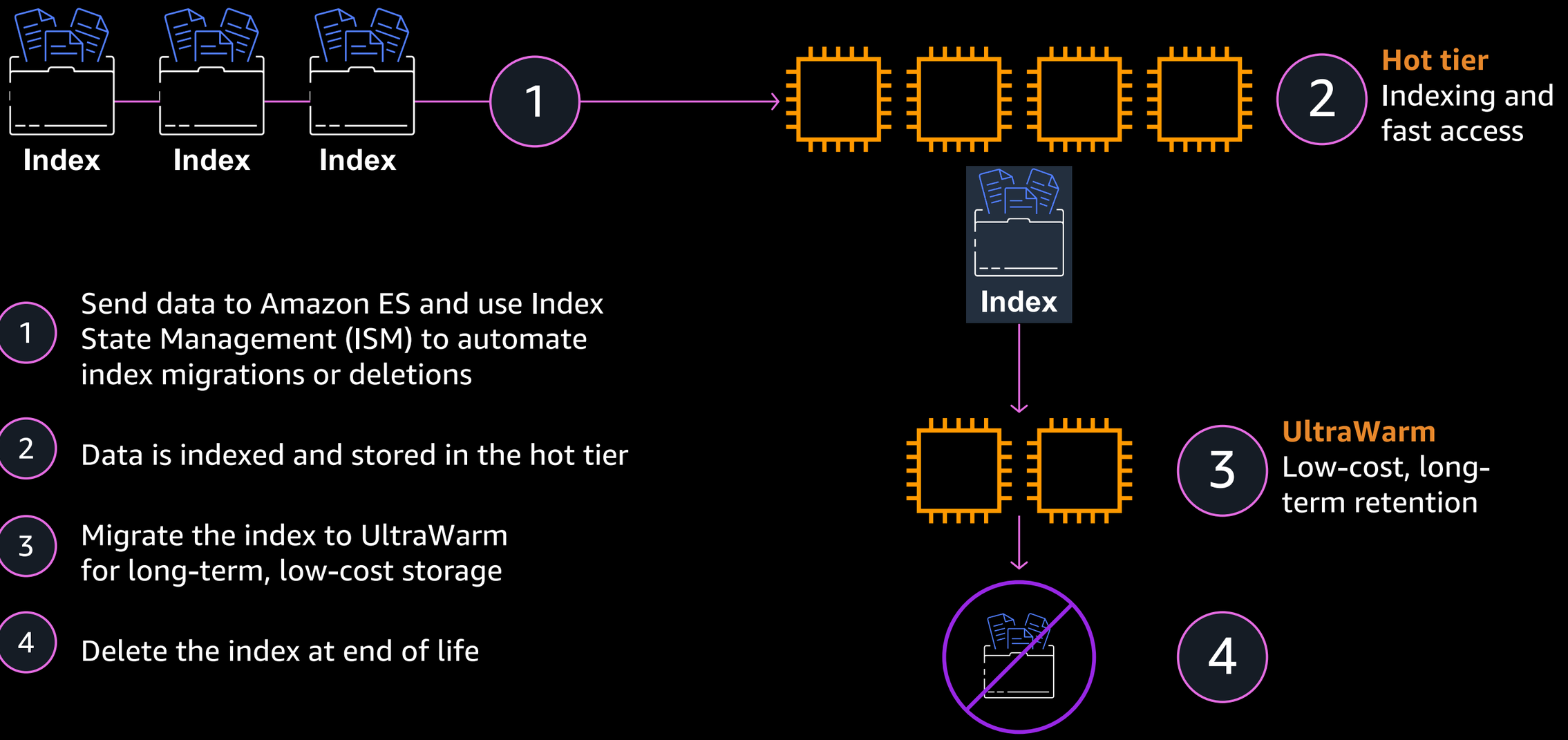
Up to 90% cost reduction vs. hot storage tier

Scale up to 3 PB per domain

Interactive log analytics and visualization

2x performance vs. OS hot/warm/cold architectures

Index State Management (ISM) for data lifecycle



1. Send data to Amazon ES and use Index State Management (ISM) to automate index migrations or deletions
2. Data is indexed and stored in the hot tier
3. Migrate the index to UltraWarm for long-term, low-cost storage
4. Delete the index at end of life

Anomaly Detection in Amazon ES

- Learns what is normal
- Doesn’t require prior ML knowledge
- Deviations from normal are anomalies
- Works in near real time on time series data
- Uses the random cut forest (RCF) algorithm





Seth Carroll
Principal DevOps Engineer
Capital One

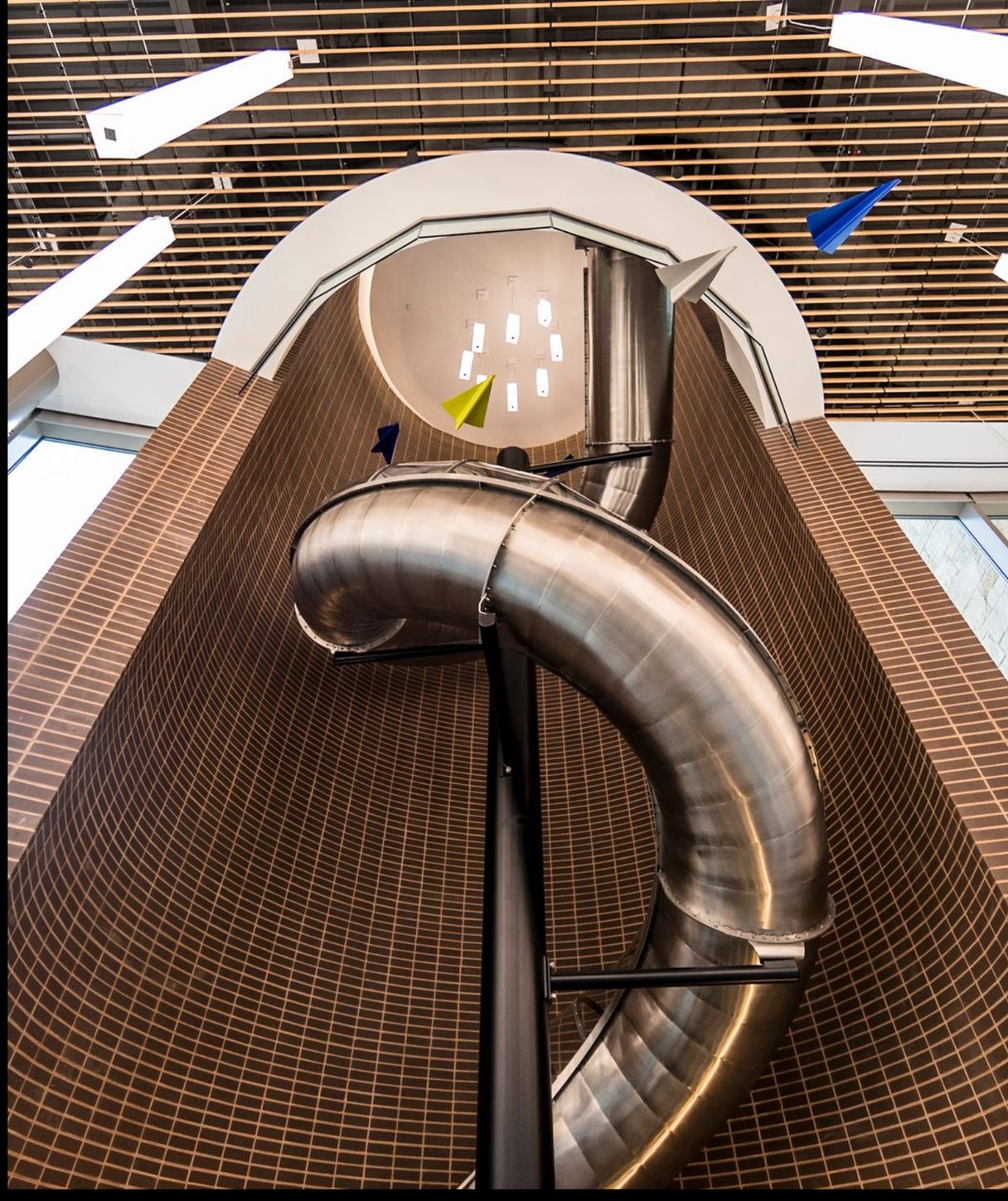
Kumar Anand
Lead Site Reliability Engineer
Capital One

Problem

- Multiple AWS accounts and Regions
- Minimal context in AWS logs
- Various log sources and formats
- Difficult to monitor and diagnose issues

Solution requirements

- Centralized and enriched AWS logs
- Consumable
- Monitoring at scale
- Intelligent alert routing
- Highly available and resilient



Why Amazon Elasticsearch Service?

- Highly available and resilient
- Scales to fit use case
- No external licensing
- Easy to set up and use
- Low cost of maintenance
- Integration into AWS services
- Kibana makes it easy to explore data

Amazon ES use cases

Data store

Metadata

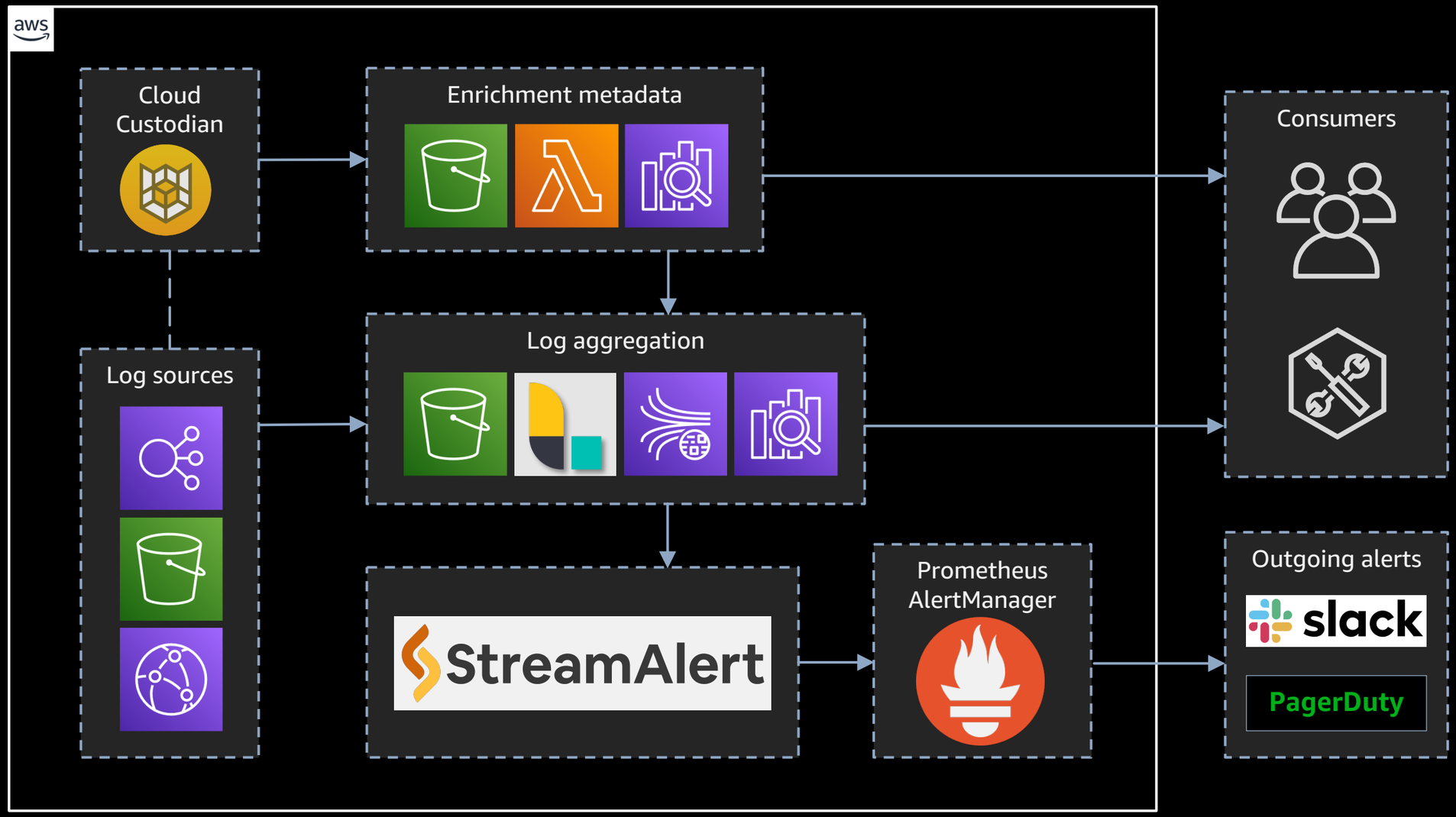
- AWS resource information
- Resource tags
- All accounts
- All active regions

Log aggregation

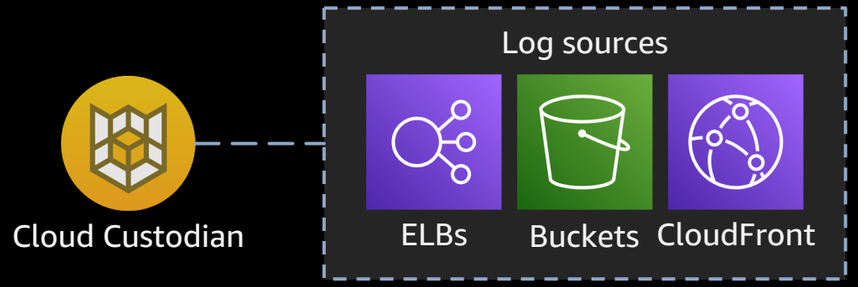
AWS logs

- VPC flow logs
- ELB/ALB access logs
- Amazon CloudFront access logs
- Amazon S3 access logs
- AWS CloudTrail event logs

High-level architecture



Cloud Custodian



- Policy-driven cloud management
- Open source
- Enforce logging configurations:
  - Elastic Load Balancers
  - S3 buckets
  - CloudFront

https://cloudcustodian.io

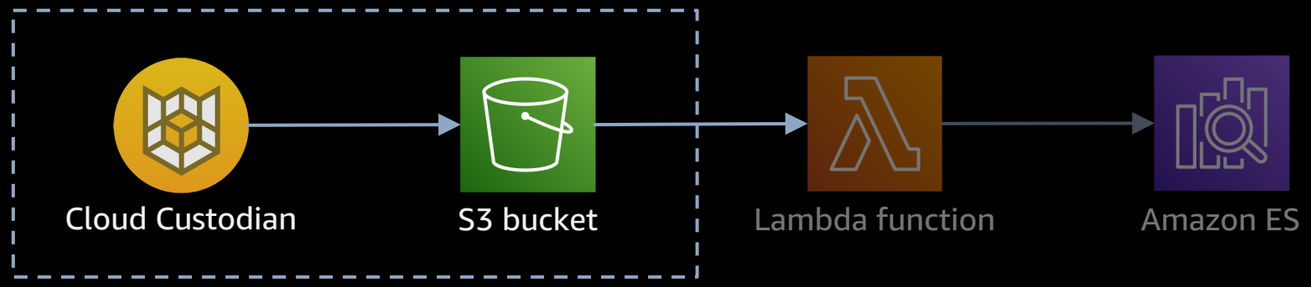
```yaml
policies:

  - name: alb-logging
    resource: app-elb
    filters:
      - type: is-not-logging
    actions:
      - type: set-s3-logging
        bucket: my-logging-bucket
        prefix: logs
        state: enabled

  - name: classic-elb-logging
    resource: elb
    filters:
      - type: is-not-logging
    actions:
      - type: enable-s3-logging
        bucket: my-logging-bucket
        prefix: logs
        emit_interval: 5

  - name: s3-logging
    resource: s3
    filters:
      - "Logging": empty
    actions:
      - type: toggle-logging
        target_bucket: my-logging-bucket
```

Cloud Custodian



- Capture AWS resource information
- Similar to AWS “Describe”
- Combines multiple API calls
- Write resources.json to an S3 bucket
- Invokes Lambda function

```yaml
policies:

  - name: eni-interfaces
    resource: eni

  - name: ec2-instances
    resource: ec2
    filters:
      - type: value
        key: "State.Name"
        op: ne
        value: "terminated"

  - name: ami-images
    resource: ami

  - name: lambda-functions
    resource: lambda

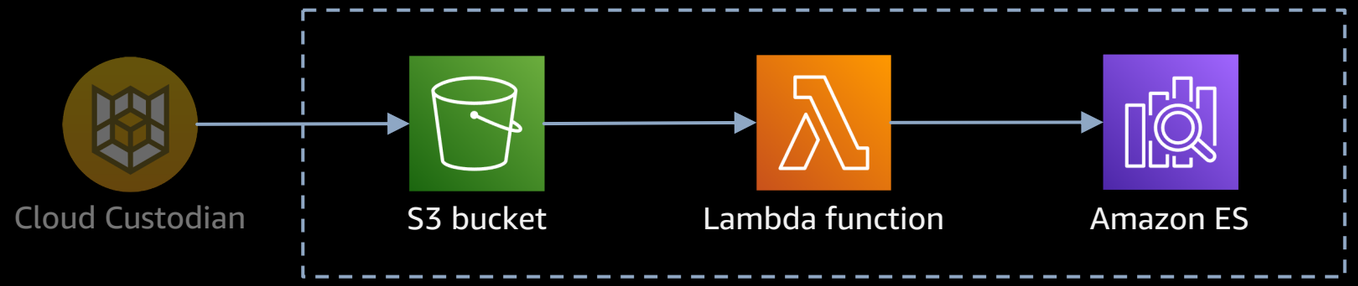
  - name: rds-instances
    resource: rds

  - name: vpcs
    resource: vpc

  - name: subnets
    resource: subnet

  - name: app-load-balancers
    resource: app-elb
```

Enrichment metadata



- Elasticsearch index per resource type
- Cross-index enrichment on ingest
- Identify ENI attachment
- Resource tags copied to ENI record
- Record a “last seen” timestamp
- Mark records inactive

Metadata search example

Amazon EC2 Instance Retirement [AWS Account: 012345678910] Inbox ×

'Amazon Web Services, Inc.' ... Sep 21, 2020, 7:34 AM (7 days ago)

to important-people

Hello,

EC2 has detected degradation of the underlying hardware hosting your Amazon EC2 instance (instance-ID: i-0bf123a4567bc8dd9) associated with your AWS account (AWS Account ID: 012345678910) in the us-east-1 region. Due to this degradation your instance could already be unreachable. We will stop your instance after 2020-10-05 14:00:00 UTC. Please take appropriate action before this time.

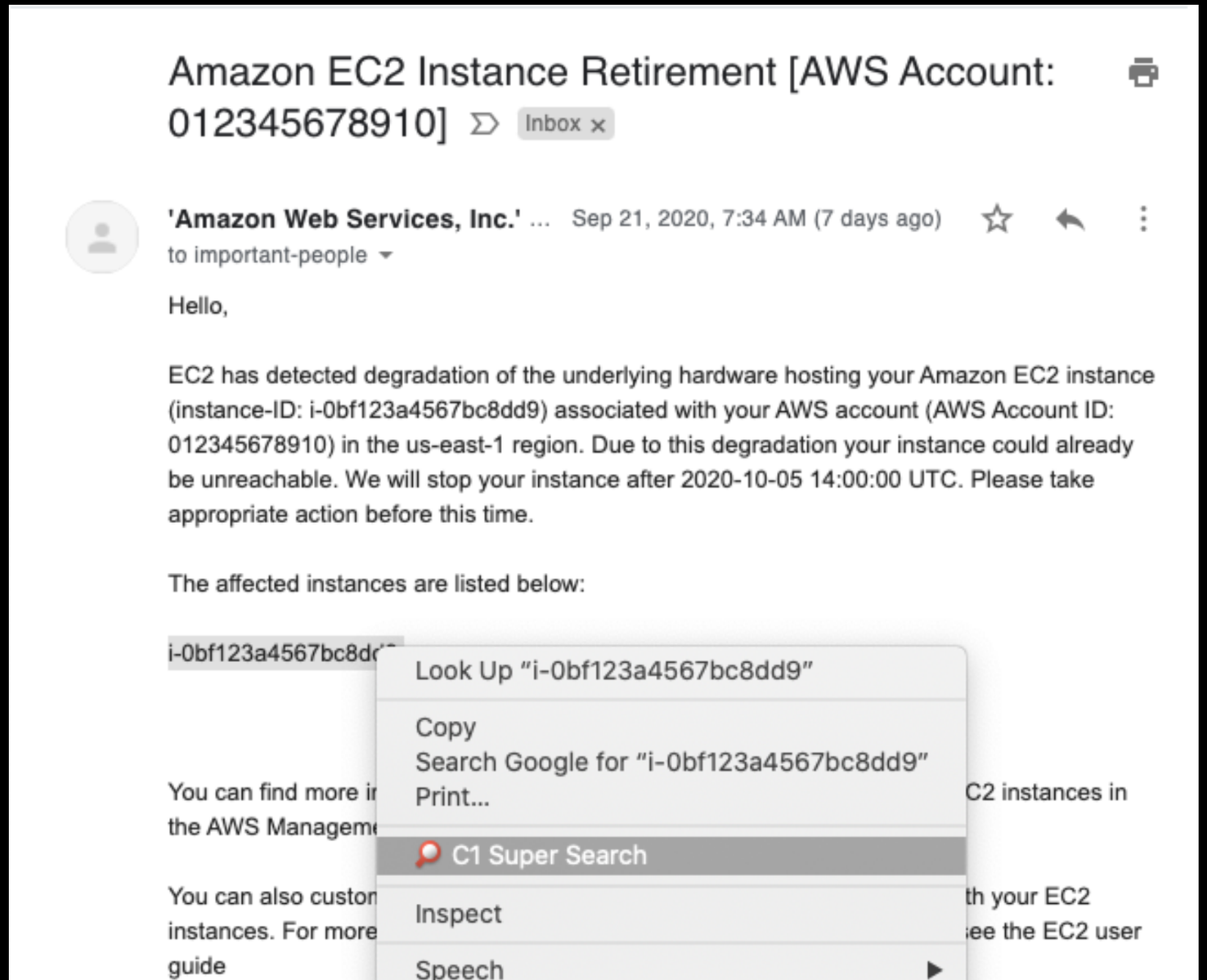
The affected instances are listed below:

i-0bf123a4567bc8dd9

You can find more information about retirement events scheduled for your EC2 instances in the AWS Management Console

You can also customize your event notification to include tags associated with your EC2 instances. For more information about customizing your event notifications see the EC2 user guide

Metadata search example



Metadata search example

Properties

Resource ID	i-0bf123a4567bc8dd9
Resource Type	EC2 Instance
Resource State	Inactive
Last Seen	Sep 21, 2020, 8:01:43 AM
AccountName	qa-sandbox-1
AccountNumber	012345678910
AvailabilityZone	us-east-1a
DaysUntilTermination	58
IamInstanceProfileArn	arn:aws:iam::012345678910:instance-profile/AppRole-Sandbox
IamRoleArn	arn:aws:iam::012345678910:role/AppRole-Sandbox
IamRoleTags	
Tier	WebApp
Owner	dev-team@example.com
ImageAge	2
ImageCreationDate	2020-09-18T19:28:09.000Z
ImageId	ami-0fa12345678910000
ImageName	AmazonLinux2-x64
ImageOsType	linux
InstanceId	i-0bf123a4567bc8dd9
InstanceType	m5.4xlarge
IpAddress	10.11.12.13
LaunchTime	2020-09-21T12:00:54+00:00
Region	us-east-1

Tags

Application	ExampleApp
Environment	QA
Name	Example-Web-App
Owner	dev-team@example.com
c7n-ResourceCreatedTime	2020-09-21 10:23:56.882289

Log enrichment

- Expose metadata through an API
- Bulk query Amazon ES and transform
- Organize data by key field
  - IP address
  - ENI identifier
  - Load balancer ARN
- All active records


```
{
    "10.10.10.101": {
        "Account": "qa-sandbox-1",
        "Component": "WebApp",
        "Environment": "QA",
        "NetworkInterfaceId": "eni-0f1234a56b7c8d9e0",
        "Owner": "dev-team@example.com
        "Region": "us-east-1",
        "SubnetId": "subnet-1a2345b6",
        "SubnetName": "QA-Sandbox-UE1-1-AZ1",
        "Tier": "Web",
        "VpcId": "vpc-abcd1234",
        "VpcName": "QA-Sandbox-UE1-1"
    },
    "10.10.10.102": {
        "Account": "qa-sandbox-1",
        "Component": "API",
        "Environment": "QA",
        "NetworkInterfaceId": "eni-0f1234a56b7c8d9e1",
        "Owner": "dev-team@example.com
        "Region": "us-east-1",
        "SubnetId": "subnet-1a2345b6",
        "SubnetName": "QA-Sandbox-UE1-1-AZ1",
        "Tier": "Web",
        "VpcId": "vpc-abcd1234",
        "VpcName": "QA-Sandbox-UE1-1"
    },
    "10.10.10.103": {
        "Account": "qa-sandbox-1",
        "Component": "Database",
        "Environment": "QA",
        "NetworkInterfaceId": "eni-0f1234a56b7c8d9e2"
```

Log enrichment

- Fetch enrichment JSON on cadence
- Logstash Translate plugin
- Match on identifying field(s)
- Enrich log with metadata

Examples:

- ELB tags, client/backend server tags
- VPC flow log source and destination
- Amazon S3 client access details

```
filter {

    # Client IP Address
    translate {
        field => "client.ip"
        dictionary_path => "${DATA_DIR}/ip-enrichment.json"
        destination => "client"
        refresh_behaviour => "replace"
        refresh_interval => "300"
    }

    # Server IP Address
    translate {
        field => "server.ip"
        dictionary_path => "${DATA_DIR}/ip-enrichment.json"
        destination => "server"
        refresh_behaviour => "replace"
        refresh_interval => "300"
    }

    # ELB ARN
    translate {
        field => "arn"
        dictionary_path => "${DATA_DIR}/elb-enrichment.json"
        destination => "elb"
        refresh_behaviour => "replace"
        refresh_interval => "300"
    }

}
```

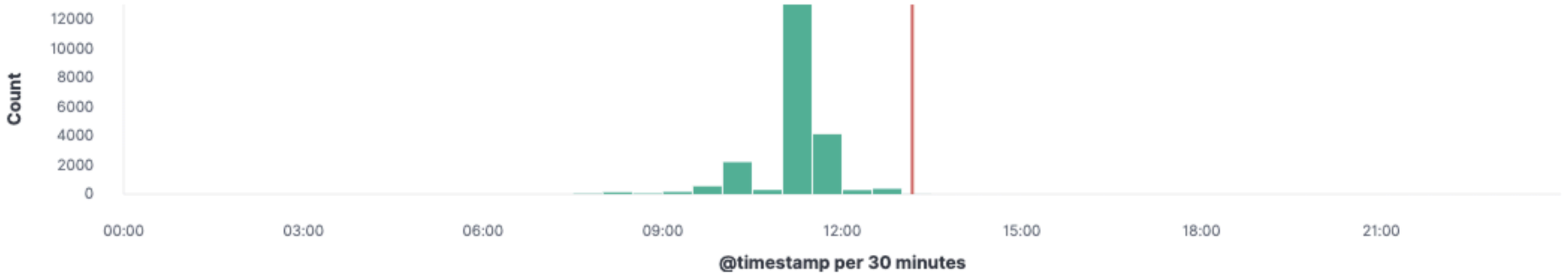
Searching by enriched fields

elb.component:"WebApp" AND server.name:"web-node-1" | Lucene | Today | Show dates

+ Add filter

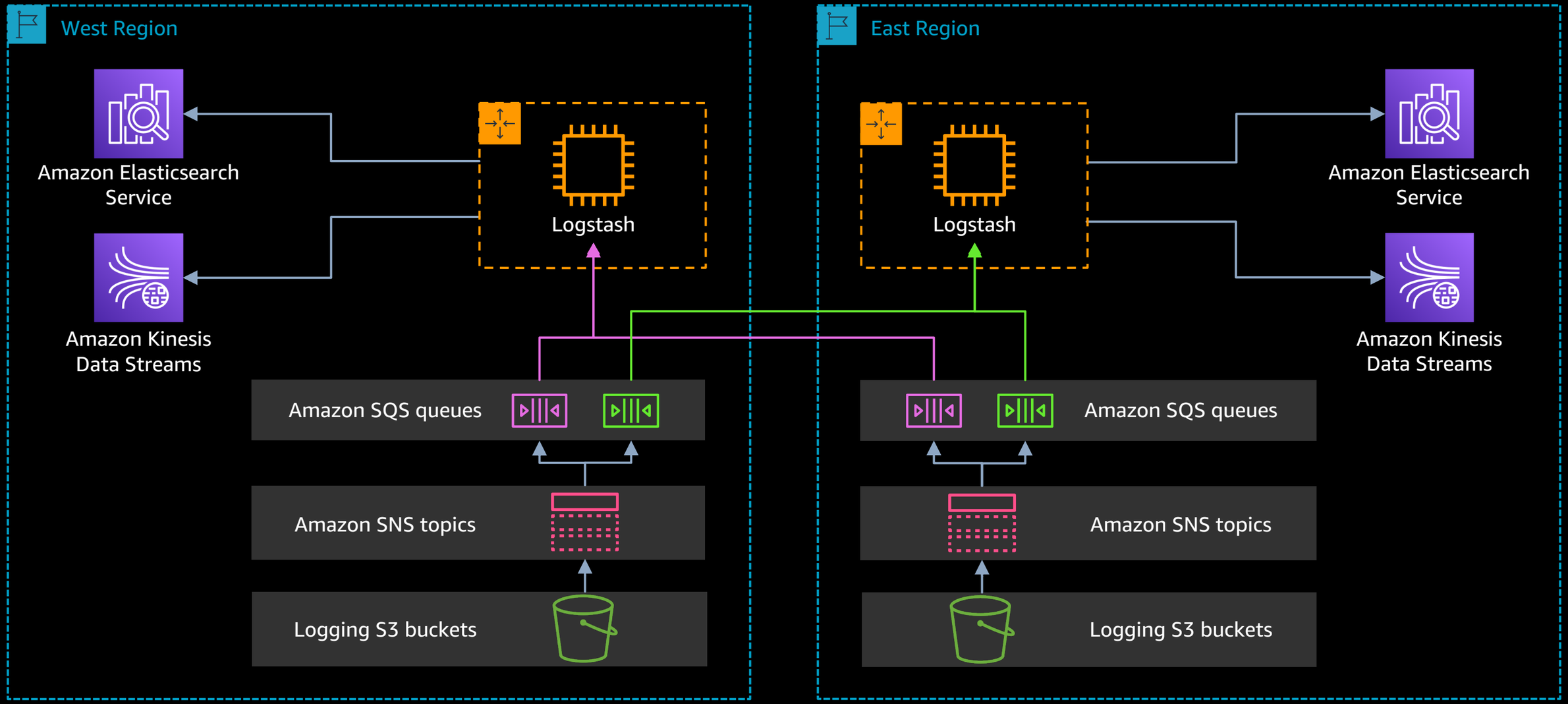
21,280 hits

Sep 30, 2020 @ 00:00:00.000 - Sep 30, 2020 @ 23:59:59.999 — Auto



Time	account	elb.component	server.service	backend_status_code	server.name
Sep 30, 2020 @ 13:04:31.555	qa-sandbox-1	WebApp	EC2	500	web-node-1
Sep 30, 2020 @ 13:04:31.377	qa-sandbox-1	WebApp	EC2	500	web-node-1
Sep 30, 2020 @ 13:04:25.832	qa-sandbox-1	WebApp	EC2	500	web-node-1
Sep 30, 2020 @ 13:04:20.970	qa-sandbox-1	WebApp	EC2	500	web-node-1

Log aggregation architecture



StreamAlert

StreamAlert is a serverless, real-time data analysis framework which ingests, analyzes, and alerts on data from any environment, using data sources and alerting rule defined.

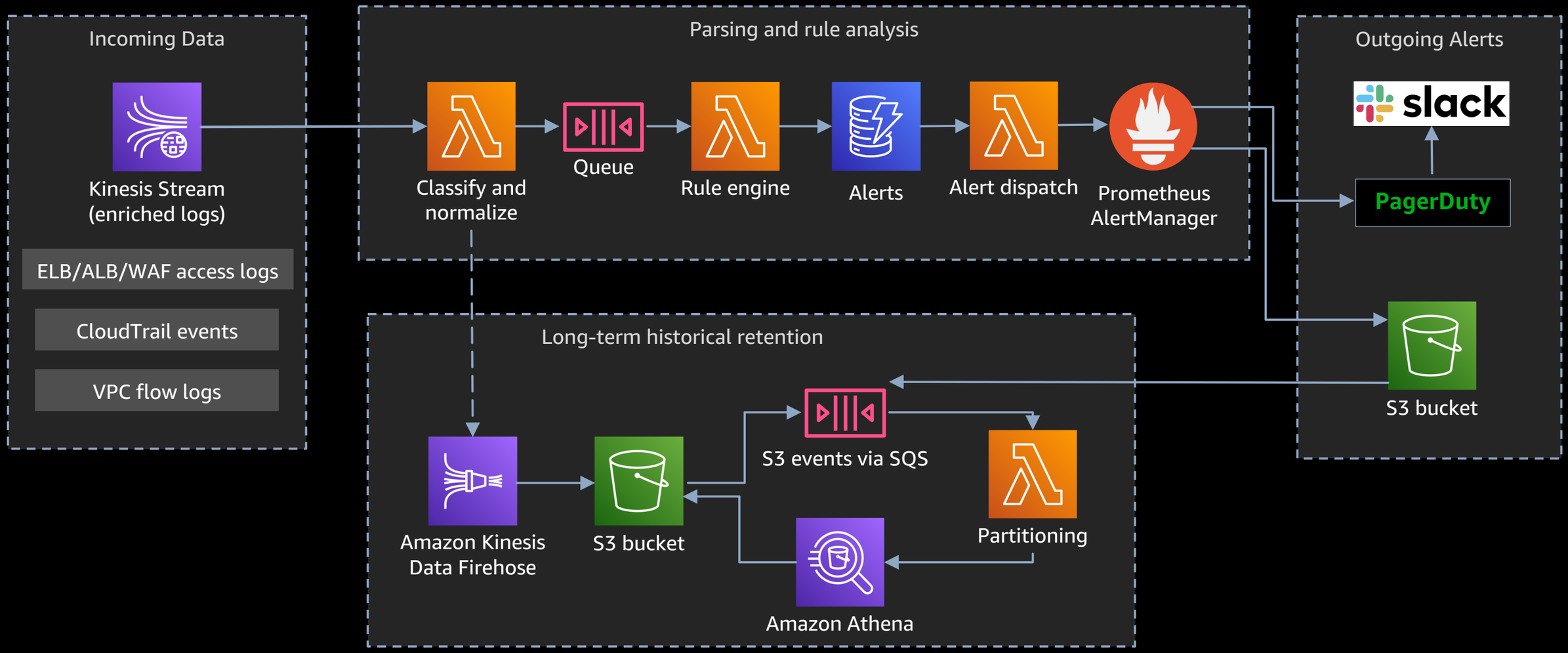
Why StreamAlert?

- Serverless
- Scalable
- Infrastructure as code
- Secure
- Real-time alerting
- Extensive data format support



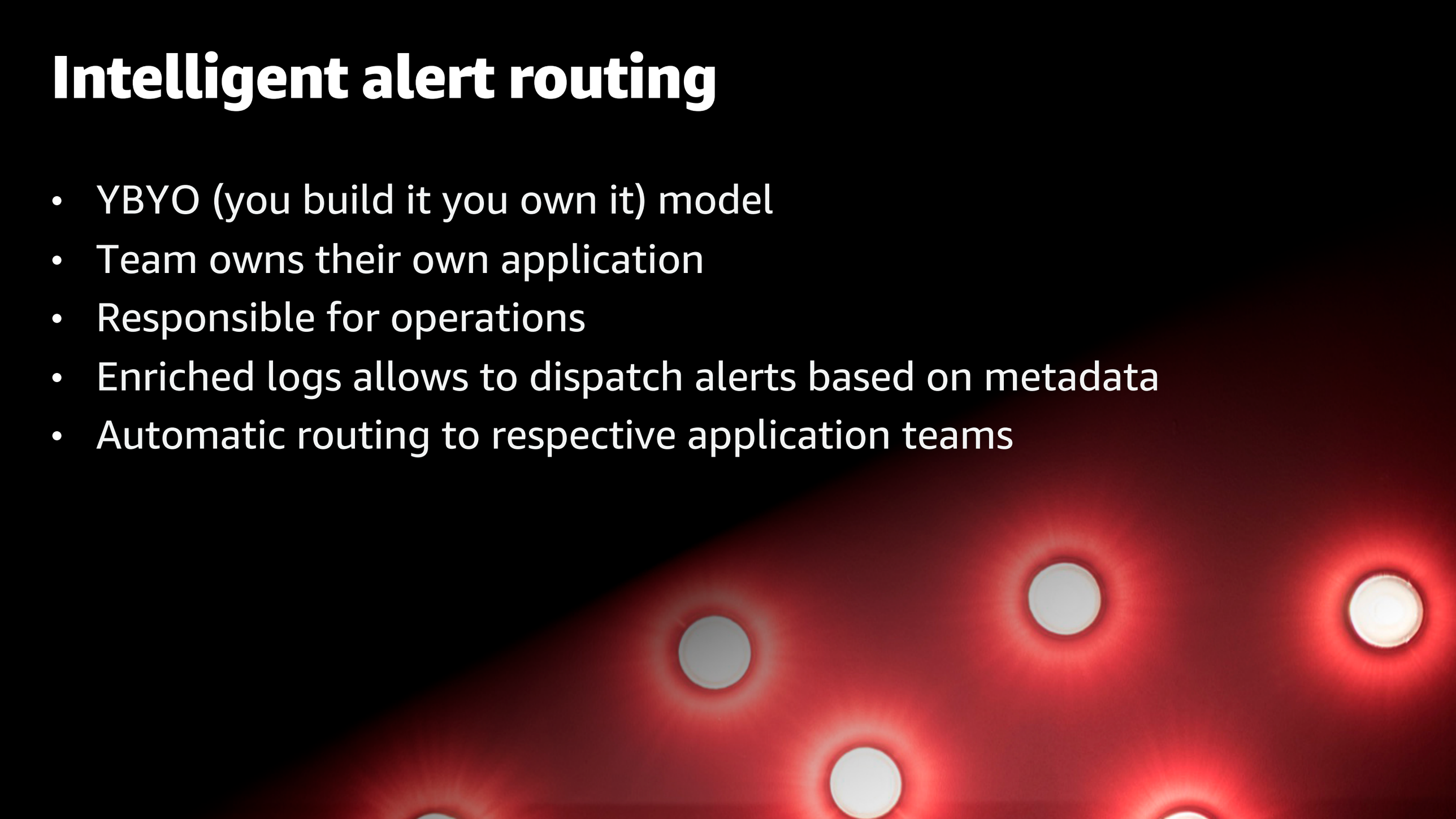
https://www.streamalert.io

StreamAlert architecture



Intelligent alert routing

- YBYO (you build it you own it) model
- Team owns their own application
- Responsible for operations
- Enriched logs allows to dispatch alerts based on metadata
- Automatic routing to respective application teams



ELB/ALB/WAF alert

Use base:

Alert on ELB call from HeadlessChrome outside Capital One subnet CIDR block

Playbook:

- Identify the AWS account and application in the log
- Determine if the intent is valid, malicious, or accidental

Sample alert:



VPC flow log alert

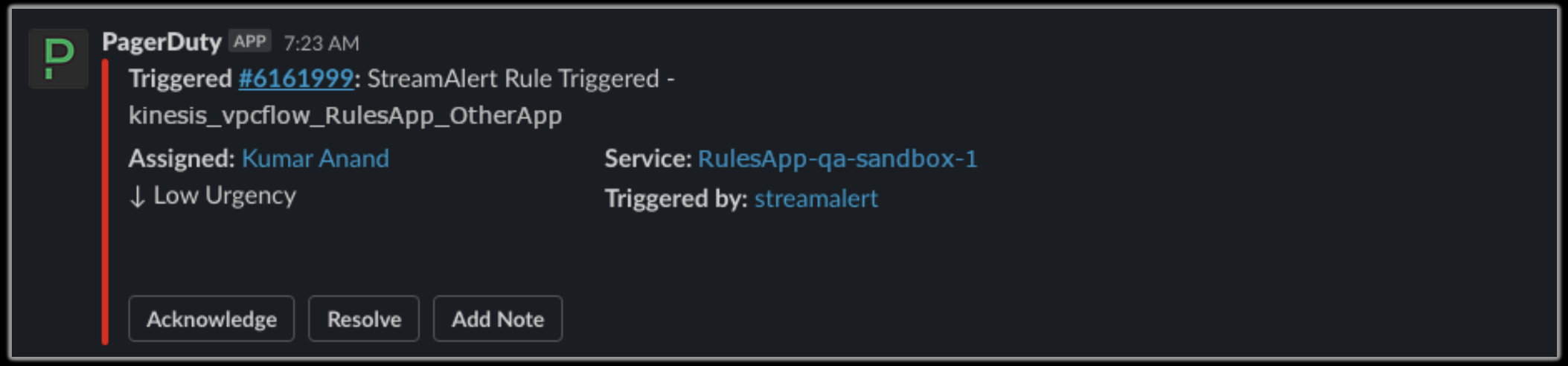
Use case:

Alert if traffic to RulesApp is not from RulesApp or WebApp

Playbook:

- Identify the Component tag in the log
- Check the intent to call RulesApp

Sample alert:



CloudTrail alert

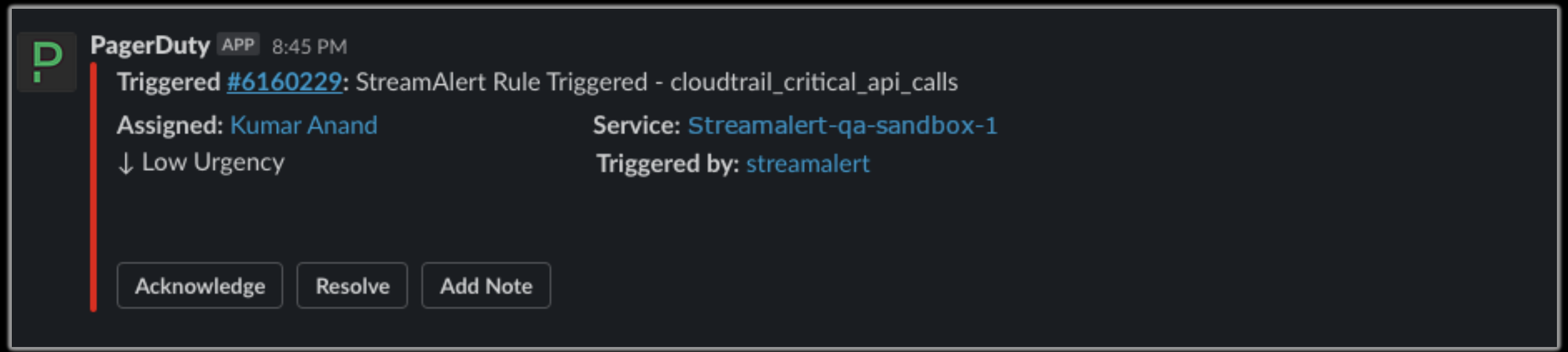
Use case:

Alert on destructive AWS API calls like DeleteFlowLogs, Deletesubnet, DeleteDBCluster, DeleteAccountPublicAccessBlock, etc.

Playbook:

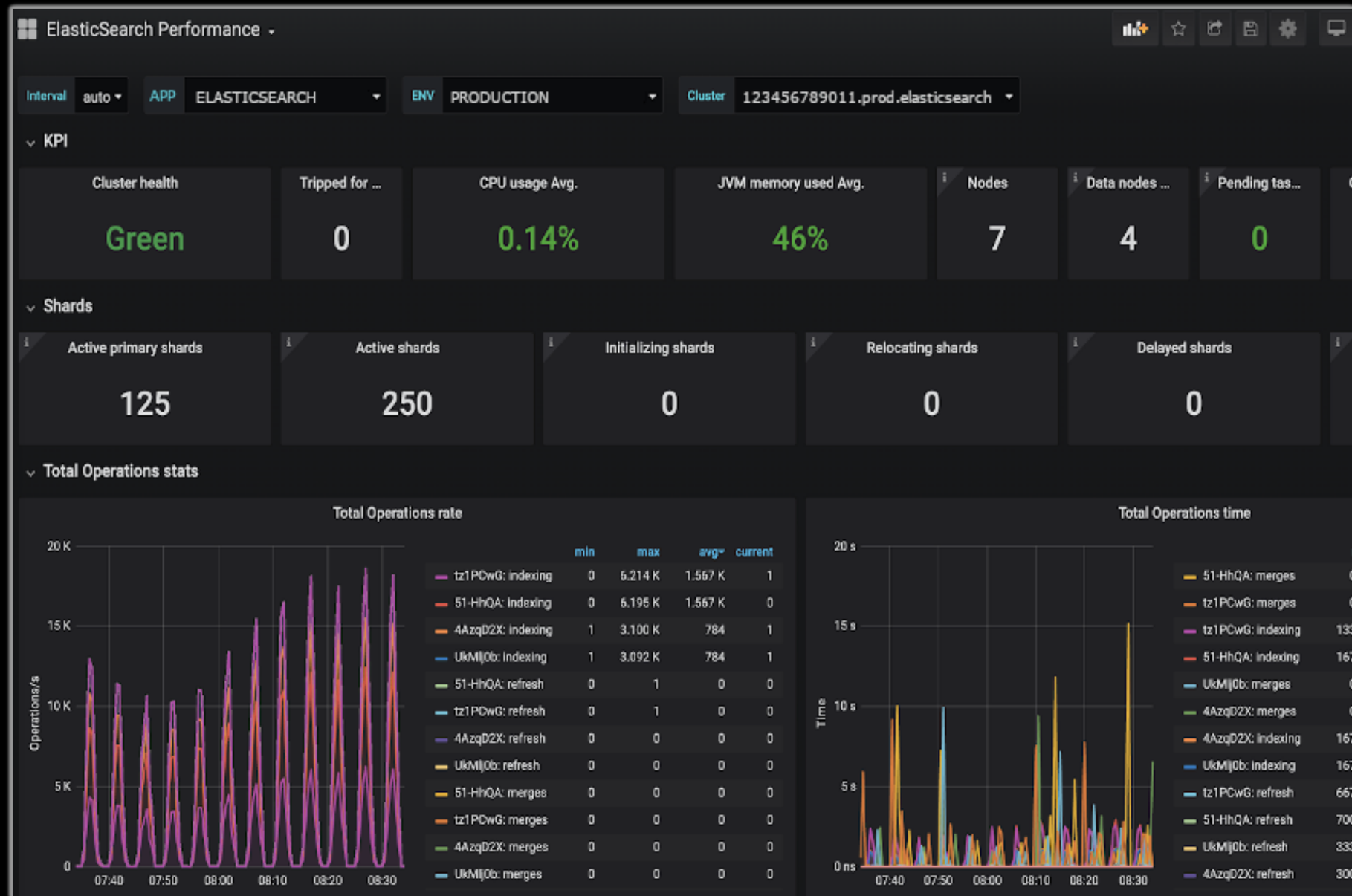
- Identify the AWS account in the log
- Identify what resource(s) are impacted by the API call
- Determine if the intent is valid, malicious, or accidental

Sample alert:



Monitoring Elasticsearch

Prometheus
Elasticsearch exporter
Grafana



UltraWarm

- Previously only kept 10-14 days
- Extend searchable log retention to 90 days
  - 7 days hot, 83 days warm
- Overall ~20% cost savings
- Index state management (ISM)
  - Rollover indices
  - Transition from hot to warm

Benefits

- Near real-time search and helps with troubleshooting
- Possible to analyze billions of records in few seconds
- Provides aggregations to explore trends and patterns of data
- Enriched log messages make searching much easier for users
- UltraWarm allows us to retain much more historical data
- Gives us a consolidated view across AWS accounts

What’s next?

Anomaly Detection

- Built-in feature of Amazon ES
- Currently using in our non-prod accounts
- Expand usage for scenarios that are difficult to monitor with traditional alerts

Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.