AWS re:Invent

ANT203

Sophos: Build security solutions using Amazon Elasticsearch Service

Carl Meadows
Senior Manger, Product Mgmt
Amazon Elasticsearch Service
AWS

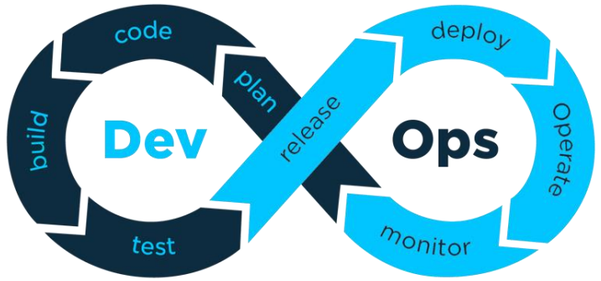
Dennis Griffin
Senior Director of Engineering: Managed Threat Response
Sophos

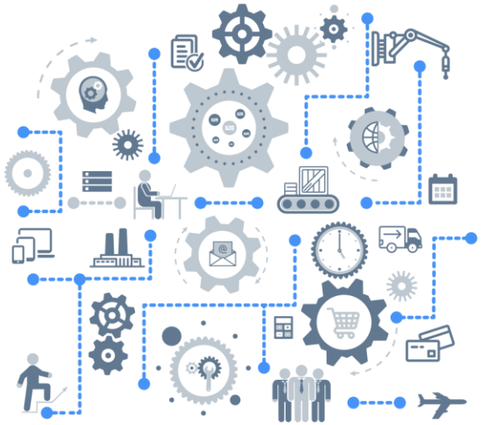
Key Vaidya
Distinguished Engineer: Central Platform Group
Sophos

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

THE **EXPLOSION** OF MACHINE-GENERATED DATA

Machine-generated data is growing **10x faster** than business data







Transition from IT to DevOps

Increase in IoT and Mobile Devices

Cloud-based architectures

Source: insideBigData - The Exponential Growth of Data, February 16, 2017

What is Elasticsearch?

Elasticsearch, Logstash, and Kibana

Distributed search and analytics engine

Easy ingestion and visualization

Rank			DBMS	Score		
Nov 2019	Oct 2019	Nov 2018		Nov 2019	Oct 2019	Nov 2018
1.	1.	1.	Oracle	1336.07	-19.81	+34.96
2.	2.	2.	MySQL	1266.28	-16.78	+106.39
3.	3.	3.	Microsoft SQL Server	1081.91	-12.81	+30.36
4.	4.	4.	PostgreSQL	491.07	+7.16	+50.83
5.	5.	5.	MongoDB	413.18	+1.09	+43.70
6.	6.	6.	IBM Db2	172.60	+1.83	-7.27
7.	7.	↑ 8.	Elasticsearch	148.40	-1.77	+4.94
8.	8.	↓ 7.	Redis	145.24	+2.32	+1.06
9.	9.	9.	Microsoft Access	130.07	-1.10	-8.36
10.	10.	↑ 11.	Cassandra	123.23	+0.01	+1.48

Source: DB-Engines.com, October 2019

Amazon Elasticsearch Service



Amazon Elasticsearch Service is a fully managed service that makes it easy to deploy, manage, and scale Elasticsearch and Kibana

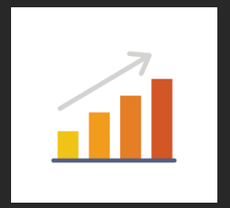
Benefits of Amazon Elasticsearch Service



Supports Open-Source APIs and Tools



Easy to Use



Scalable



Secure



Highly Available



Tightly Integrated with Other AWS Services

Sophos:
Finding the needle in the data stack

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

Introduction



Dennis Griffin

Senior Director of Engineering: Managed Threat Response

Sophos

- Monterey, CA
- Father of 3rd generation twins
- Breaks stuff in production



Key Vaidya

Distinguished Engineer: Central Platform Group

Sophos

- Boston, MA
- Loves dogs
- Stops people from breaking production

Dennis Griffin

Former Founder/CTO at DarkBytes

Mountain biking, golfing, gaming

Pentester, sysadmin, technologist

Sophos MTR platform engineering

Heavily using AWS since 2012



NoSQL for Security

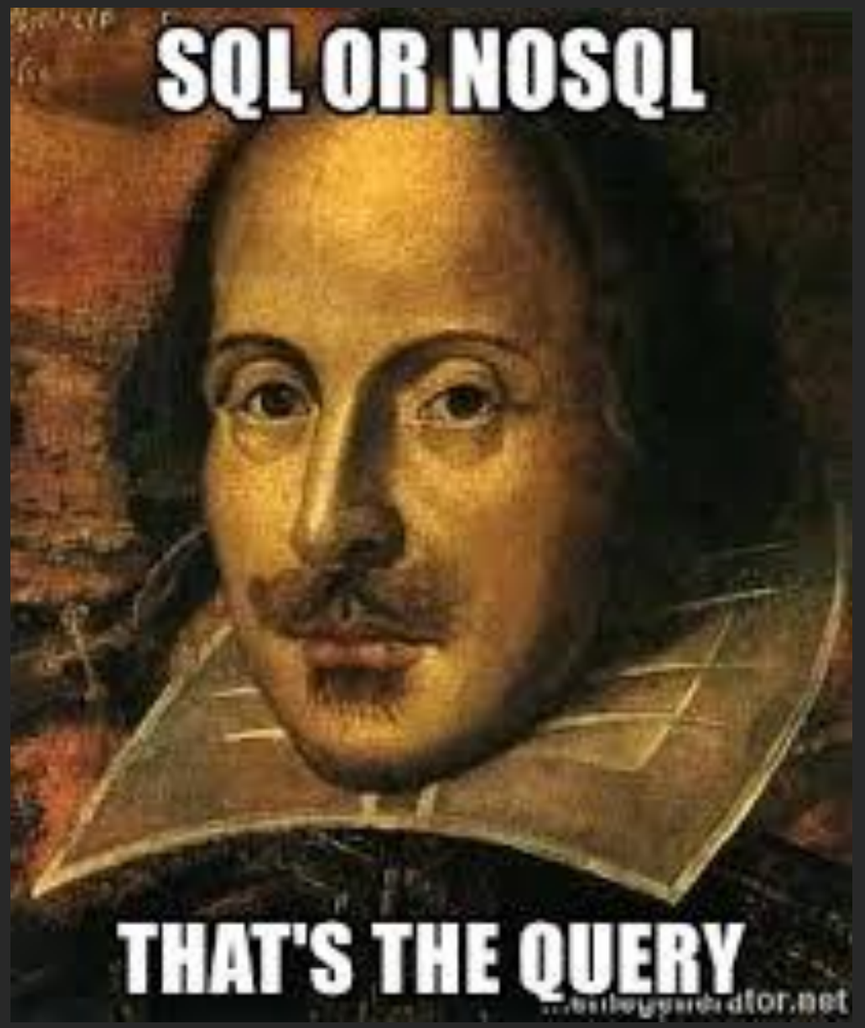
Relational database vs NoSQL

Efficient time-series data

Flexibility for many data types

Minimize migration headaches

Searchable and performant



Battle of NoSQL

Elasticsearch

- Initial release February 2010
- Apache 2.0 license (mostly)
- Supports SQL syntax
- AWS hosted/managed option

MongoDB

- Initial release February 2009
- SSPL license (similar to GPLv3)
- No SQL syntax support
- No AWS hosted/managed option



Amazon Elasticsearch Service

The good

- Dead simple deployment (a few clicks)
- Minimal on-going maintenance
- Very simple upgrades (most the time)
- Lots of scaling options
- Auth'n/Auth'z AWS Sigv4 over HTTPS



Amazon Elasticsearch Service

The bad and the ugly

- Limited _cluster API access
- No SSH access to nodes
- Open ticket and wait for response



Sophos MTR Platform

Collect

- Osquery-based endpoint agent
- Firewall data connectors
- Cloud service data connectors

Detect

- Identify threats/vulnerabilities from Osquery data
- Integrate detections from other Sophos products

Respond

- SOC investigation and case management
- Secure remote access via live terminal
- Automated response playbooks



MTR is fundamentally search

Time-series data analytics

Full-text search for investigations and hunting

Aggregations for data exploration

Dynamically generated queries correlate context

Flexibility to adjust to changing threat landscape

Performant – scalable and distributed



Osquery scheduled query document


```
{
  "name": "open_sockets",
  "hostIdentifier": "EC23B894-6DAD-2244-A44A-9915712F6EE9",
  "calendarTime": "Wed Oct  9 20:33:49 2019 UTC",
  "unixTime": "1570653229",
  "decorations": {
    "eid": "cf71051d-96d2-4cac-875c-ba1bdb6e625d",
    "hostname": "ip-10-0-2-196",
  },
  "columns": {
    "local_address": "10.0.2.196",
    "name": "bundle",
    "remote_address": "10.0.2.168",
    "remote_port": "30176"
  },
  "action": "added",
  "log_type": "result"
}
```

Query string

```
{
  "size": 10,
  "query": {
    "query_string": {
      "query": "decorations.hostname:DENNIS-* AND unixTime:{now-1d TO now}"
    }
  },
  "sort": {
    "unixTime": {
      "order": "desc"
    }
  }
}
```

Painless aggregations

```
"aggs": {
    ”unique_hostnames: {
      "terms": {
        "script": {
          "lang": "painless",
          "inline": "doc['decorations.hostname'].value"
        },
        "min_doc_count": 1,
        "size": 1000
      }
    }
  }
```

Timeline aggregations

```
"aggs": {
    "data_over_time": {
      "date_histogram": {
        "field": "unixTime",
        "interval": "1d"
      }
    }
  }
}
```

Demo

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Hosts | Last 7 Days | DEFAULT | LIVE

SOPHOS

BACK TO SOC

Moosifer (Ashek)

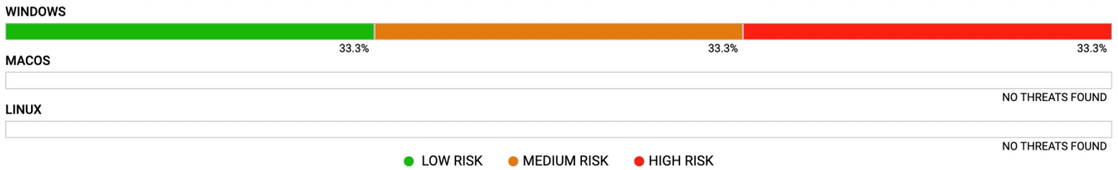
- DASHBOARD
- COLLECT
- DETECT
- RESPOND
- REPORTS
- NOTES
- CONTACTS
- ADMIN
- CUSTOMER

HIGH-RISK HOSTS	HIGH-RISK THREATS	HIGH-RISK VULNERABILITIES
1 ↑ 1	12 ↑ 3	2 ··· 0

WINDOWS HOSTS	MACOS HOSTS	LINUX HOSTS
1 ↑ 1	0 ··· 0	0 ··· 0

THREAT RISK SPREAD

Breakdown of detected threats by risk for each platform.



NETWORK RISK ASSESSMENT

Highest risk, threat, and vulnerability scores across all your assets.



HIGH-RISK THREATS

Top 10 threats by risk.

Page size: 500

Threat ID	Risk ▼	Hostname	Timestamp	Details
COMMAND-b9b3df2c6627e9b60ff6deb0b012dd64	8.0	prod-region-pc	Nov 25, 2019 2:45:40 PM	Process: powershell.exe, PID: 5620
COMMAND-d09c4261d38831b8fa486ac0b7b4bf21	3.0	prod-region-pc	Nov 25, 2019 9:49:29 AM	Process: ngen.exe, PID: 7884
SOPHOS-SAV-AMSI-Mimikatz-C	2.5	prod-region-pc	Nov 25, 2019 2:49:32 PM	Threat: AMSI/Mimikatz-C
SOPHOS-SAV-EICAR-AV-Test	2.0	prod-region-pc	Nov 25, 2019 5:17:43 PM	Threat: EICAR-AV-Test
EXEC-regsvr32.exe	2.0	prod-region-pc	Nov 25, 2019 1:19:04 PM	Process: regsvr32.exe, PID: 888
EXEC-arp.exe	2.0	prod-region-pc	Nov 25, 2019 11:36:51 AM	Process: ARP.EXE, PID: 6552
EXEC-ping.exe	1.0	prod-region-pc	Nov 25, 2019 11:36:51 AM	Process: PING.EXE, PID: 9136
EXEC-route.exe	1.0	prod-region-pc	Nov 25, 2019 11:40:18 AM	Process: ROUTE.EXE, PID: 7176
SOPHOS-CLEAN-Generic-ML-PUA	0.6	prod-region-pc	Nov 25, 2019 2:28:03 PM	Path: C:\Users\Michael Bolton\AppData\Local\Temp\TOKBybs_.exe.part
SOPHOS-CLEAN-CTF-Hacking-Tool	0.6	prod-region-pc	Nov 25, 2019 1:09:25 PM	Path: C:\Users\Michael Bolton\Downloads\ctftool\ctftool\ctftool.exe

HIGH-RISK VULNERABILITIES

Top 10 vulnerabilities by CVSS.

Page size: 500

Key Vaidya

Former CTO at Reflexion Networks

Surfing, autocross, maker

Distinguished Engineer for Central Platform Group (CPG)



Plan for Success

Elasticsearch is Awesome!

There are pitfalls, but you can avoid them with some planning



The Must Do!

Monitor Everything

--All Queries

--Client applications

--Recommended CloudWatch Metrics

Upgrade regularly

--ES is rapidly maturing and upgrades are mostly painless now

Plan to reindex

--As your data access patterns change, be willing to change to your structure to suit



The First Issue – Mapping explosion and index design

- You must plan your data mapping carefully from the beginning
- It really easy to just “Dump Everthing” and it will work for a while, but you won’t be able to scale it. This will create either sparse indexes or Mappapoloza.

- Group logical data together into separate indices

- Use common prefixes for your indexes so that you can still search all of the data

The First Issue – Example

In these documents on a small percentage of field have values

In versions prior to 6.0 this caused both disk space and search performance issues

It’s always better to have higher density

usernid	first	middle	last	city	state
john123	John	James	Smith	Alamo	California
jrice	Jill	Amy	Rice		
mt123	Jeff		Twain	Toledo	Ohio
sadams	Sue		Adams		
adoe	Amy		Doe	Miami	Florida

Image Credit – Elastic Search Blog

The First Issue – Mapping Explosions

- You have too many fields in a document. This is a common problem when using this for security.
  - Everyone wants the data immediately
- Plan your mapping to match the data source
- When gathering output from applications like wireshark it knows over 200,000 fields, the vast majority of these will never be searched
- Monitor your mappings!
  - GET /_all/_mapping ← If you are allowing other users to add data keep a close eye on the changes

The Second Issue – Bad queries issued by good users

If you are providing a dataset that multiple consumers are consuming that dataset you must protect the cluster availability

Tune the following settings:

indices.breaker.total.limit

indices.breaker.request.limit

search.max_buckets

indices.breaker.fielddata.limit

Cluster killers:

- Using group_by and aggregating on high cardinality fields – It's all got to fit in memory
- Deep pagination – You will block merges while the scroll is active
- Leading wildcards – This is a full scan of the whole content of the cluster

Bad queries issued by good users

Let’s explore some of the use cases that cause users to issue these types of queries



Bad queries issued by good users

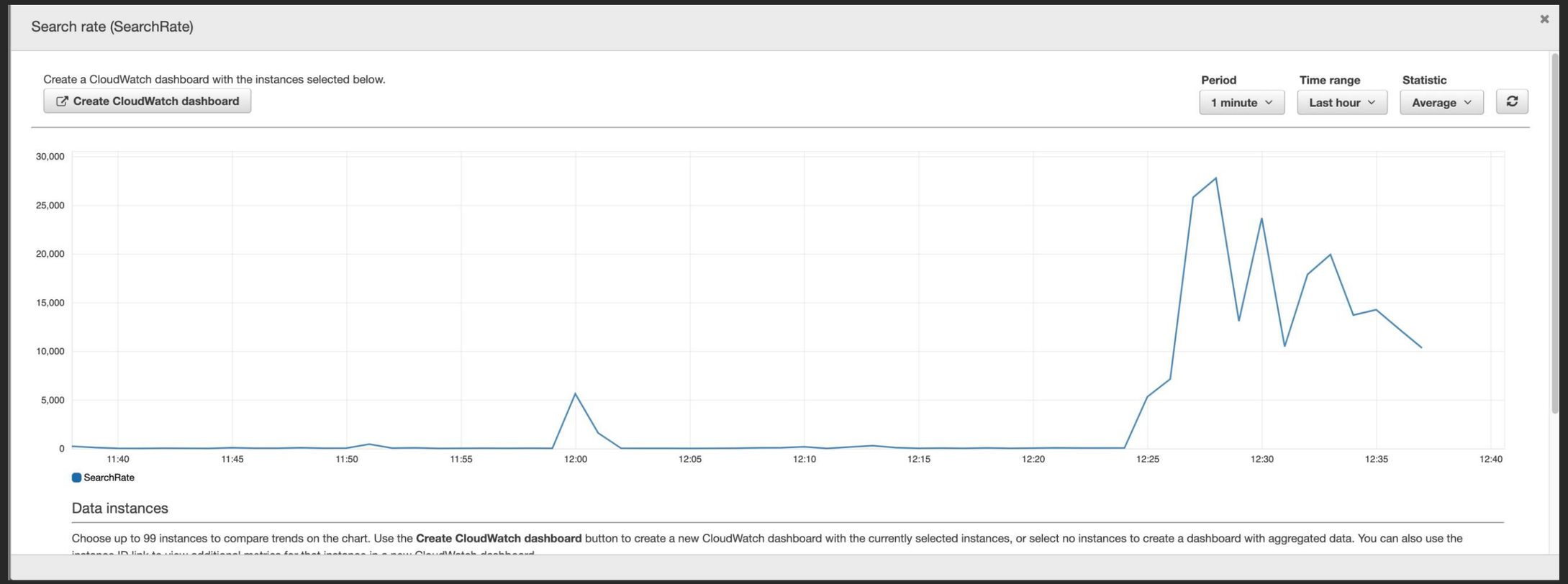
Show me how many time powershell was run on every computer and how many times it was run and organize by the parameters

A query like this is which has a bunch of highly cardinal fields being used this way is going to scan your entire datastore and then try to create aggregation buckets for each unique parameter. This will not hurt at a scale of 1000s but your resources are going to grow as the date range or events increases.


```
{
…
}
},
"aggs": {
"blah2": {
"terms": {
"field": "columns.name",
"size": 100,
"order": {
"_count": "desc"
}
},
"aggs": {
"blah3": {
"terms": {
"field": "decorations.hostname",
"size": 100,
"order": {
"_count": "desc"
}
…
},
"query": {
"query_string": {
"query": "decorations.hostname:* AND columns.cmdline:*cmd* AND
unixTime:{now-1d TO now} AND NOT (columns.name:csc.exe OR
columns.cmdline:*mpcmdrun* OR columns.name:cmd.exe
columns.name:vbc.exe OR columns.name:git.exe)"
}
},
"sort": {
"unixTime": {
"order": "desc"
}
}
```

Bad queries issued by good users

This is going to result in unhappy cluster
and lack availability for other users



Show me the number of events per day by customer/device/type/status group by hour

```
{ "aggregations": { "date": { "terms": { "field": "evt_date" },
       "aggregations": { "customer": { "terms": { "field": "cust_id" },
              "aggregations": { "device": { "terms": { "field": "devi_id" },
                     "aggregations": { "type": { "terms": { "field": "type" },
                            "aggregations": { "action": { "terms": { "field": "action" },
                                   "aggregations": { "hr": { "terms": { "field": "hr" } }
} } } } } } } } } } } } }
```

The use of aggregation here is going to cause havoc in your cluster as the number of nest aggregation is very large. This is an example by an operator who was attempting to create a dashboard by hand and didn’t fully understand how to optimize his query.

If you need this type of aggregation, consider using nested datatypes

Show me all the events for the computers in the EU

```
{
      "query": {
            “wildcard"  : {
                  “field” : “*EU*”
                        }
      }
}
```

In this example a customer used a name convention for their machines : /domain/Function-**region**-hostname

The operator would make queries of this type to try to find specific machines, but this type of wildcard search will cause a full scan, similarly regex scans are generally a bad idea. Better to handle this at ingest and create additional metadata tags.

Give me all the data for an audit report

```
_search?scroll=10m
{
        "query": {
                "match_all"  : {}
        }
}
```

The merger will continue to work during this operation, but older segments will not be deleted while the cursor is active as they are in use. This is meant for batch operations, like when re-indexing. Best to avoid these all together and build specific tooling to handle batch exports for operators.

Resources

"

https://docs.aws.amazon.com/elasticsearch-service/latest/developerguide/cloudwatch-alarms.html

https://docs.aws.amazon.com/en_pv/elasticsearch-service/latest/developerguide/es-managedomains.html

https://docs.aws.amazon.com/en_pv/elasticsearch-service/latest/developerguide/curator.html

https://opendistro.github.io/for-elasticsearch/features/analyzer.html







Summing it all up!

Managing your data and the usage of your operators is critical to success

Monitor your mappings
Monitor you cluster usage
Monitor your queries
Educate your devs and operators
Consider what data your operators need and index appropriately

SOPHOS





Thank you!

Carl Meadows
carlmead@amazon.com

Dennis Griffin
Dennis.Griffin@sophos.com

Key Vaidya
Key.Vaidya@sophos.com

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey in the mobile app.

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.