Objectivity

WYDZIAŁ ELEKTRONIKI,
TELEKOMUNIKACJI
I INFORMATYKI

Elasticsearch

Damian Strojek, Gdansk University of Technology

www.objectivity.co.uk

Table of Contents

- What exactly is „Elastic”?
- Architecture of Elasticsearch
- Example of a query
- My experience
- Real life examples
- Similar solutions
- Questions

What exactly is „Elastic"?

Fast

Applications usually have a huge amount of saved data. You have to search through them to find the right ones. Therefore, the order of the steps performed and the structure of the record itself are important.

Scalable

Elasticsearch can be used to search any kind of document. It provides scalable search, has near real-time search, and supports multitenancy. It is scalable up to petabytes of structured and unstructured data.

Source-available

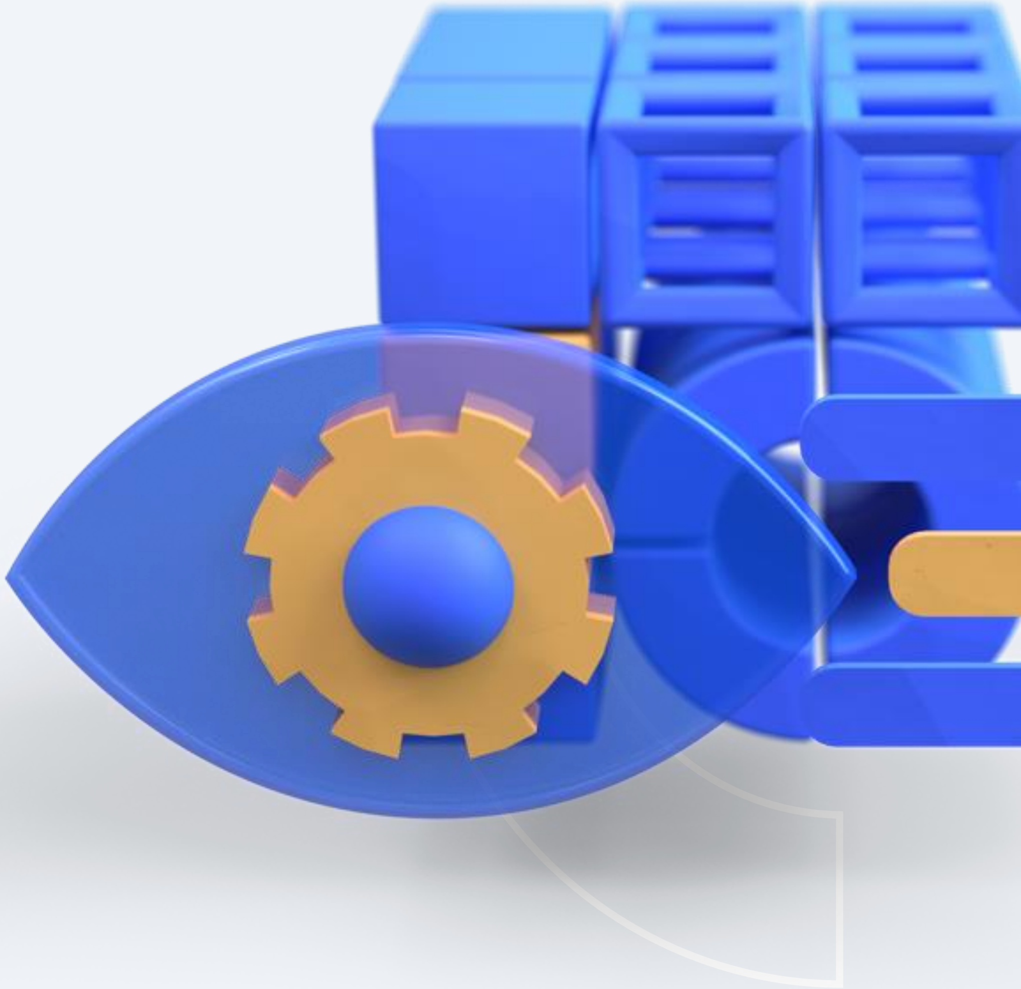
The corresponding source code is available under the "Elastic License", a source-available license.

Real-time search

When a document is stored in Elasticsearch, it is indexed and fully searchable in near real-time - within 1 second.

Github repo

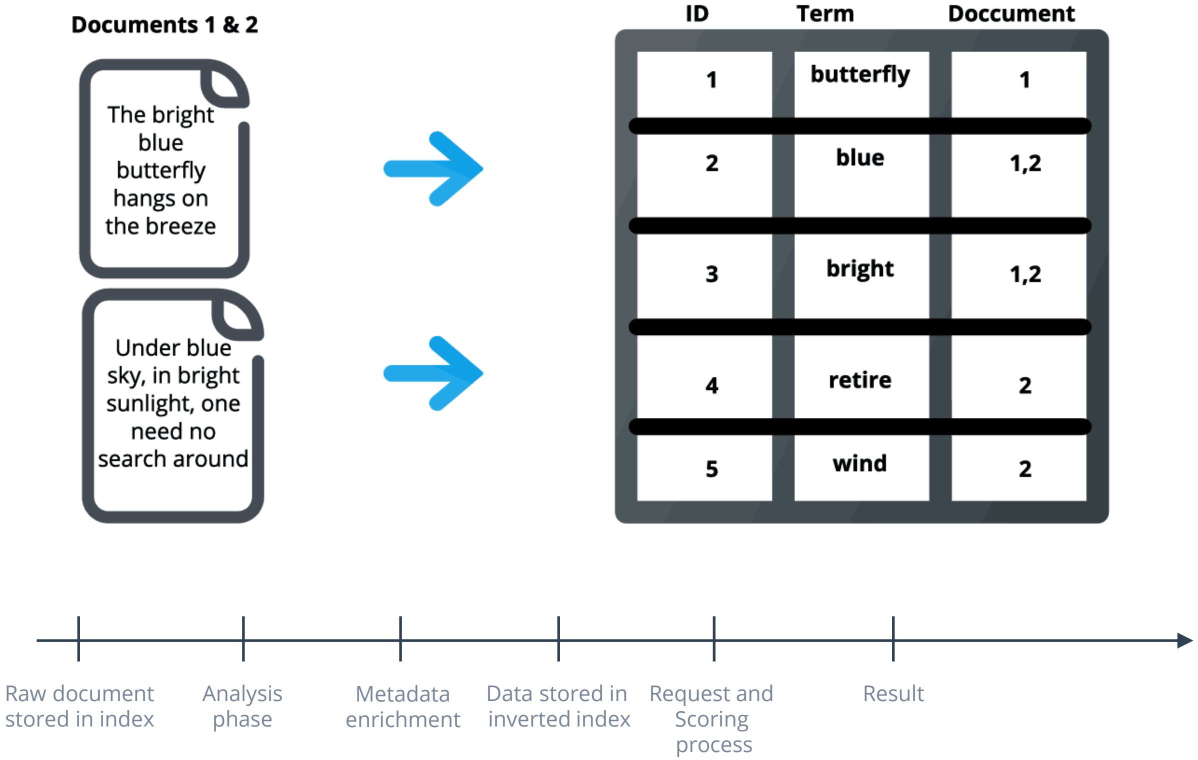
github.com/elastic/elasticsearch



Architecture of Elasticsearch

Key Concepts

- *Node* – Single running instance of Elasticsearch;
- *Cluster* – Collection of one or more nodes It provides collective indexing and search capabilities acrros all the nodes for entire data;
- *Index* – Collection of different type of documents and their properties. Index also uses the concept of shards to improve the performance;
- *Document* – Collection of fields in a specific manner defined in JSON format;
- *Shard* – Indexes are horizontally subdivided into shards. This means each shard contains all the properties of document but contains a smaller number of JSON objects than index
- *Replicas* – Allows a user to create replicas of their indexes and shards.



Documents 1 & 2

The bright blue butterfly hangs on the breeze

Under blue sky, in bright sunlight, one need no search around

ID	Term	Doccument
1	butterfly	1
2	blue	1,2
3	bright	1,2
4	retire	2
5	wind	2

Raw document stored in index
Analysis phase
Metadata enrichment
Data stored in inverted index
Request and Scoring process
Result

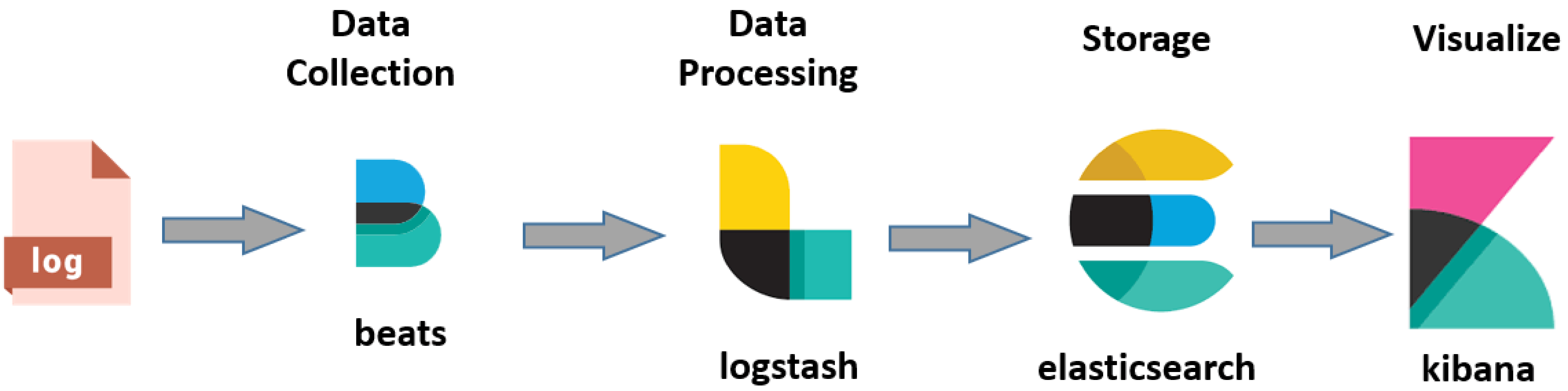
Architecture of Elasticsearch

Elasticsearch is developed alongside:

- **Kibana** – data visualiation dashboard software for Elasticsearch. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster.
- **Logstash** – tool for managing events and logs. Various applications can send log events to Logstash, which gathers the messages, converts them into JSON documents, and stores them in an ***OpenSearch** cluster.
- **Beats** – platform for single-purpose data shippers. They send data from hundreds of thousands of machines and systems to Logstash or Elasticsearch.

***OpenSearch** is a community-driven open source serach and analytics suite that replaced ElasticSearch and Kibana in 2021.

Architecture of Elasticsearch



Data Collection

Data Processing

Storage

Visualize

log

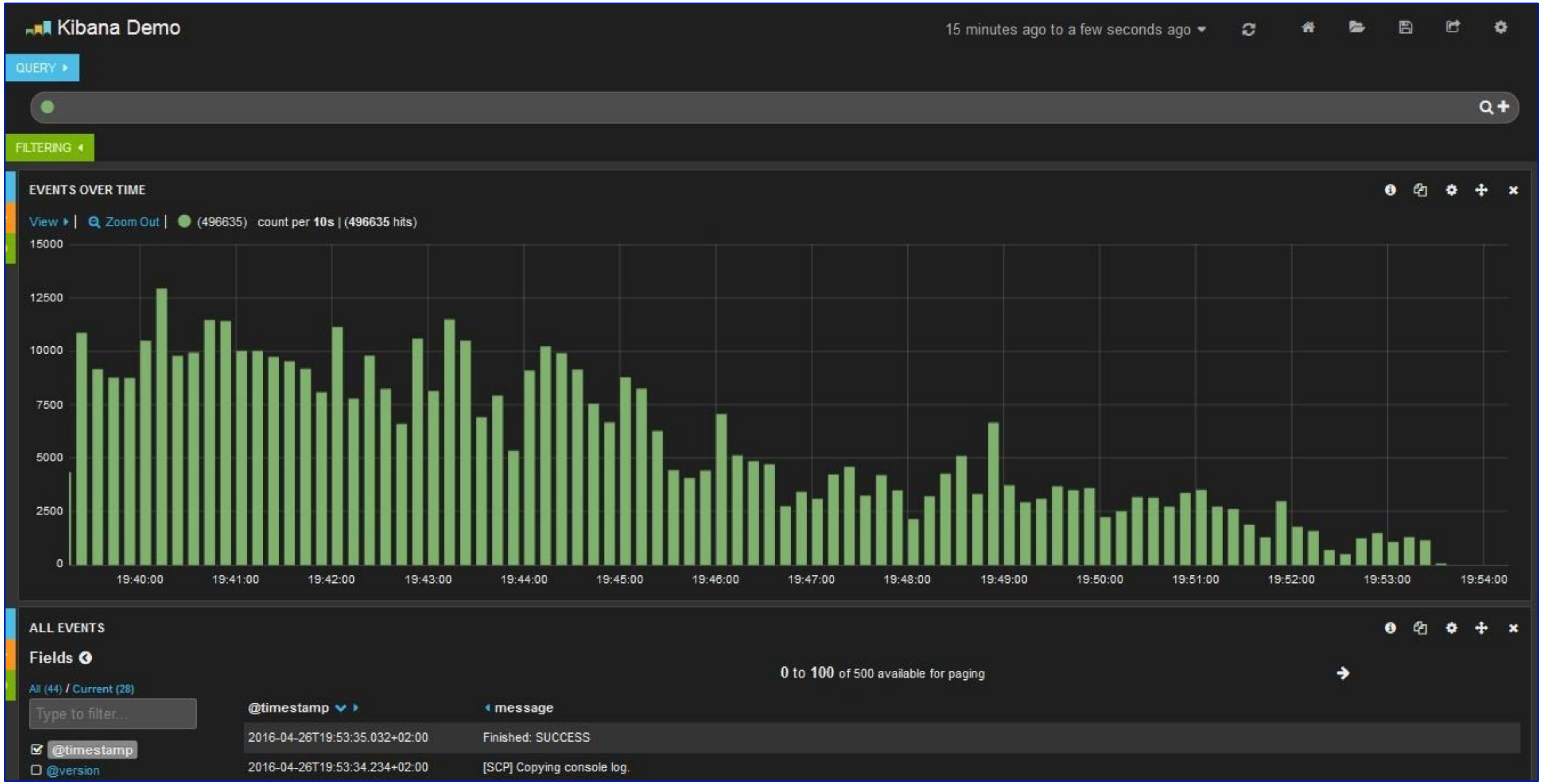
beats

logstash

elasticsearch

kibana

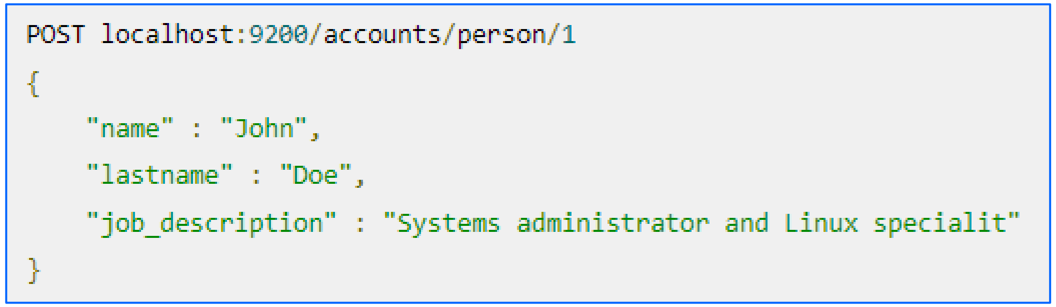
Kibana Demo



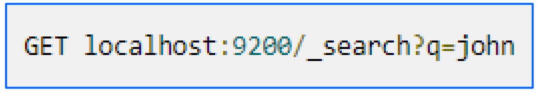
0 to 100 of 500 available for paging

```
@timestamp                       message
2016-04-26T19:53:35.032+02:00    Finished: SUCCESS
2016-04-26T19:53:34.234+02:00    [SCP] Copying console log.
```

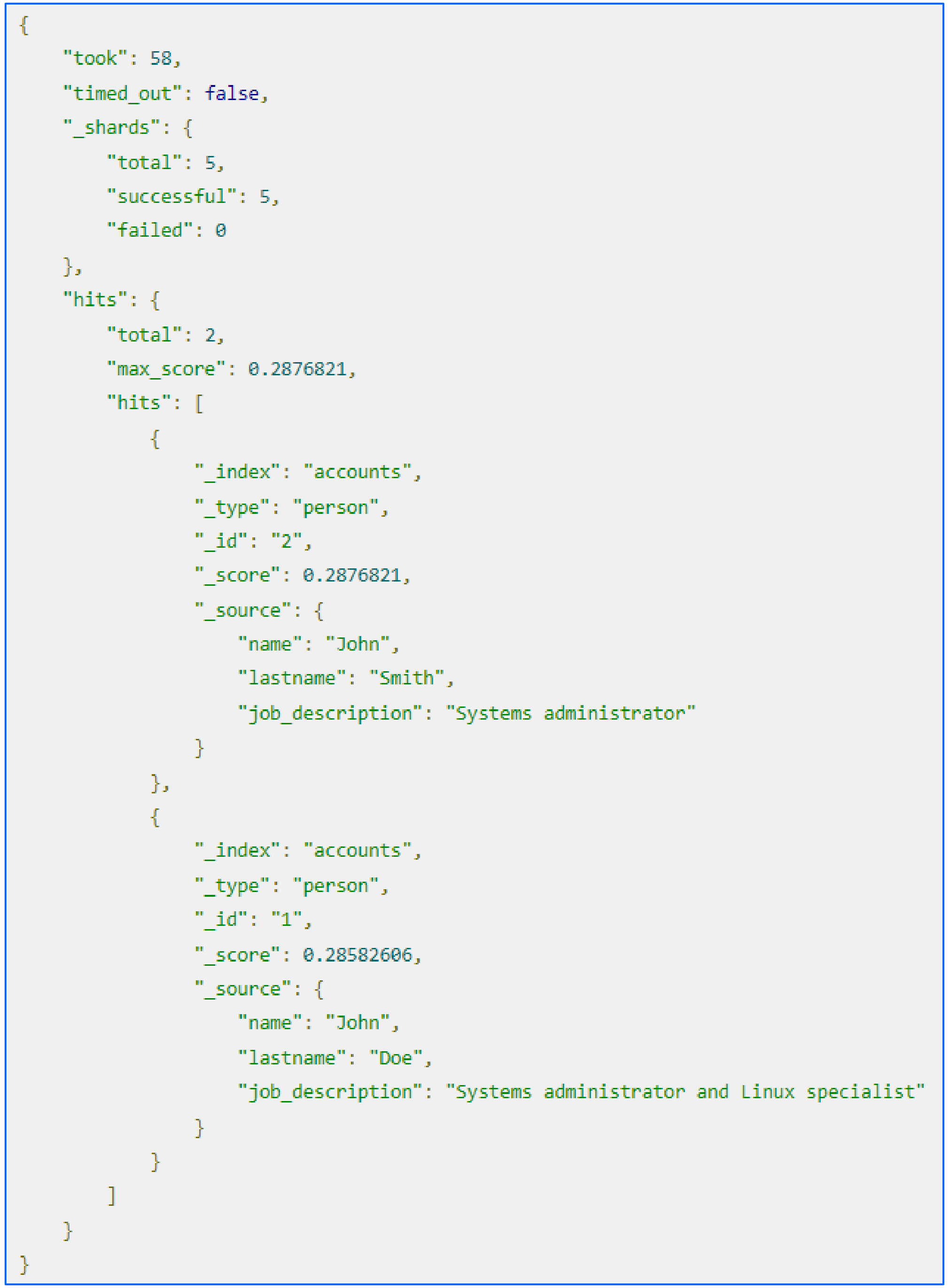
Example of a query



```
POST localhost:9200/accounts/person/1
{
    "name" : "John",
    "lastname" : "Doe",
    "job_description" : "Systems administrator and Linux specialit"
}
```



```
GET localhost:9200/_search?q=john
```

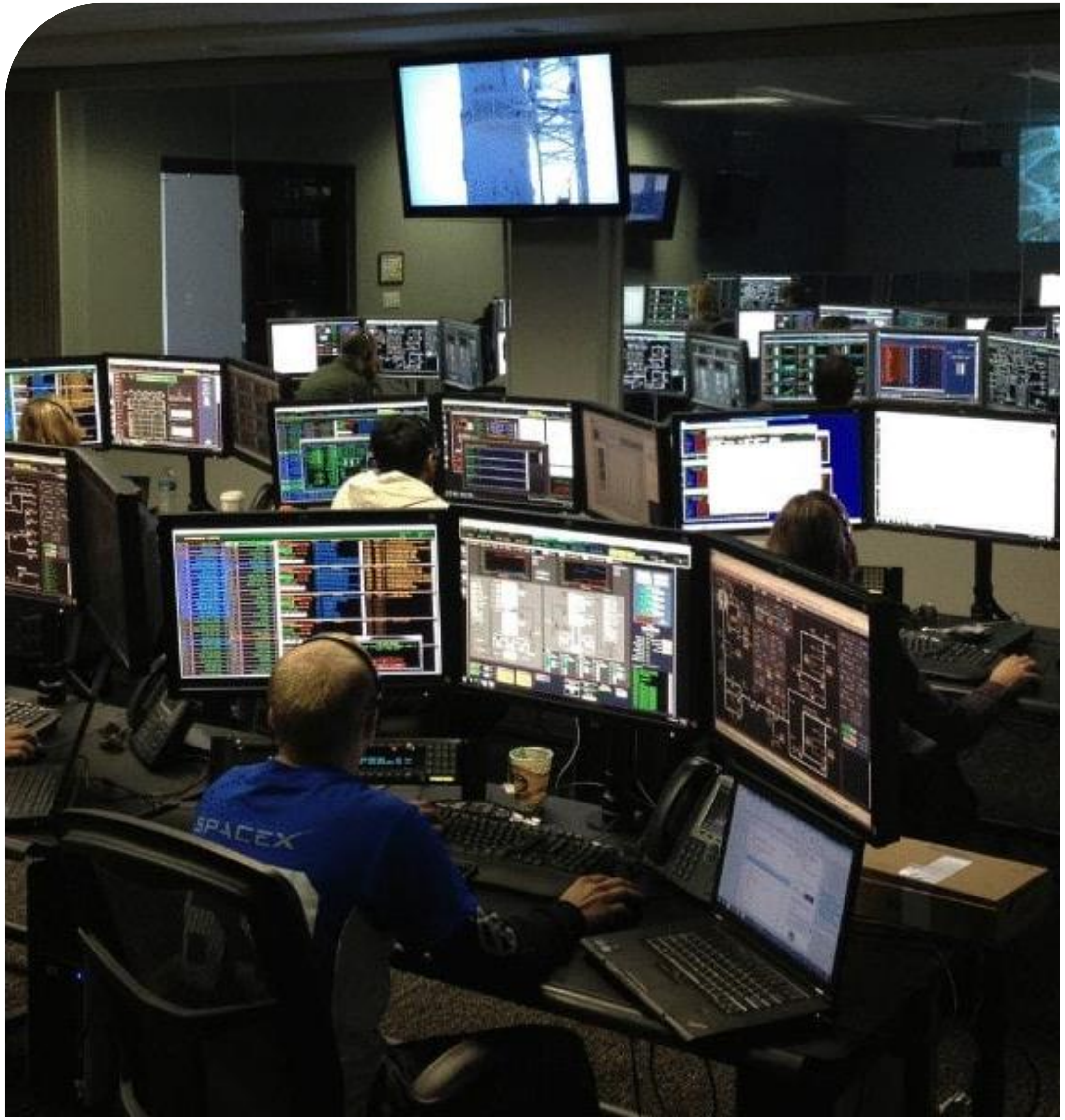


```
{
    "took": 58,
    "timed_out": false,
    "_shards": {
        "total": 5,
        "successful": 5,
        "failed": 0
    },
    "hits": {
        "total": 2,
        "max_score": 0.2876821,
        "hits": [
            {
                "_index": "accounts",
                "_type": "person",
                "_id": "2",
                "_score": 0.2876821,
                "_source": {
                    "name": "John",
                    "lastname": "Smith",
                    "job_description": "Systems administrator"
                }
            },
            {
                "_index": "accounts",
                "_type": "person",
                "_id": "1",
                "_score": 0.28582606,
                "_source": {
                    "name": "John",
                    "lastname": "Doe",
                    "job_description": "Systems administrator and Linux specialist"
                }
            }
        ]
    }
}
```

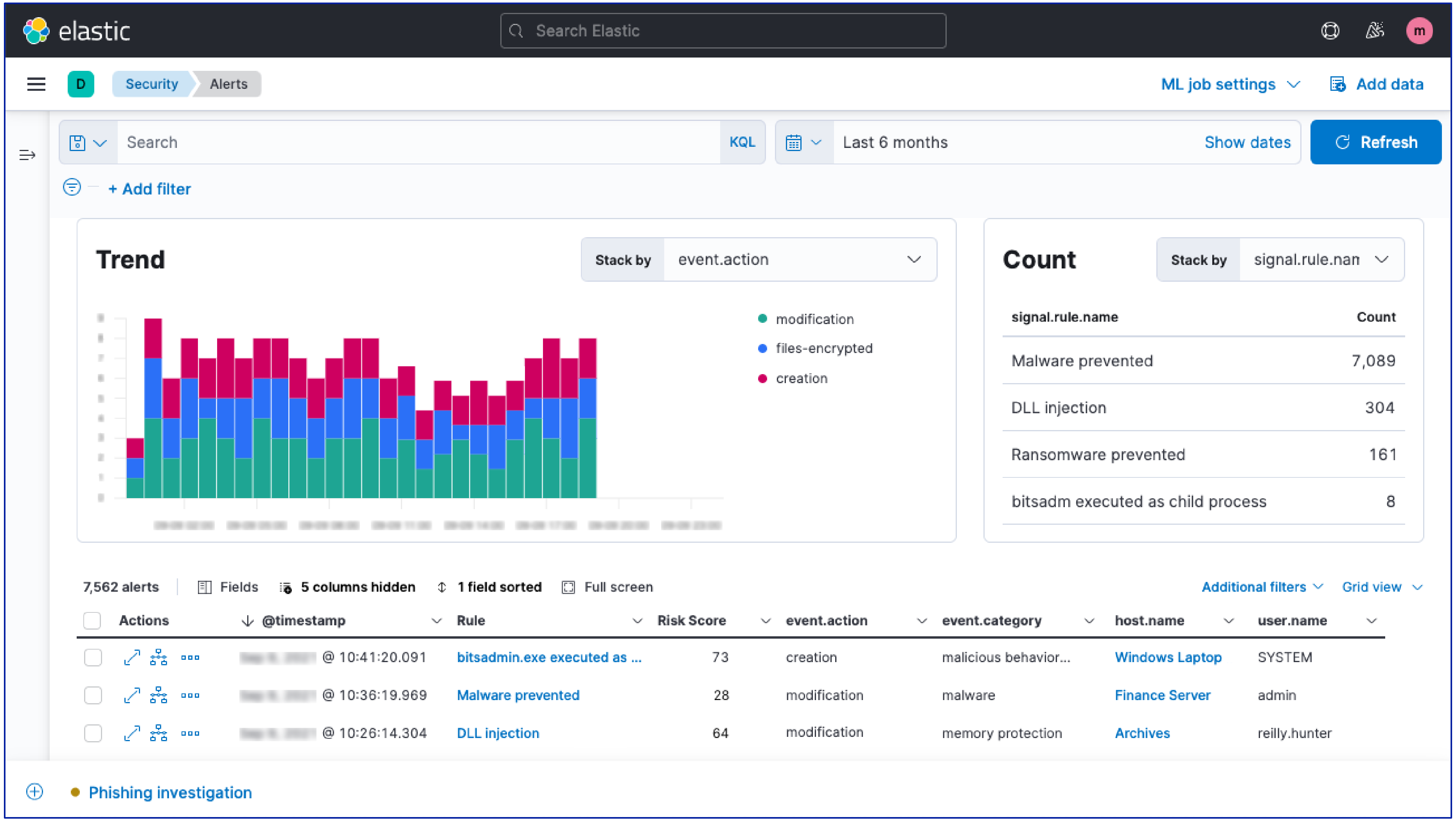
My experience – Elastic Security

At the heart of any SIEM (Security Information and Event Management) system is log data. A lot of it. Whether from servers, firewalls, databases, or network routers — logs provide analysts with the raw material for gaining insight into events taking place in an IT environment.

Elastic Security combines Elastic SIEM, whose detection engine automates threat detection so you can quickly investigate and respond to threats, and Endpoint Security into a single solution that unifies prevention, detection, and response across your entire network.



My experience – Elastic Security



signal.rule.name	Count
Malware prevented	7,089
DLL injection	304
Ransomware prevented	161
bitsadm executed as child process	8

7,562 alerts

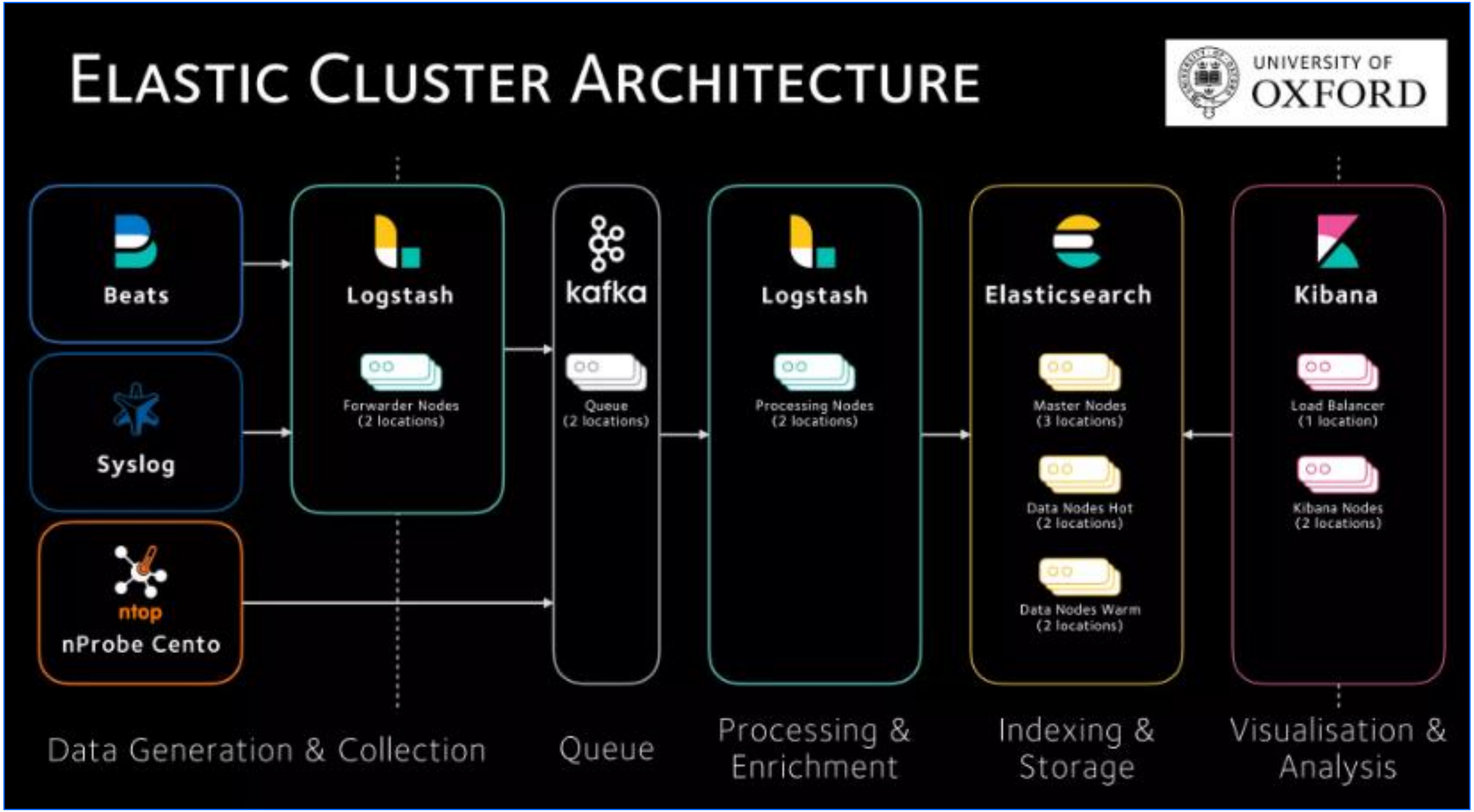
□	Actions	@timestamp	Rule	Risk Score	event.action	event.category	host.name	user.name
□		[redacted] @ 10:41:20.091	bitsadmin.exe executed as ...	73	creation	malicious behavior...	Windows Laptop	SYSTEM
□		[redacted] @ 10:36:19.969	Malware prevented	28	modification	malware	Finance Server	admin
□		[redacted] @ 10:26:14.304	DLL injection	64	modification	memory protection	Archives	reilly.hunter

Real life examples

UNIVERSITY OF OXFORD

Oxford operates one of the largest private networks in the country with ~100,000 registered devices and redundant 40GBit/s Internet uplink. University employs 800 members of IT staff across over 110 units and 17 security experts split across two teams that cover all five functions: identify, protect, detect, respond, and recover.

In 2015 they made PoC using Elasticsearch 1.5.1 and in 2016 they had first production service using ES 2.3. In 2019 they moved to Platinum subscription of Elastic stack.



ELASTIC CLUSTER ARCHITECTURE
UNIVERSITY OF OXFORD
Beats
Syslog
ntop
nProbe Cento
Logstash
Forwarder Nodes
(2 locations)
kafka
Queue
(2 locations)
Logstash
Processing Nodes
(2 locations)
Elasticsearch
Master Nodes
(3 locations)
Data Nodes Hot
(2 locations)
Data Nodes Warm
(2 locations)
Kibana
Load Balancer
(1 location)
Kibana Nodes
(2 locations)
Data Generation & Collection
Queue
Processing & Enrichment
Indexing & Storage
Visualisation & Analysis

Real life examples

UNIVERSITY OF OXFORD

Oxford operates one of the largest private networks in the country with ~100,000 registered devices and redundant 40GBit/s Internet uplink. University employs 800 members of IT staff across over 110 units and 17 security experts split across two teams that cover all five functions: identify, protect, detect, respond, and recover.

In 2015 they made PoC using Elasticsearch 1.5.1 and in 2016 they had first production service using ES 2.3. In 2019 they moved to Platinum subscription of Elastic stack.

U.S. AIR FORCE

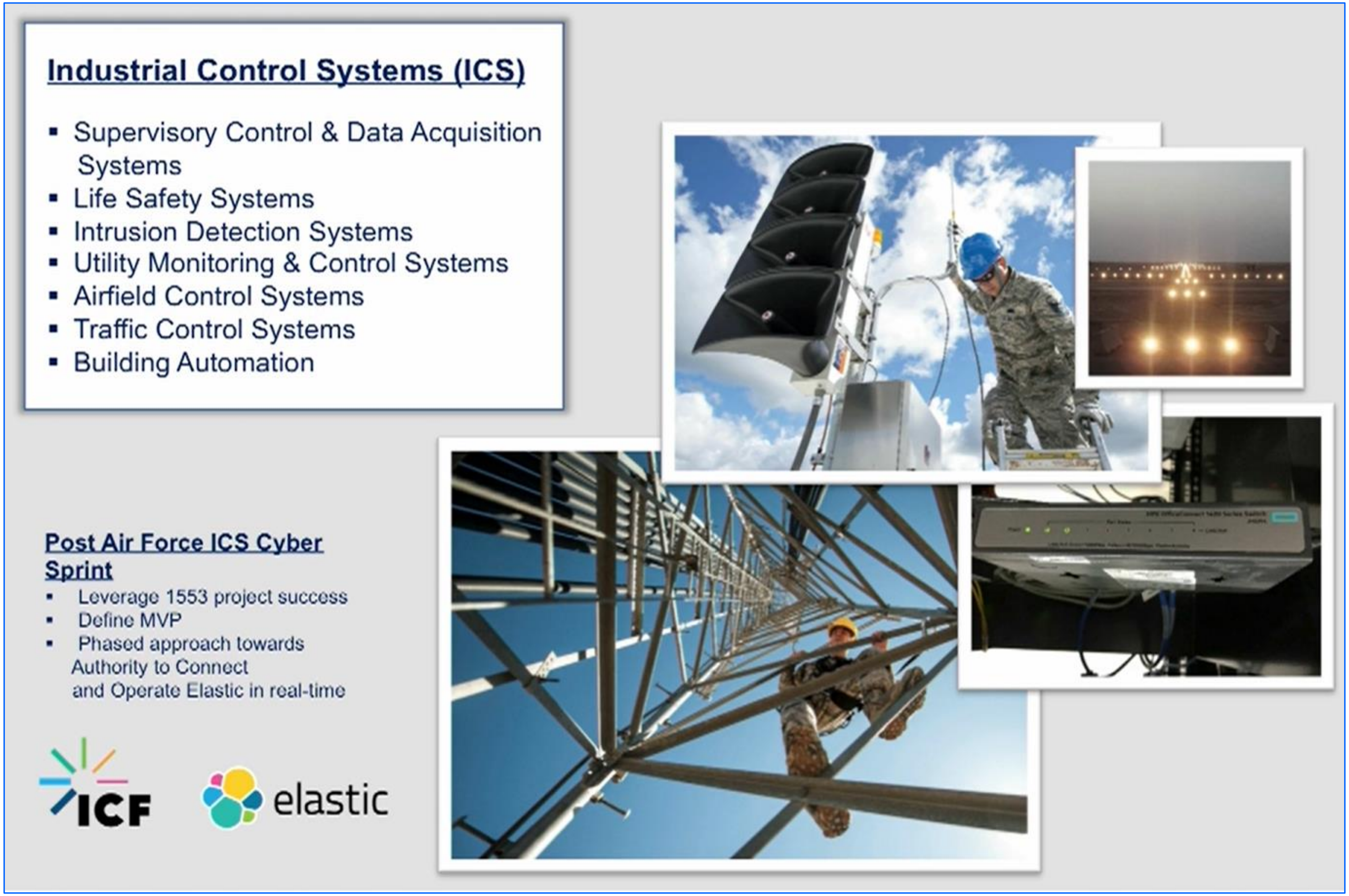
In 2017 the department launched Misison Defense Team (MDT). Their responsibility was to organize, train, and equipt the cyber defense force with the most effective and relevant tools to perform their missions.

Elastic is a part of the toolset provided for Defensive Cyber Operations that Air Combat Command has successfully leveraged for various missions.

The **E-3 Sentry, or AWACS**, is an airborne warning and control system aircraft with an integrated command and control battle management, surveillance, target detection, and tracking platform. AWACS provides situational awareness of friendly, neutral and hostile activity, command and control of an area of responsibility, battle management of theater forces and early warning of enemy actions during joint, allied, and coalition operations.

1553 Data Bus Threat Detection Efforts

- Proof-of-Concept Data baseline
- Initial goal to gain Interim Authority to Test
- Phased towards Authority to Connect and Operate Elastic in real-time



Industrial Control Systems (ICS)

- Supervisory Control & Data Acquisition Systems
- Life Safety Systems
- Intrusion Detection Systems
- Utility Monitoring & Control Systems
- Airfield Control Systems
- Traffic Control Systems
- Building Automation

Post Air Force ICS Cyber Sprint

- Leverage 1553 project success
- Define MVP
- Phased approach towards Authority to Connect and Operate Elastic in real-time

Real life examples

UNIVERSITY OF OXFORD

Oxford operates one of the largest private networks in the country with ~100,000 registered devices and redundant 40GBit/s Internet uplink. University employs 800 members of IT staff across over 110 units and 17 security experts split across two teams that cover all five functions: identify, protect, detect, respond, and recover.

In 2015 they made PoC using Elasticsearch 1.5.1 and in 2016 they had first production service using ES 2.3. In 2019 they moved to Platinum subscription of Elastic stack.

U.S. AIR FORCE

In 2017 the department launched Misison Defense Team (MDT). Their responsibility was to organize, train, and equipt the cyber defense force with the most effective and relevant tools to perform their missions.

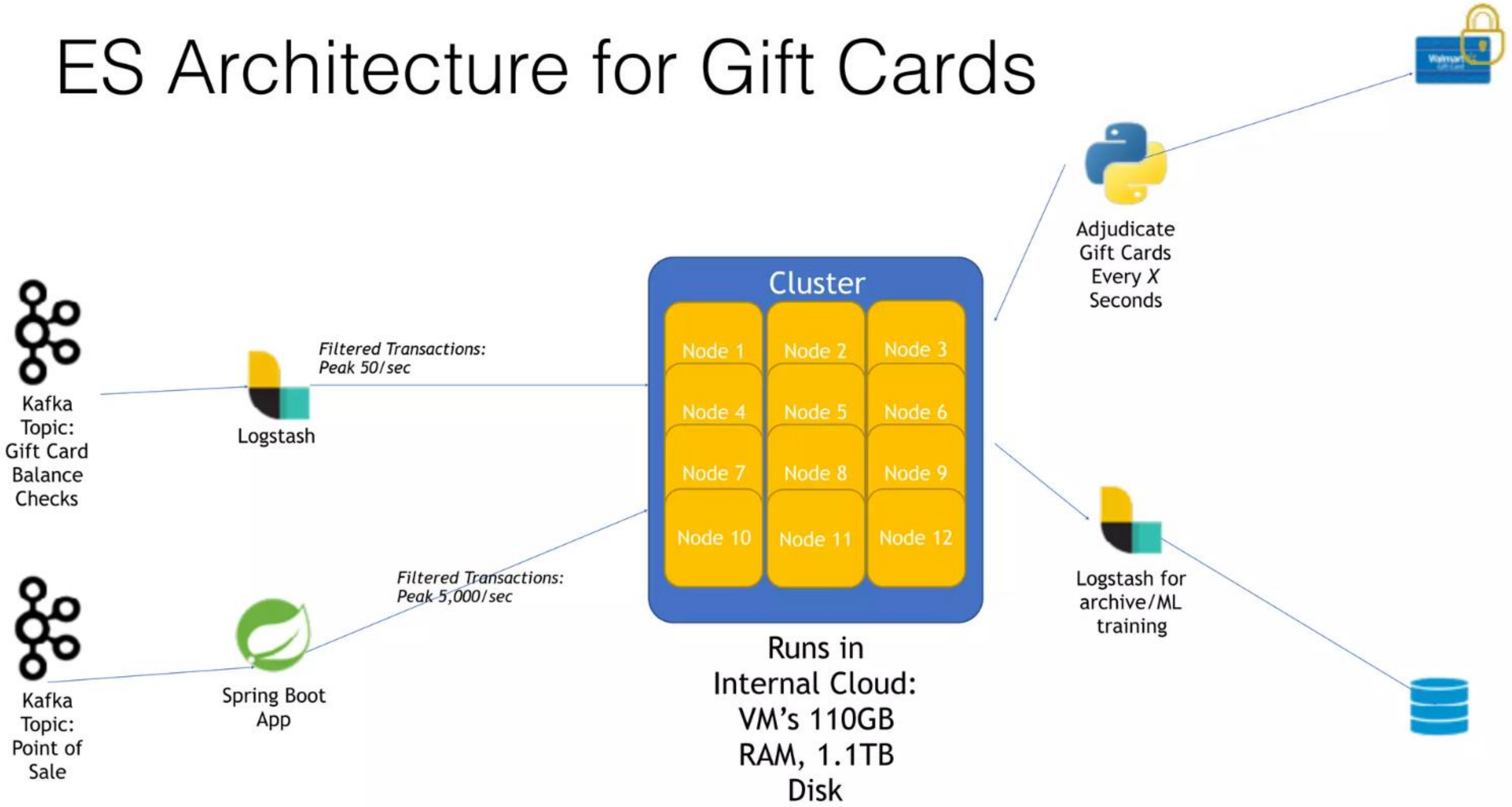
Elastic is a part of the toolset provided for Defensive Cyber Operations that Air Combat Command has successfully leveraged for various missions.

Walmart Technology

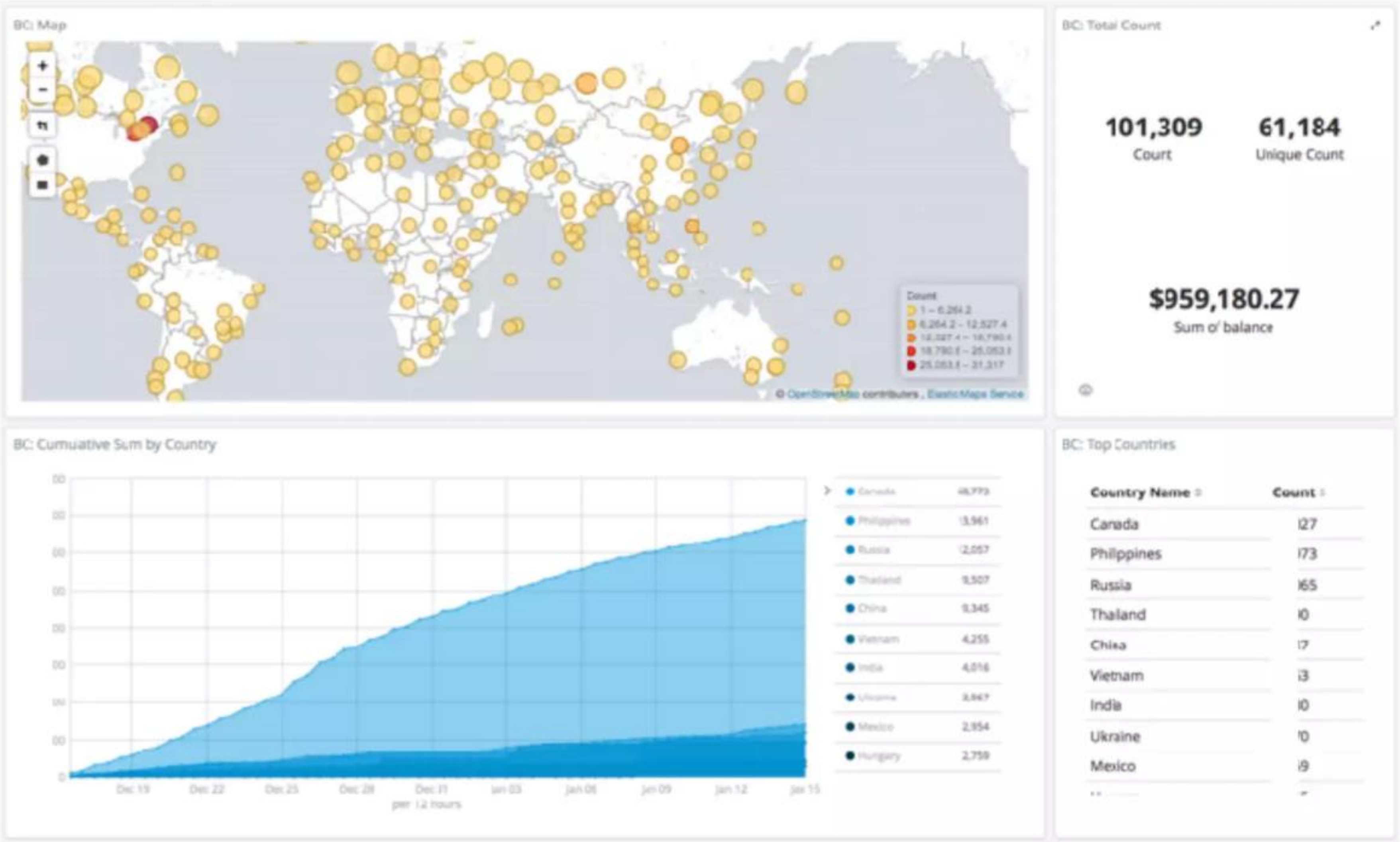
With the Elastic Stack, Walmart has ingested more than 4 billion metadata records over the past 5 years to combat fraud and protect customers. By ingesting IP address, point of sale, and other traffic data into Elasticsearch, the Walmart Global Risk Analysis team can identify instances of fraud in real time, especially gift card scams targeted at senior citizens.

In result, Walmart protected millions in customer dollars, made fraud harder to commit, and was able to automate a 24/7 protection system w/out humans in the loop.

ES Architecture for Gift Cards


Kafka Topic: Gift Card Balance Checks
Logstash
Filtered Transactions: Peak 50/sec
Kafka Topic: Point of Sale
Spring Boot App
Filtered Transactions: Peak 5,000/sec
Cluster
Node 1
Node 2
Node 3
Node 4
Node 5
Node 6
Node 7
Node 8
Node 9
Node 10
Node 11
Node 12
Runs in Internal Cloud: VM's 110GB RAM, 1.1TB Disk
Adjudicate Gift Cards Every X Seconds
Logstash for archive/ML training

Gift Card Balance Check I



BC: Total Count

101,309
Count

61,184
Unique Count

$959,180.27
Sum of balance

BC: Cumulative Sum by Country

● Canada	46,773
● Philippines	13,961
● Russia	12,057
● Thailand	9,507
● China	9,345
● Vietnam	4,255
● India	4,016
● Ukraine	3,967
● Mexico	2,954
● Hungary	2,759

BC: Top Countries

Country Name ⇕	Count ⇕
Canada	27
Philippines	73
Russia	65
Thailand	0
China	7
Vietnam	3
India	0
Ukraine	0
Mexico	9

Similar solutions

meilisearch

- Open source solution accessible to everyone and requiring very little configuration to be installed, yet highly customizable;
- Instant search experience including **typo handling, filters, custom rankings, etc.**;
- Answers <50 milliseconds;
- Stands out by being open source (while commercial), and aims to be simpler to deploy and maintain than other competitors.

Github: MeiliSearch

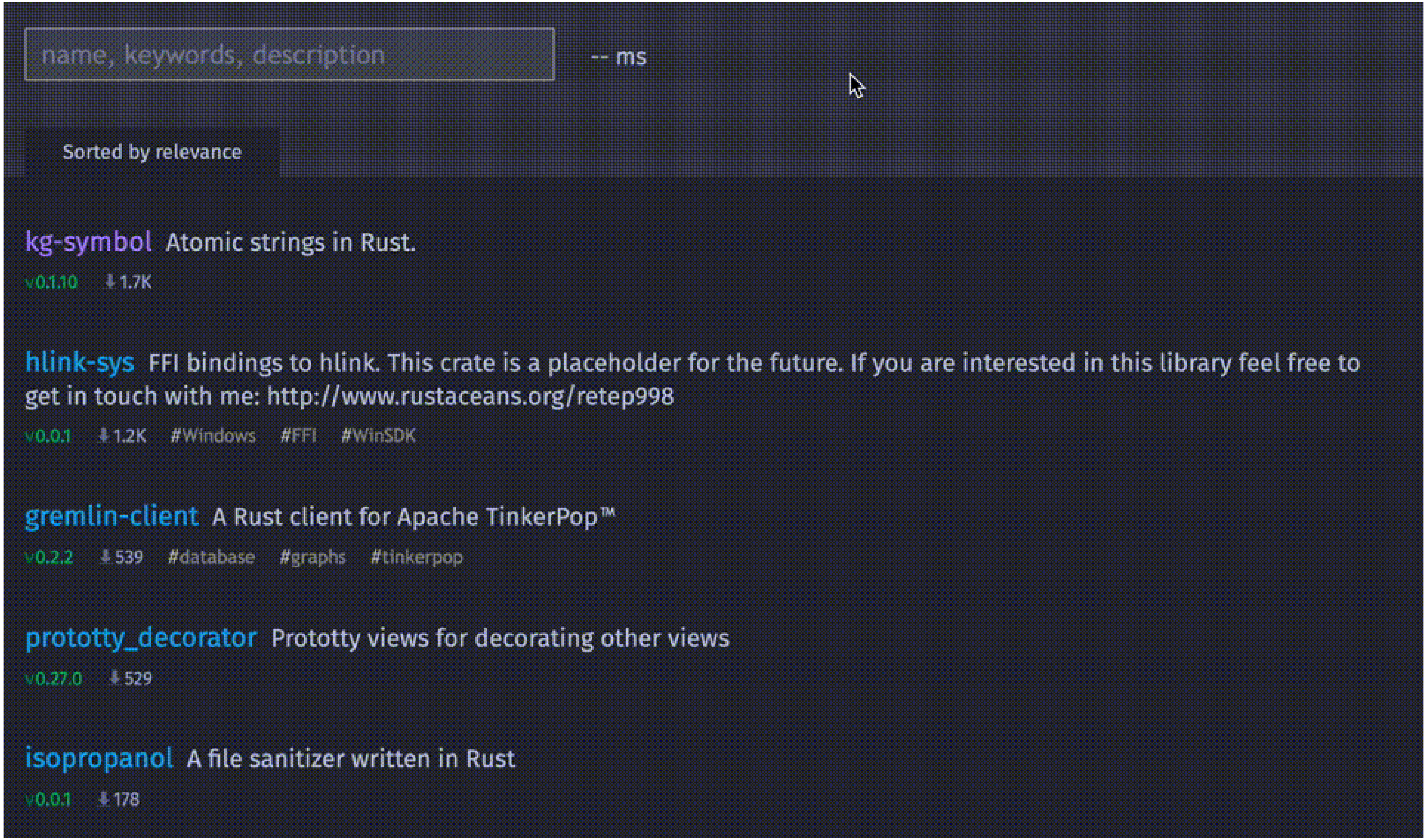
- **Sonic** can be used as a simple alternative to super-heavy and full-featured search backends such as Elasticsearch in some use-cases;
- Capable of **normalizing natural language search queries, auto-completing a search query** and **providing the most relevant results for a query**;
- Strong attention to performance and code cleanliness;
- Crash-free, super-fast and puts minimum strain on server resources.

Github: Sonic

manticoresearch

- Fork of an open source version of Sphinx Search;
- Can be used as an alternative to Elasticsearch for both full-text search and data analytics;
- Able to **parallelize the search query to all CPU cores** unconditionally and by default;
- Faster search and data ingestion than elasticsearch.

Github: manticoresearch



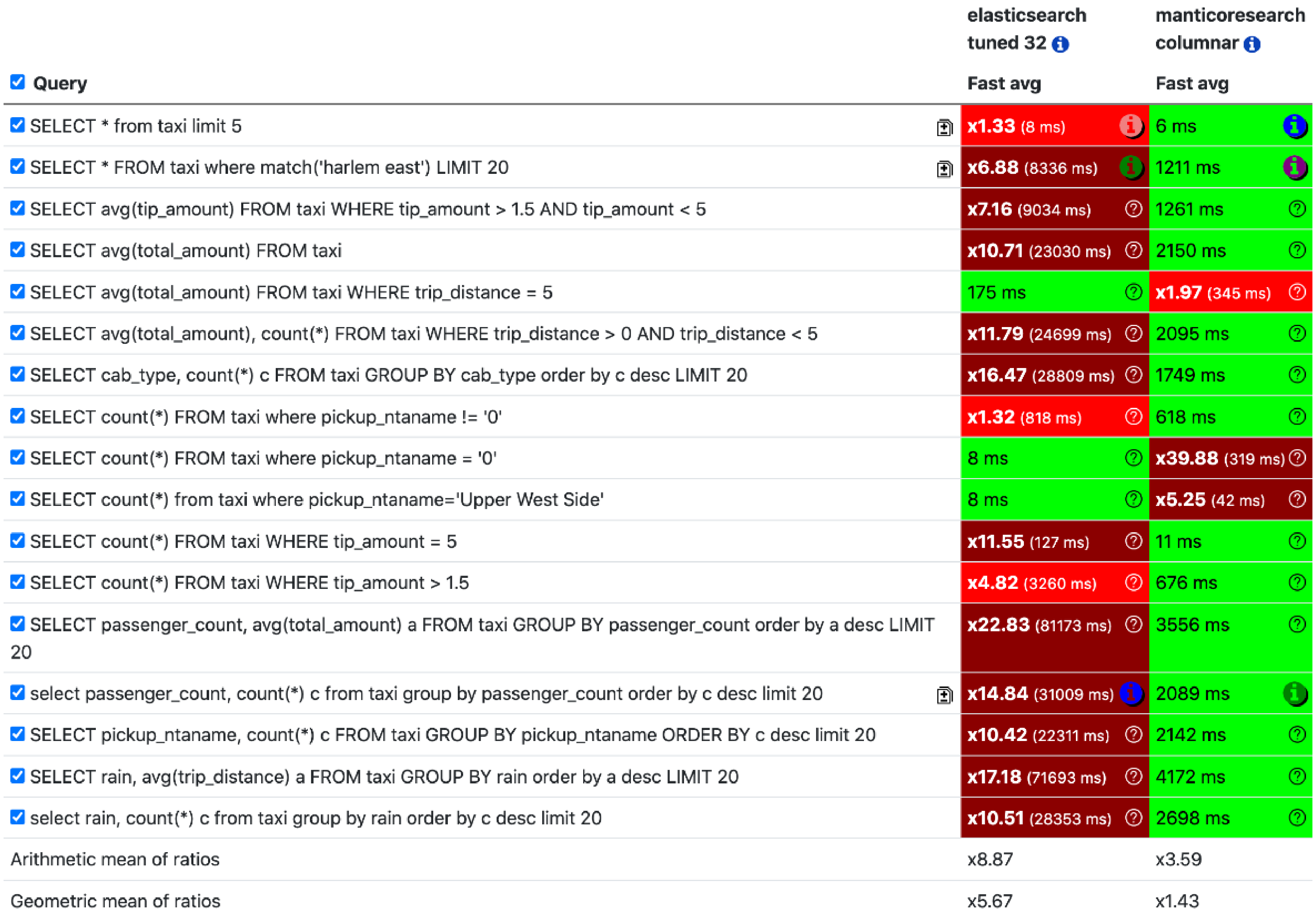
kg-symbol Atomic strings in Rust.
v0.1.10 ⬇ 1.7K

hlink-sys FFI bindings to hlink. This crate is a placeholder for the future. If you are interested in this library feel free to get in touch with me: http://www.rustaceans.org/retep998
v0.0.1 ⬇ 1.2K #Windows #FFI #WinSDK

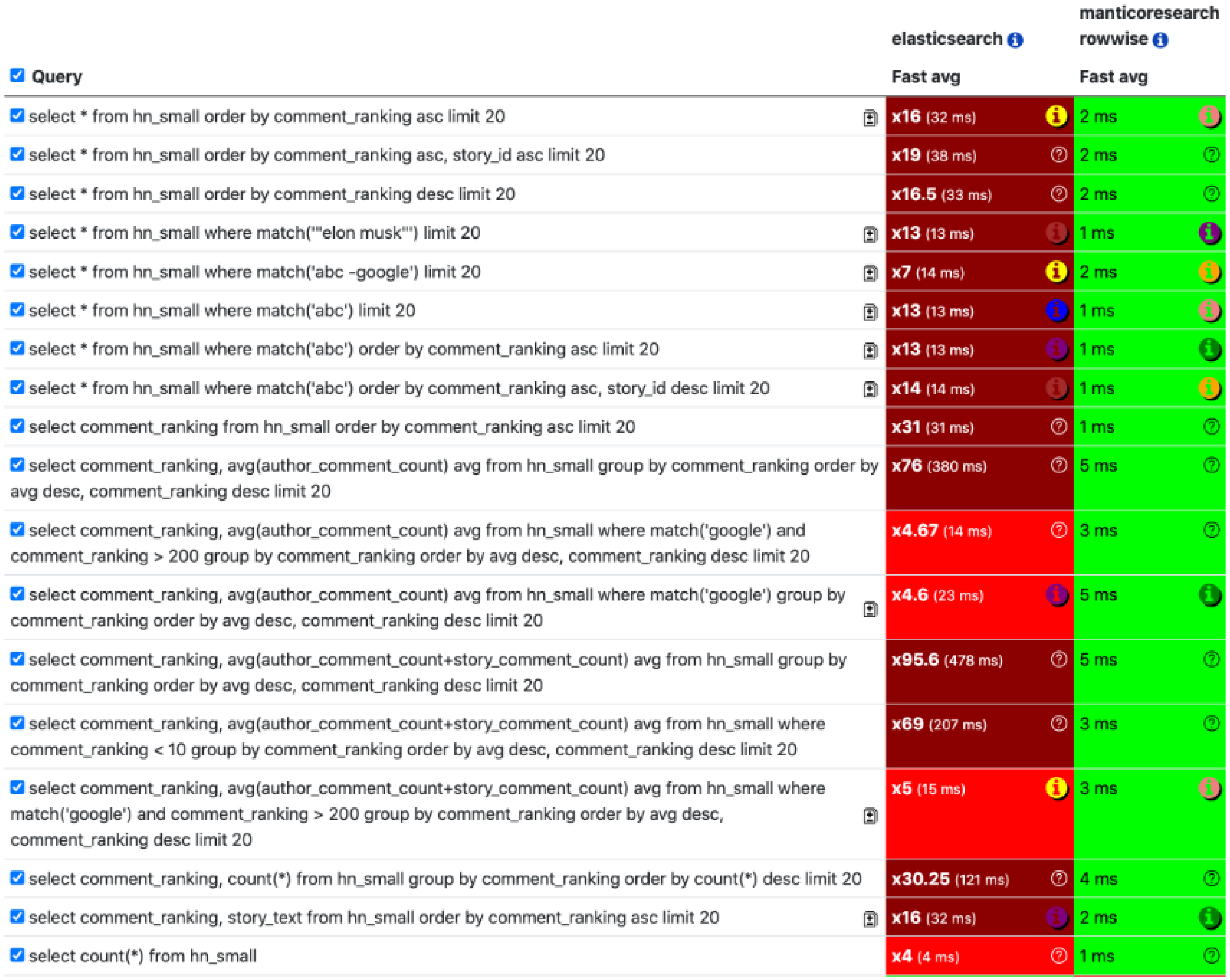
gremlin-client A Rust client for Apache TinkerPop™
v0.2.2 ⬇ 539 #database #graphs #tinkerpop

prototty_decorator Prototty views for decorating other views
v0.27.0 ⬇ 529

isopropanol A file sanitizer written in Rust
v0.0.1 ⬇ 178



☑ Query	elasticsearch tuned 32 Fast avg	manticoresearch columnar Fast avg
☑ SELECT * from taxi limit 5	**x1.33** (8 ms)	6 ms
☑ SELECT * FROM taxi where match('harlem east') LIMIT 20	**x6.88** (8336 ms)	1211 ms
☑ SELECT avg(tip_amount) FROM taxi WHERE tip_amount > 1.5 AND tip_amount < 5	**x7.16** (9034 ms)	1261 ms
☑ SELECT avg(total_amount) FROM taxi	**x10.71** (23030 ms)	2150 ms
☑ SELECT avg(total_amount) FROM taxi WHERE trip_distance = 5	175 ms	**x1.97** (345 ms)
☑ SELECT avg(total_amount), count(*) FROM taxi WHERE trip_distance > 0 AND trip_distance < 5	**x11.79** (24699 ms)	2095 ms
☑ SELECT cab_type, count(*) c FROM taxi GROUP BY cab_type order by c desc LIMIT 20	**x16.47** (28809 ms)	1749 ms
☑ SELECT count(*) FROM taxi where pickup_ntaname != '0'	**x1.32** (818 ms)	618 ms
☑ SELECT count(*) FROM taxi where pickup_ntaname = '0'	8 ms	**x39.88** (319 ms)
☑ SELECT count(*) from taxi where pickup_ntaname='Upper West Side'	8 ms	**x5.25** (42 ms)
☑ SELECT count(*) FROM taxi WHERE tip_amount = 5	**x11.55** (127 ms)	11 ms
☑ SELECT count(*) FROM taxi WHERE tip_amount > 1.5	**x4.82** (3260 ms)	676 ms
☑ SELECT passenger_count, avg(total_amount) a FROM taxi GROUP BY passenger_count order by a desc LIMIT 20	**x22.83** (81173 ms)	3556 ms
☑ select passenger_count, count(*) c from taxi group by passenger_count order by c desc limit 20	**x14.84** (31009 ms)	2089 ms
☑ SELECT pickup_ntaname, count(*) c FROM taxi GROUP BY pickup_ntaname ORDER BY c desc limit 20	**x10.42** (22311 ms)	2142 ms
☑ SELECT rain, avg(trip_distance) a FROM taxi GROUP BY rain order by a desc LIMIT 20	**x17.18** (71693 ms)	4172 ms
☑ select rain, count(*) c from taxi group by rain order by c desc limit 20	**x10.51** (28353 ms)	2698 ms
Arithmetic mean of ratios	x8.87	x3.59
Geometric mean of ratios	x5.67	x1.43



	elasticsearch	manticoresearch rowwise
☑ **Query**	**Fast avg**	**Fast avg**
☑ select * from hn_small order by comment_ranking asc limit 20	**x16** (32 ms)	2 ms
☑ select * from hn_small order by comment_ranking asc, story_id asc limit 20	**x19** (38 ms)	2 ms
☑ select * from hn_small order by comment_ranking desc limit 20	**x16.5** (33 ms)	2 ms
☑ select * from hn_small where match('"elon musk"') limit 20	**x13** (13 ms)	1 ms
☑ select * from hn_small where match('abc -google') limit 20	**x7** (14 ms)	2 ms
☑ select * from hn_small where match('abc') limit 20	**x13** (13 ms)	1 ms
☑ select * from hn_small where match('abc') order by comment_ranking asc limit 20	**x13** (13 ms)	1 ms
☑ select * from hn_small where match('abc') order by comment_ranking asc, story_id desc limit 20	**x14** (14 ms)	1 ms
☑ select comment_ranking from hn_small order by comment_ranking asc limit 20	**x31** (31 ms)	1 ms
☑ select comment_ranking, avg(author_comment_count) avg from hn_small group by comment_ranking order by avg desc, comment_ranking desc limit 20	**x76** (380 ms)	5 ms
☑ select comment_ranking, avg(author_comment_count) avg from hn_small where match('google') and comment_ranking > 200 group by comment_ranking order by avg desc, comment_ranking desc limit 20	**x4.67** (14 ms)	3 ms
☑ select comment_ranking, avg(author_comment_count) avg from hn_small where match('google') group by comment_ranking order by avg desc, comment_ranking desc limit 20	**x4.6** (23 ms)	5 ms
☑ select comment_ranking, avg(author_comment_count+story_comment_count) avg from hn_small group by comment_ranking order by avg desc, comment_ranking desc limit 20	**x95.6** (478 ms)	5 ms
☑ select comment_ranking, avg(author_comment_count+story_comment_count) avg from hn_small where comment_ranking < 10 group by comment_ranking order by avg desc, comment_ranking desc limit 20	**x69** (207 ms)	3 ms
☑ select comment_ranking, avg(author_comment_count+story_comment_count) avg from hn_small where match('google') and comment_ranking > 200 group by comment_ranking order by avg desc, comment_ranking desc limit 20	**x5** (15 ms)	3 ms
☑ select comment_ranking, count(*) from hn_small group by comment_ranking order by count(*) desc limit 20	**x30.25** (121 ms)	4 ms
☑ select comment_ranking, story_text from hn_small order by comment_ranking asc limit 20	**x16** (32 ms)	2 ms
☑ select count(*) from hn_small	**x4** (4 ms)	1 ms

Questions

Objectivity

WYDZIAŁ ELEKTRONIKI,
TELEKOMUNIKACJI
I INFORMATYKI

Thank you for your attention

If you want to know more, please contact me.

LinkedIn: damianstrojek

GitHub: damianStrojek

Work: dstrojek@objectivity.co.uk

www.objectivity.co.uk | www.objectivity.de