Making story from system logs with elastic stack
Logs
syslog
NETFLOW
METRIC
SNMP
Audit
dns
http
ids
What is Elastic Stack?

- **Store, Analyze**: elasticsearch
- **Ingest**: logstash, beats
- **User Interface**: kibana
Elasticsearch is a full-text based, distributed NoSQL database.

Written in Java, built on Apache Lucene

Commonly used for log analytics, full-text search, security intelligence, business analytics, and operational intelligence use cases.

Use REST API (GET, PUT, POST, and DELETE) for storing and searching data.
Data is stored as documents
(rows in relational database)

Data is separated into fields
(columns in relational database)
<table>
<thead>
<tr>
<th>Relational Database</th>
<th>Elasticsearch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database</td>
<td>Index</td>
</tr>
<tr>
<td>Table</td>
<td>Type</td>
</tr>
<tr>
<td>Row/Record</td>
<td>Document</td>
</tr>
<tr>
<td>Column Name</td>
<td>Field</td>
</tr>
</tbody>
</table>
**Terminology**

**Cluster:** A cluster consists of one or more nodes which share the same *cluster name*.

**Node:** A node is a running instance of Elasticsearch which belongs to a cluster.
**Index:** Collection of documents

**Shard:** An index is split into elements known as shards that are distributed across multiple nodes. There are two types of shard, Primary and replica. By default elasticsearch creates 1 primary shard and 1 replica shard for each index.
Elasticsearch Terminology

Cluster

Node 1
- Shard 1
- Replica 2
- Replica 1

Node 2
- Shard 2
Documents

• Indices hold **documents** in serialized **JSON objects**
• 1 document = 1 log entry
• Contains "field : value" pairs
• Metadata
  • **_index** – Index the document belongs to
  • **_id** – unique ID for that log
  • **_source** – parsed log fields

```json
{
    "_index": "netflow-2020.10.08",
    "_type": "_doc",
    "_id": "ZwkiB3UBULotwSOX3Bdb",
    "_version": 1,
    "_score": null,
    "_source": {
        "@timestamp": "2020-10-08T07:35:32.000Z",
        "host": "172.20.0.1",
        "netflow": {
            "ipv4_dst_addr": "103.12.179.136",
            "l4_dst_port": 80,
            "src_tos": 0,
            "l4_src_port": 53966,
            "ipv4_src_addr": "192.168.110.18",
            "application_id": "13..0",
            "version": 9,
        }
    }
}
```
Elasticsearch Index creation

- netflow-2020.10.08
- netflow-2020.10.09
- syslog-2020.10.08
- syslog-2020.10.09
Shards and Documents

Index → Shards → Documents
Hosted Elasticsearch:

Elastic cloud, AWS, GCP and Azure. Nothing to install, just login and run instances. Free 14 day trial.

Own hardware:

Linux and MacOS tar.gz archive
Windows .zip archive
deb, rpm, msi, brew, docker
Install JAVA

curl -fsSL https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -

echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list

sudo apt update

sudo apt install elasticsearch
Installation (docker example)

Pull the image

docker pull docker.elastic.co/elasticsearch/elasticsearch:7.x

Start a single node cluster

docker run -p 9200:9200 -p 9300:9300 -e "discovery.type=single-node"
docker.elastic.co/elasticsearch/elasticsearch:7.x
**Installation** (docker example)

Multi node cluster with **docker-compose**

```yaml
es01:
  image: ${ELASTICSEARCH_IMAGE}
  container_name: es01
  restart: always
  ports:
    - "9200:9200"
  environment:
    - "ES_JAVA_OPTS=-Xms1g -Xmx1g"
  networks:
    - net
  volumes:
    - type: bind
      source: some_path_your_host
      target: some_path_your_container

es02:
  image: ${ELASTICSEARCH_IMAGE}
  container_name: es02
  restart: always
  ports:
    - "9201:9201"
  environment:
    - "ES_JAVA_OPTS=-Xms1g -Xmx1g"
  networks:
    - net
  volumes:
    - type: bind
      source: some_path_your_host
      target: some_path_your_container
```
Location and the config file

/usr/share/elasticsearch/config/elasticsearch.yml

Key configuration elements

**Single node**

- network.host: 0.0.0.0
- http.port: 9200
- discovery.type=single-node

**Cluster**

- node.name: es01
- node.master: true
- cluster.name: training
- discovery.seed_hosts: es02
- cluster.initial_master_nodes: es01,es02
- network.host: 0.0.0.0
[root@4f8cd6658b1b elasticsearch]# curl http://localhost:9200

{
    "name": "es01",
    "cluster_name": "training",
    "cluster_uuid": "vE9SZr8oRFK0A0HTq9U_oA",
    "version": {
        "number": "7.7.0",
        "build_flavor": "default",
        "build_type": "docker",
        "build_hash": "81a1e9eda8e6183f5237786246f6dced26a10eaf",
        "build_date": "2020-05-12T02:01:37.602180Z",
        "build_snapshot": false,
        "lucene_version": "8.5.1",
        "minimum_wire_compatibility_version": "6.8.0",
        "minimum_index_compatibility_version": "6.0.0-beta1"
    },
    "tagline": "You Know, for Search"
}
logstash

Free, developed and maintained by Elastic

Integrates with Beats

Integrates with Elasticsearch

Tons of plugins
Logstatsh has three stages

INPUT

input {
  tcp {
    port => 5002
    type => "syslog"
  }
}

FILTER

filter {
  if [type] == "syslog" {
    grok {
    }
  }
}

OUTPUT

output {
  if [type] == "syslog" {
    elasticsearch {
      hosts => "http://es01:9200"
      index => "syslog-%{+YYYY.MM.dd}"
    }
  }
}

beats, file, syslog, udp, snmp, etc...

http, kv, xml, json, etc...

csv, file, http, stdout, etc....
Grok is a great way to parse unstructured log data into something structured and queryable.

The syntax for a grok pattern is `{SYNTAX:SEMANTIC}`

**SYNTAX**: is the name of the pattern that will match your text

**SEMANTIC**: is the identifier to the piece of text being matched
raw log

192.168.8.1 GET /index.html 15824 0.04

grok pattern

{%IP:client} {%WORD:method} {%URIPATHPARAM:request} {%NUMBER:bytes} {%NUMBER:duration}

output

{
  "duration": "0.04",
  "request": "/index.html",
  "method": "GET",
  "bytes": "15824",
  "client": "192.168.8.1"
}
Sample Data

```
1  192.168.8.1 GET /index.html 15824 0.04
```

Grok Pattern

```
1  %{IP:client} %{WORD:method} %{URIPATHPARAM:request} %{NUMBER:bytes} %{NUMBER:duration}
```

Custom Patterns

Simulate

Structured Data

```
1  {
2   "duration": "0.04",
3   "request": "/index.html",
4   "method": "GET",
5   "bytes": "15824",
6   "client": "192.168.8.1"
7  }
```
Installation

From binaries
Download the package from: https://www.elastic.co/downloads/logstash

Options are: tar.gz, deb, zip, rpm

Package manager: yum, apt-get, homebrew

Container: docker
**logstash **Installation **(Ubuntu example)**

**Install JAVA**

```bash
wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch
| sudo apt-key add -

sudo apt-get install apt-transport-https

echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main"
| sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list

sudo apt update

sudo apt install logstash
```
Installation (docker example)

Pull the image

docker pull docker.elastic.co/logstash/logstash:7.x

Start the container

docker run --rm -it -v
~/settings/:/usr/share/logstash/config/
docker.elastic.co/logstash/logstash:7.x

Dockerfile

FROM docker.elastic.co/logstash/logstash:7.x
RUN rm -f /usr/share/logstash/pipeline/logstash.conf
ADD pipeline/ /usr/share/logstash/pipeline/
ADD config/ /usr/share/logstash/config/
Logs for single host

```
vagrant@logger:~/suricata-update$ tail -f /var/log/suricata/fast.log
.15:33726 -> 91.189.00.149:80
.15:33726 -> 91.189.88.149:80
.15:33726 -> 91.189.88.149:80
.15:33726 -> 91.189.00.149:80
.15:33726 -> 91.189.88.149:80
.15:33726 -> 91.189.88.149:80
.15:33726 -> 91.189.00.149:80
.15:33726 -> 91.189.88.149:80
.15:33726 -> 91.189.88.149:80
.15:33726 -> 91.189.00.149:80
```
Command line logs

- cat
- tail
- grep
- vi/ vim/ nano /event viewer
216.245.66.239 -- [05/Jan/2018:05:09:26 -0700] "GET /wp-content/ VendingMachine.jpg HTTP/1.1" 200 195309 - "Mozilla/5.0 (X;
1; http://www.opensiteexplorer.org/dotbot, help@moz.com)"
216.245.66.239 -- [05/Jan/2018:05:08:25 -0700] "GET /the-direc
to-archivne/ HTTP/1.1" 200 74500 - "Mozilla/5.0 (compatib
leb/11; www.opensiteexplorer.org/dotbot, help@moz.com)"
" "Feedin feed-id:481336 - 13 subscribers"
" "Feedin feed-id:481336 - 13 subscribers"
st-suite-management-and-build-integration/feed HTTP/1.1" 301
0 (X11; Linux 6866 AppleWebKit/537.36 (KHTML, like Gecko) C
safari/537.36)"
st-suite-management-and-build-integration/feed HTTP/1.1" 200
5 (X11; Linux 6866 AppleWebKit/537.36 (KHTML, like Gecko) C
safari/537.36)"
66.249.93.53 -- [05/Jan/2018:05:09:02 -0700] "GET /sofware-
odie-business/ HTTP/1.1" 200 18770 - "Mozilla/5.0 (X11; Linux
6866 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/49.0.2623.75 Safari/537.36"
84.30.36.214 -- [05/Jan/2018:05:09:02 -0700] "GET /feed HTTP/1.1" 301 466 "-" "Tiny Tiny RSS 16.8 (http://tt-rss.org)"

---More---(05)
kibana

Not available on cli

- Multiple source
- Corelating
- Searching, filtering
- Visualize
Installation

Hosted Kibana:

Elastic cloud, AWS, GCP and Azure. Nothing to install, just login and run instances. Free 14 day trial.

Own hardware:

Linux and MacOS tar.gz archive
Windows .zip archive
deb, rpm, msi, brew, docker
kibana Installation (Ubuntu example)

wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch |
| sudo apt-key add -

sudo apt-get install apt-transport-https

echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" |
| sudo tee -a /etc/apt/sources.list.d/elastic-7.x.list

sudo apt update

sudo apt install kibana
Installation (Docker example)

Pull the image

docker pull docker.elastic.co/kibana/kibana:7.x

Start the container

docker run --link ELASTICSEARCH_CONTAINER:elasticsearch -p 5601:5601
docker.elastic.co/kibana/kibana:7.x
**kibana** Installation *(Docker example)*

**Docker-compose**

```yaml
kibana:
  image: ${KIBANA_IMAGE}
  container_name: kibana
  restart: always
  ports:
    - "5601:5601"
  volumes:
    - type: bind
      source: ./kibana/conf/kibana.yml
      target: /usr/share/kibana/config/kibana.yml
  depends_on:
    - es01
  networks:
    - net
```

Docker-compose configuration for kibana installation.
Location and the config file

/usr/share/kibana/config/kibana.yml

Key configuration elements

server.name: kibana
server.host: "0"
elasticsearch.hosts:
  - http://es01:9200
  - http://es02:9200
bin/elasticsearch-certutil ca
bin/elasticsearch-certutil cert --ca elastic-stack-ca.p12

vim /usr/share/elasticsearch/config/elasticsearch.yml

xpack.security.enabled: true
xpack.security.transport.ssl.enabled: true
xpack.security.transport.ssl.keystore.type: PKCS12
xpack.security.transport.ssl.verification_mode: certificate
xpack.security.transport.ssl.keystore.path: elastic-certificates.p12
xpack.security.transport.ssl.truststore.path: elastic-certificates.p12
xpack.security.transport.ssl.truststore.type: PKCS12
Setup password for built-in users

bin/elasticsearch-setup-passwords auto/interactive

Changed password for user **apm_system**
PASSWORD apm_system = JreXXXXXXXXXXXXXDM2F

Changed password for user **kibana**
PASSWORD kibana = YKvXXXXXXXXXXXXXiCZ

Changed password for user **logstash_system**
PASSWORD logstash_system = jUcXXXXXXXXXXXXXNkP

Changed password for user **beats_system**
PASSWORD beats_system = uAkXXXXXXXXXXXXXv42

Changed password for user **remote_monitoring_user**
PASSWORD remote_monitoring_user = 9LdXXXXXXXXXXXXX1KC

Changed password for user **elastic**
PASSWORD elastic = GUdXXXXXXXXXXXXX8Ze
**built-in users**

These users have a fixed set of privileges and cannot be authenticate/use without setup the credentials.

**elastic: superuser**

**Kibana**: to connect and communicate with Elasticsearch

**apm_system, logstash_system, beats_system, remote_monitoring_user**: uses when storing monitoring information in Elasticsearch.
vim config/kibana.yml

server.name: kibana
server.host: "0"

elasticsearch.hosts:
  - http://es01:9200
  - http://es02:9200

elasticsearch.username: "user_namae"
elasticsearch.password: "password"
Welcome to Elastic Kibana
Your window into the Elastic Stack

Username

Password

Log in
beats

Lightweight data shippers install as agents on your servers

Available in Linux/Windows/Mac
Installation (Auditbeat)

Download and install

```
curl -L -O https://artifacts.elastic.co/downloads/beats/auditbeat/auditbeat-7.7.0-amd64.deb

sudo dpkg -i auditbeat-7.7.0-amd64.deb
```

Edit configuration (/etc/auditbeat/auditbeat.yml)

```
output.elasticsearch:
  hosts: ['es_host:9200']
  username: "elastic"
  password: "<password>"
setup.kibana:
  host: "http://kibana_host:5601"
```
beats Installation (Auditbeat)

Start auditbeat

```
sudo auditbeat setup
sudo service auditbeat start
```

Status

Check that data is received from Auditbeat

Data successfully received
Alerting

Elastalert
Elastalert

ElastAlert is a simple framework for alerting on anomalies, spikes, or other patterns of interest from data in Elasticsearch.

X events in Y time (frequency type)
rate of events increases or decreases" (spike type)
matches a blacklist/whitelist" (blacklist and whitelist type)
less than X events in Y time" (flatline type)

Email, JIRA, HipChat, MS Teams, Slack, Telegram etc..
Elastalert Installation

```bash
sudo apt-get install python-minimal

sudo apt-get install python-pip python-dev libffi-dev libssl-dev

sudo git clone https://github.com/Yelp/elastalert.git

sudo pip install "setuptools>=11.3"

sudo python setup.py install

sudo pip install "elasticsearch>=5.0.0"
```
Elastalert configuration

vim /opt/elastalert/config.yaml

es_host: elk-server
es_port: 9200
es_username: es_user
es_password: password

sudo elastalert-create-index
Demo

1. Run and explore Elastic Stack with `docker-compose`

2. Install, configure "Auditbeat" and send the logs to the Elastic

3. Configure FIM in Auditbeat

4. Alerting log event using elastalert to slack channel