Log Analysis

When CLI get's complex

ITNOG3

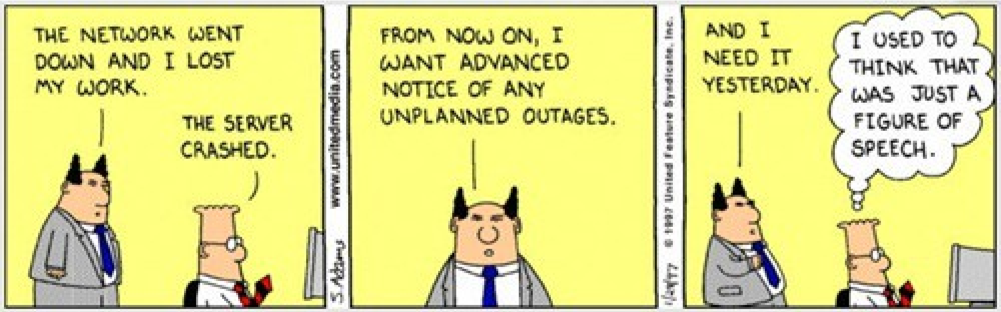
Octavio Melendres
Network admin - Fastnet Spa

Introduction

- Network engineer at Fastnet Spa from 2003
- Fastnet Spa is an ISP from Marche Region located in Ancona
- Company started in 1995 with analog modem access lines
- Today connecting companies and citizens using several technologies from wireless, DSL to fiber optics
- Uplinks to MIX and NAMEX up to 10Gbps
- Providing cloud, colocation and backup services at own Ancona datacenters

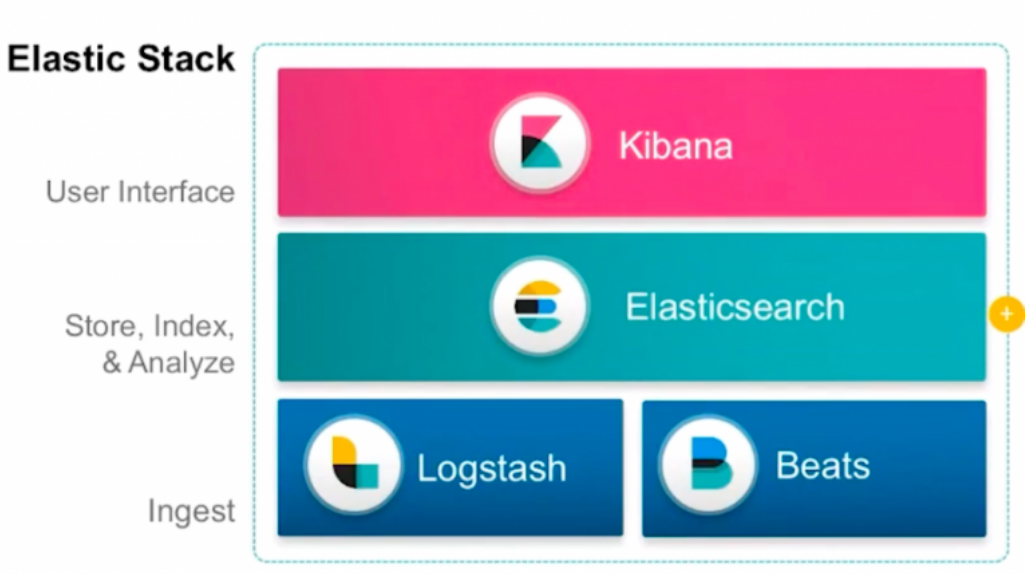
Log analysis for a network admin

Log analysis is often a challenging task. (Even for a vi expert)
Requires the analysis of great amount of data in short periods of time
Usually under pressure from management in response of a network failure or attack


THE NETWORK WENT DOWN AND I LOST MY WORK.
THE SERVER CRASHED.
www.unitedmedia.com
S. Adams
FROM NOW ON, I WANT ADVANCED NOTICE OF ANY UNPLANNED OUTAGES.
© 1997 United Feature Syndicate, Inc.
AND I NEED IT YESTERDAY.
I USED TO THINK THAT WAS JUST A FIGURE OF SPEECH.

Solution used

- There are several log analysis solutions available today!
- Most of solutions found are commercial
- Decided to use elasticsearch open source for Log Analysis
- Elasticsearch project is open source with commercial add-on modules



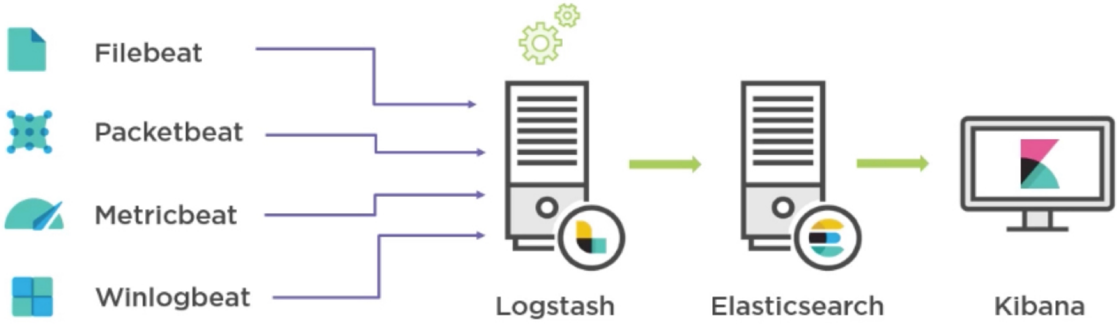
Log analysis process steps

1. Generate & collect
2. Aggregate & normalize
3. Store & optimize
4. Analysis & Alert

Generate & Collect

Generate & Collect - Send all logs!!

Log messages are generated directly by network devices and sent to Logstash module
Logdata from servers is collected using Beats package
On old servers, used sshmount from the logserver to load the files



Generate & Collect - Filebeat

Filebeat module uses a simple configuration with sections input, output

Includes several libraries with predefined file formats like: ngix, apache, mysql

Support load balancing and reliable export to multiple servers

Example configuration:

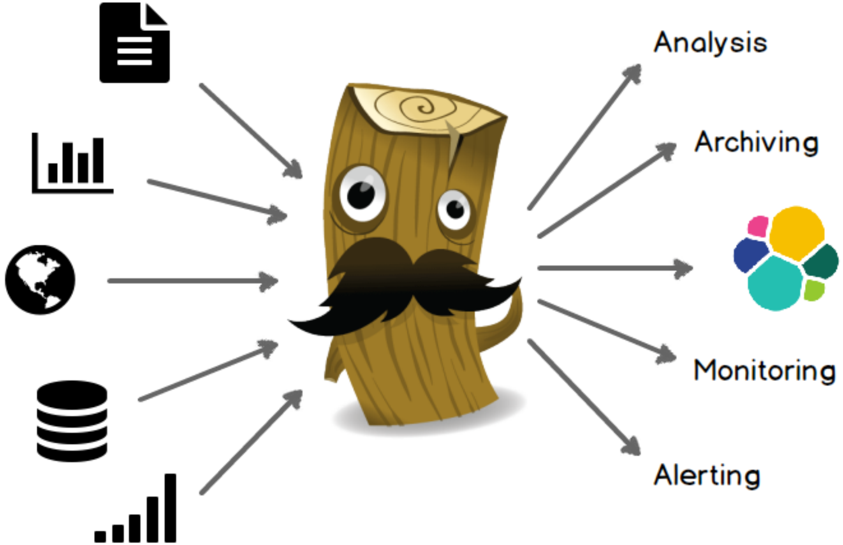
```
filebeat.prospectors:
- input_type: log
  paths:
    - /var/log/fastnetmng/*.log

output.logstash:
  hosts: ["localhost:5044"]
```

Aggregate & Normalize

Aggregate & Normalize - LOGSTASH

Logstash is a data processing pipeline
Ingest data from different sources
Transforms data
Sends reliably to elasticsearch



Aggregate & Normalize - LOGSTASH

Example configuration:

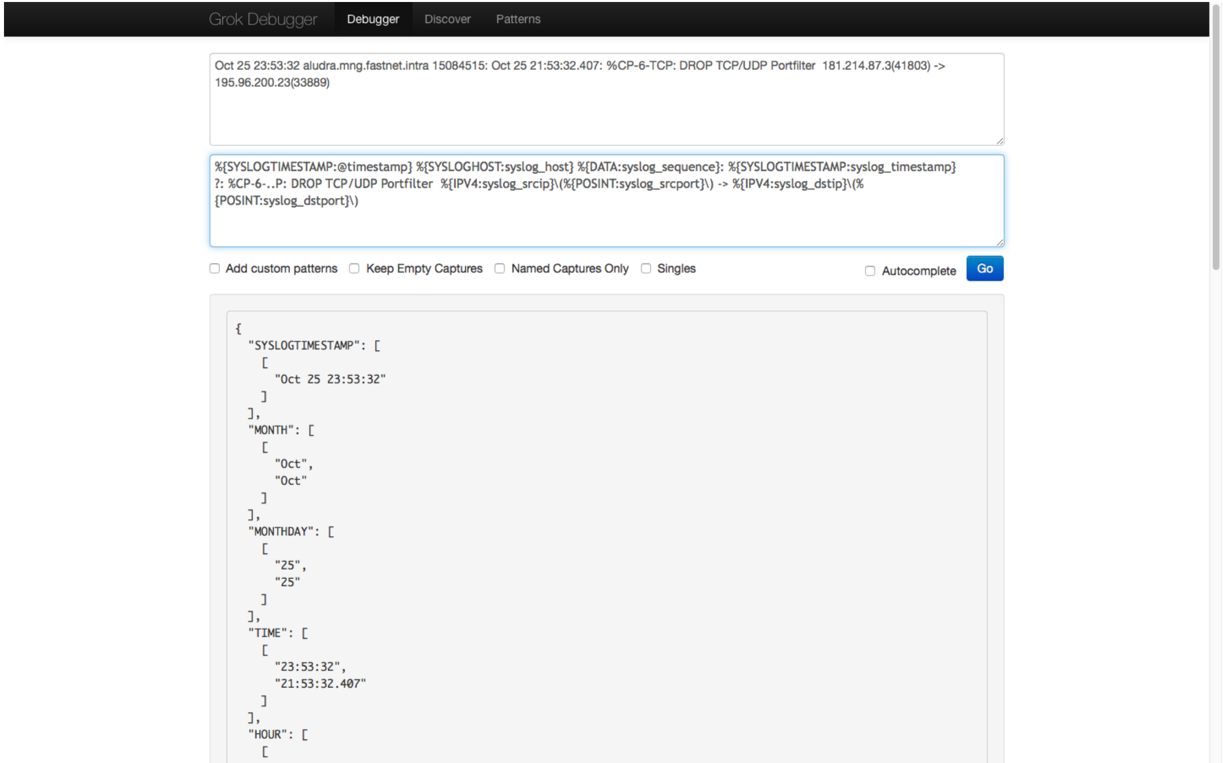
```
input {
 udp {
   port => 42186
   type => syslog
   tags => "cisco-fw"
 } }
filter {
 if  "cisco-fw" in [tags]  {
        grok {
            match => ["message", "^<%{POSINT:syslog_pri}>:%{CISCOTIMESTAMP:timestamp} ?(CEDT:|CEST:) %.*: %
{GREEDYDATA:cisco_message}"]
                        tag_on_failure => "_grokparsefailure1"
                }   }   }
output {
          if "cisco-fw" in [tags] {
        elasticsearch {
              hosts => "127.0.0.1:9200"
              index => "firewall-%{+YYYY.MM.dd}"
  }}}
```

Aggregate & Normalize - GROK language

Grok is a language to parse unstructured data using pattern matching

A great tool for development is the Grok debugger interactive web page

https://grokdebug.herokuapp.com



```
Oct 25 23:53:32 aludra.mng.fastnet.intra 15084515: Oct 25 21:53:32.407: %CP-6-TCP: DROP TCP/UDP Portfilter  181.214.87.3(41803) ->
195.96.200.23(33889)
```

```
%{SYSLOGTIMESTAMP:@timestamp} %{SYSLOGHOST:syslog_host} %{DATA:syslog_sequence}: %{SYSLOGTIMESTAMP:syslog_timestamp}
?: %CP-6-..P: DROP TCP/UDP Portfilter  %{IPV4:syslog_srcip}\(%{POSINT:syslog_srcport}\) -> %{IPV4:syslog_dstip}\(%
{POSINT:syslog_dstport}\)
```

☐ Add custom patterns ☐ Keep Empty Captures ☐ Named Captures Only ☐ Singles ☐ Autocomplete Go

```
{
  "SYSLOGTIMESTAMP": [
    [
      "Oct 25 23:53:32"
    ]
  ],
  "MONTH": [
    [
      "Oct",
      "Oct"
    ]
  ],
  "MONTHDAY": [
    [
      "25",
      "25"
    ]
  ],
  "TIME": [
    [
      "23:53:32",
      "21:53:32.407"
    ]
  ],
  "HOUR": [
    [
```

Aggregate & Normalize - Better syslog reliability

Syslog uses mostly UDP unreliable protocol
With logstash is possible to save unique logs from multiple copies, using hashing techniques

Example logstash configuration:

```
filter {
    fingerprint {
      source => ["message"]
      target => "fingerprint"
      key => "fastnethash"
      method => "SHA256"
      concatenate_sources => true
    }}
output {
  elasticsearch {
    document_id => "%{fingerprint}"
  }}
```

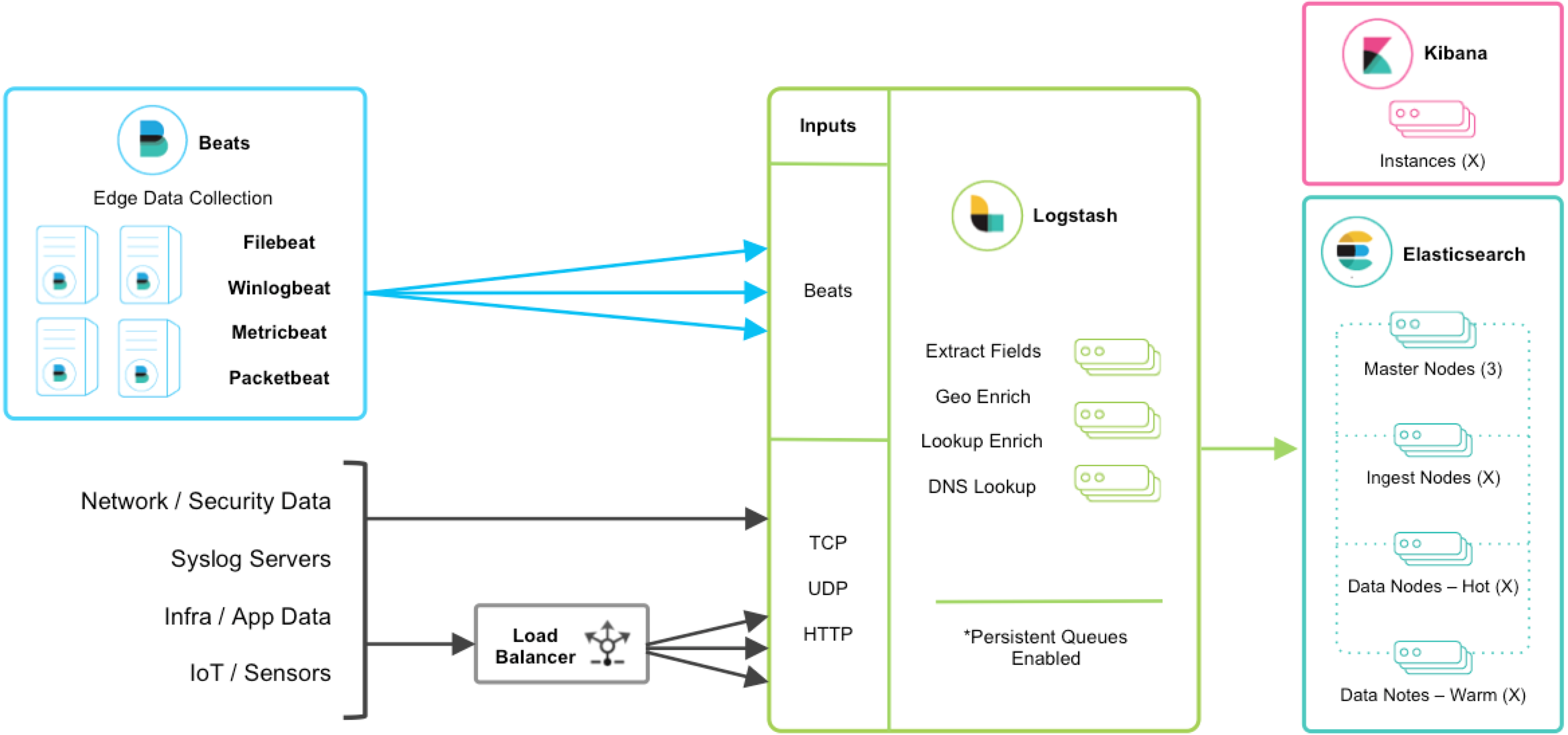
Store & Optimize

Store & Optimize - ELASTICEASEARCH

- Elasticsearch is a search and analysis distributed engine
- Open source project based on Apache Lucene project
- Engine stores and indexes data

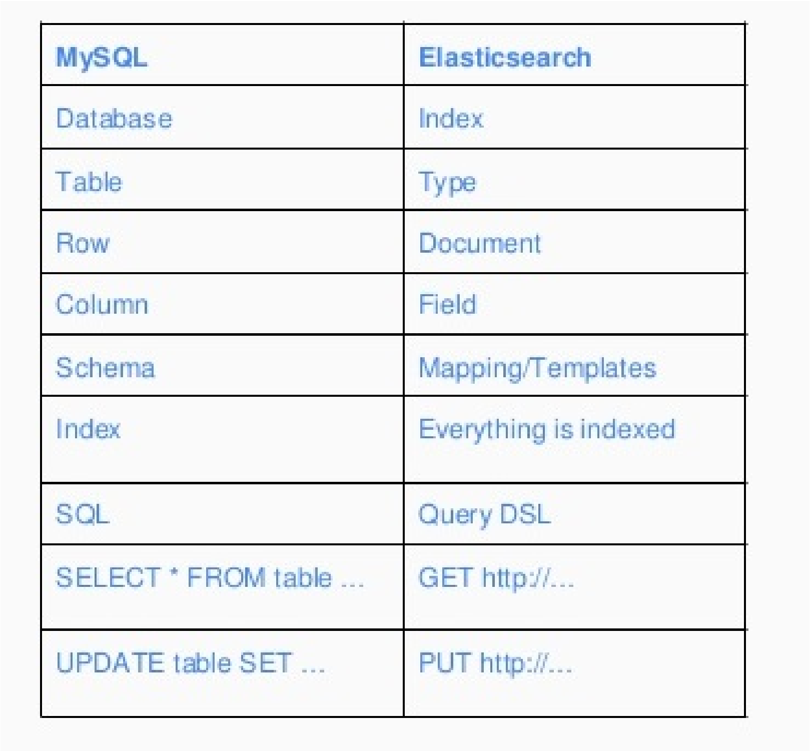
The Heart of the Elastic Stack

Store & Optimize - ELASTICEASEARCH Deployment



Store & Optimize - ELASTICEASERCH MySQL differences

Elasticsearch stores and indexes data like a database with some differences:



MySQL	Elasticsearch
Database	Index
Table	Type
Row	Document
Column	Field
Schema	Mapping/Templates
Index	Everything is indexed
SQL	Query DSL
SELECT * FROM table ...	GET http://...
UPDATE table SET ...	PUT http://...

Store & Optimize - ELASTICSEARCH Security

- Encryption and authentication is implemented in commercial module
- A **workaround** used for deployments used by small group of admins:
    - Isolated Vlan for elastic cluster communication
    - Firewall publishes only the ports used to ingest data, filter on source
    - Isolated Kibana with NGIX server as authenticated proxy

Store & Optimize - Index Maintenance

Elasticsearch module CURATOR performs maintenance on stored data

Used CURATOR to automate remove or archive old data using CRON jobs

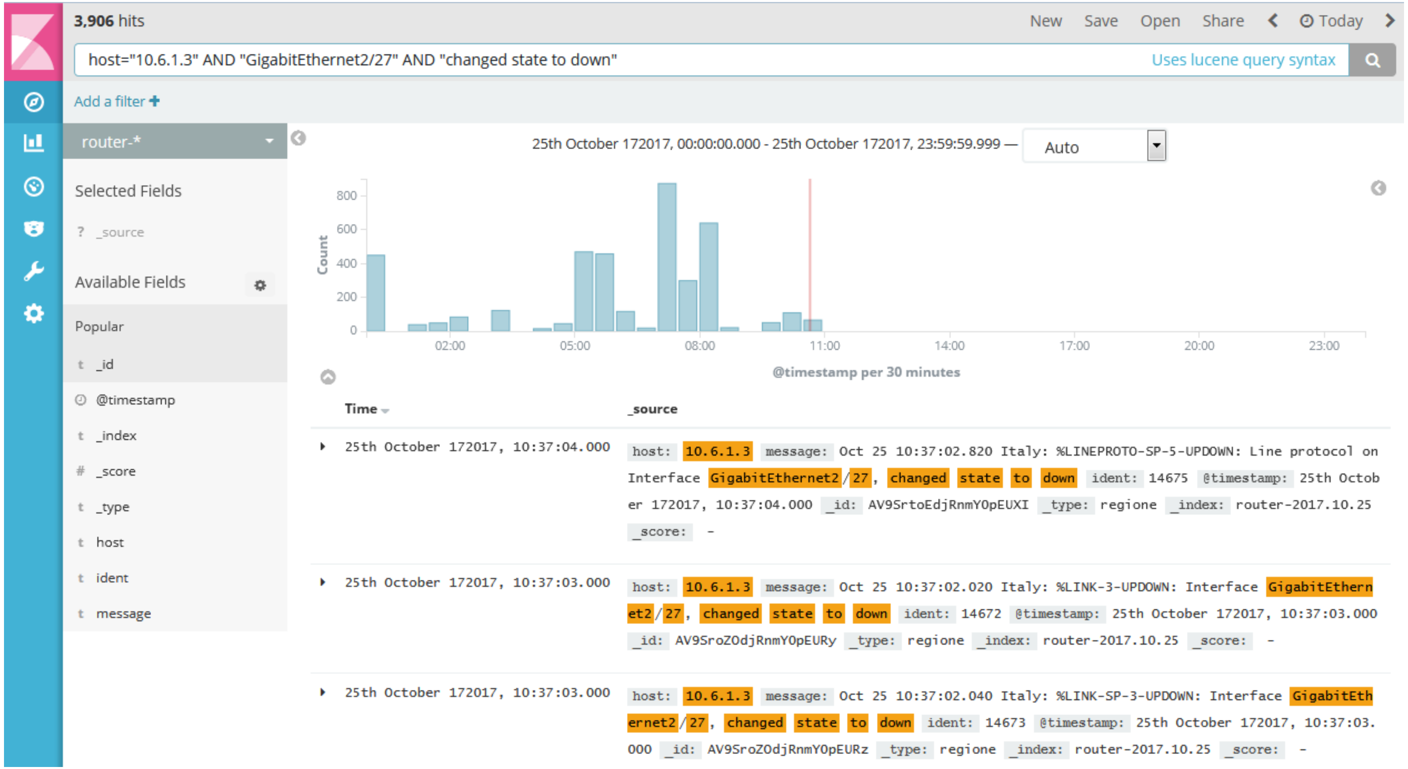
Example configuration:

```
actions:
  1:
    action: delete_indices
    description: >-
      Delete indices older than 7 days
    filters:
    - filtertype: pattern
      kind: prefix
      value: firewall-syslogs-
      exclude:
    - filtertype: age
      source: name
      direction: older
      timestring: '%Y.%m.%d'
      unit: days
      unit_count: 7
```

Analysis & Alert

Analysis &Alert - Search data with Kibana

Full text search with trends



host="10.6.1.3" AND "GigabitEthernet2/27" AND "changed state to down"

25th October 172017, 00:00:00.000 - 25th October 172017, 23:59:59.999 —

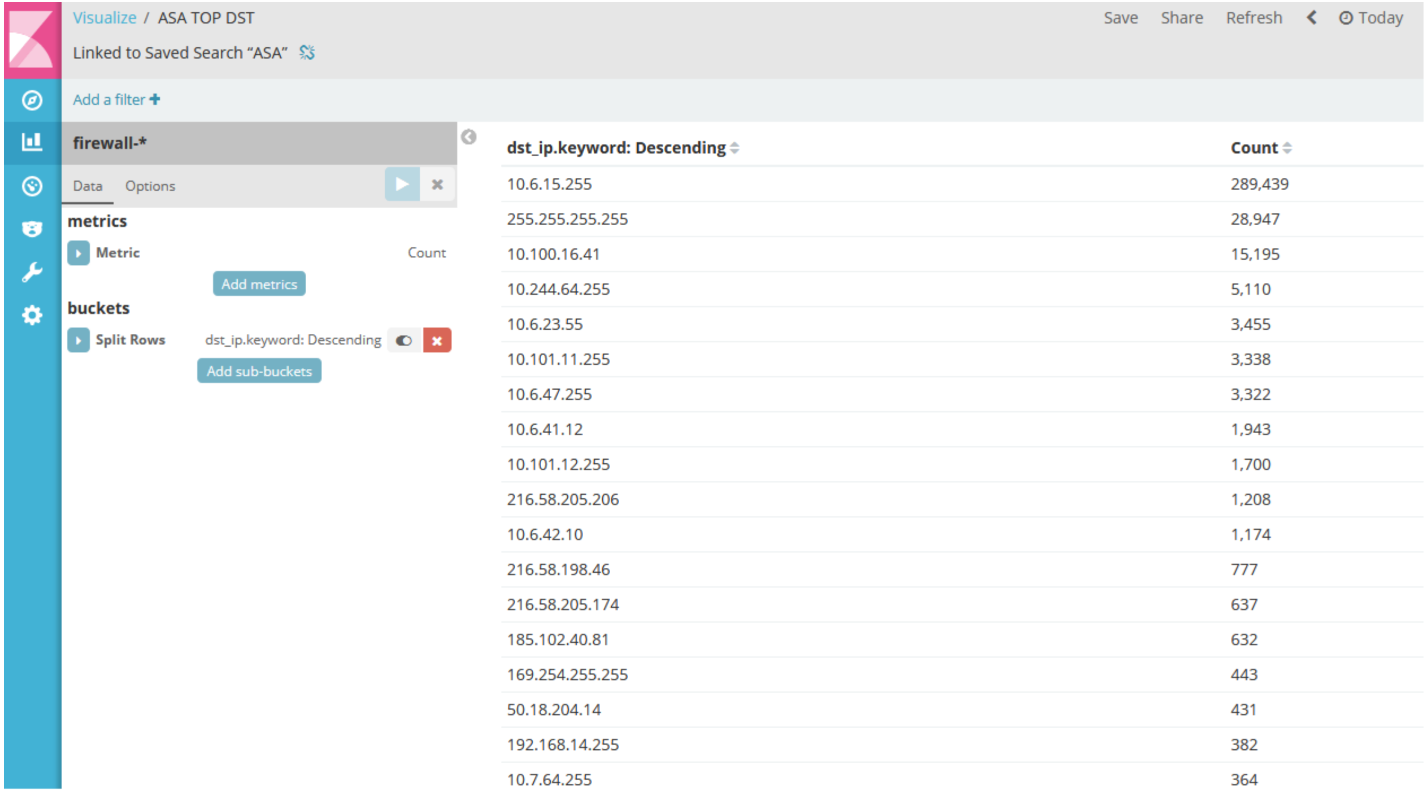
```
25th October 172017, 10:37:04.000   host: 10.6.1.3 message: Oct 25 10:37:02.820 Italy: %LINEPROTO-SP-5-UPDOWN: Line protocol on
                                    Interface GigabitEthernet2/27, changed state to down ident: 14675 @timestamp: 25th Octob
                                    er 172017, 10:37:04.000 _id: AV9SrtoEdjRnmY0pEUXI _type: regione _index: router-2017.10.25
                                    _score: -

25th October 172017, 10:37:03.000   host: 10.6.1.3 message: Oct 25 10:37:02.020 Italy: %LINK-3-UPDOWN: Interface GigabitEthern
                                    et2/27, changed state to down ident: 14672 @timestamp: 25th October 172017, 10:37:03.000
                                    _id: AV9SroZOdjRnmY0pEURy _type: regione _index: router-2017.10.25 _score: -

25th October 172017, 10:37:03.000   host: 10.6.1.3 message: Oct 25 10:37:02.040 Italy: %LINK-SP-3-UPDOWN: Interface GigabitEth
                                    ernet2/27, changed state to down ident: 14673 @timestamp: 25th October 172017, 10:37:03.
                                    000 _id: AV9SroZOdjRnmY0pEURz _type: regione _index: router-2017.10.25 _score: -
```

Analysis &Alert

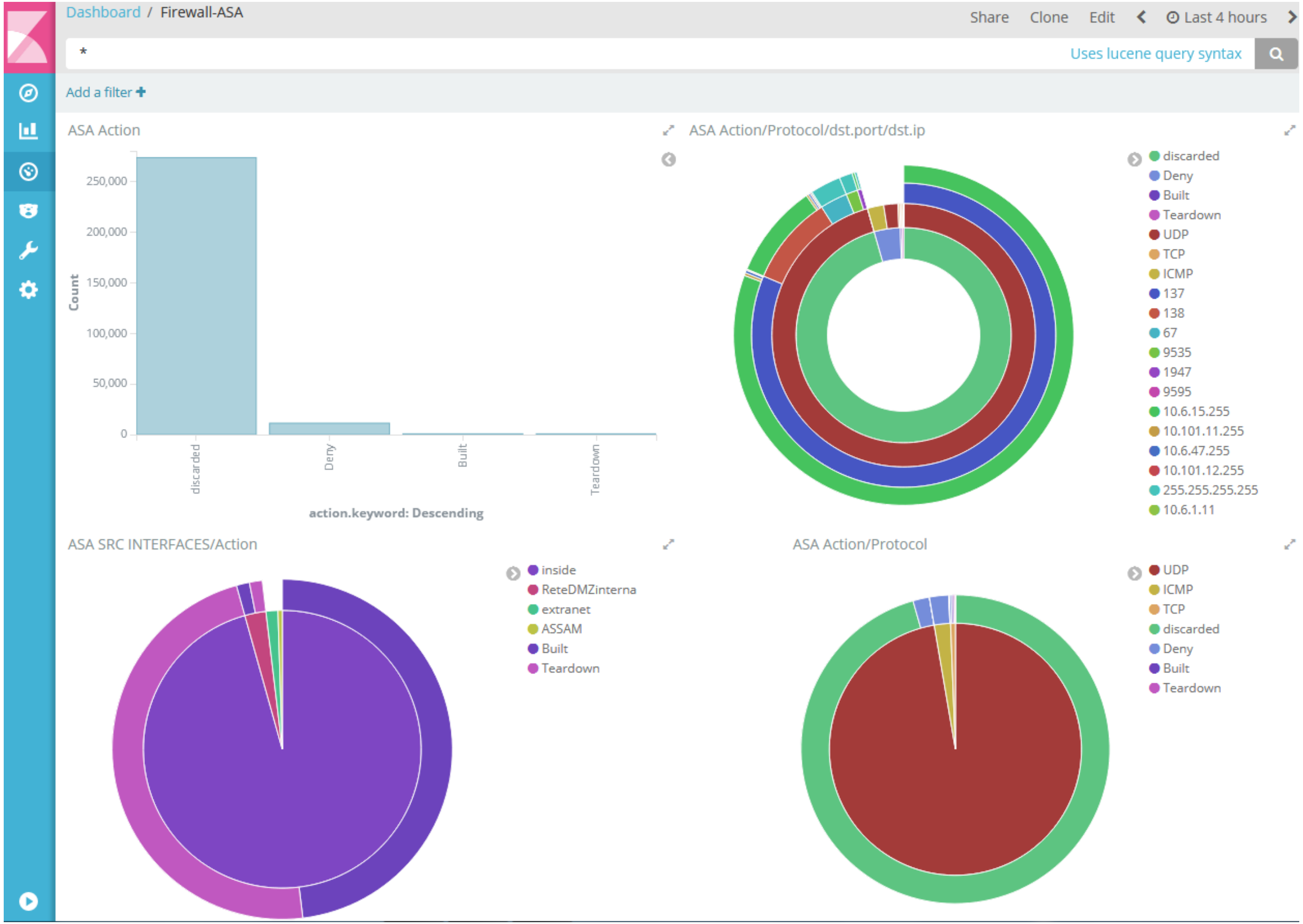
Counting based on occurrences of field gathered from Logstash



dst_ip.keyword: Descending	Count
10.6.15.255	289,439
255.255.255.255	28,947
10.100.16.41	15,195
10.244.64.255	5,110
10.6.23.55	3,455
10.101.11.255	3,338
10.6.47.255	3,322
10.6.41.12	1,943
10.101.12.255	1,700
216.58.205.206	1,208
10.6.42.10	1,174
216.58.198.46	777
216.58.205.174	637
185.102.40.81	632
169.254.255.255	443
50.18.204.14	431
192.168.14.255	382
10.7.64.255	364

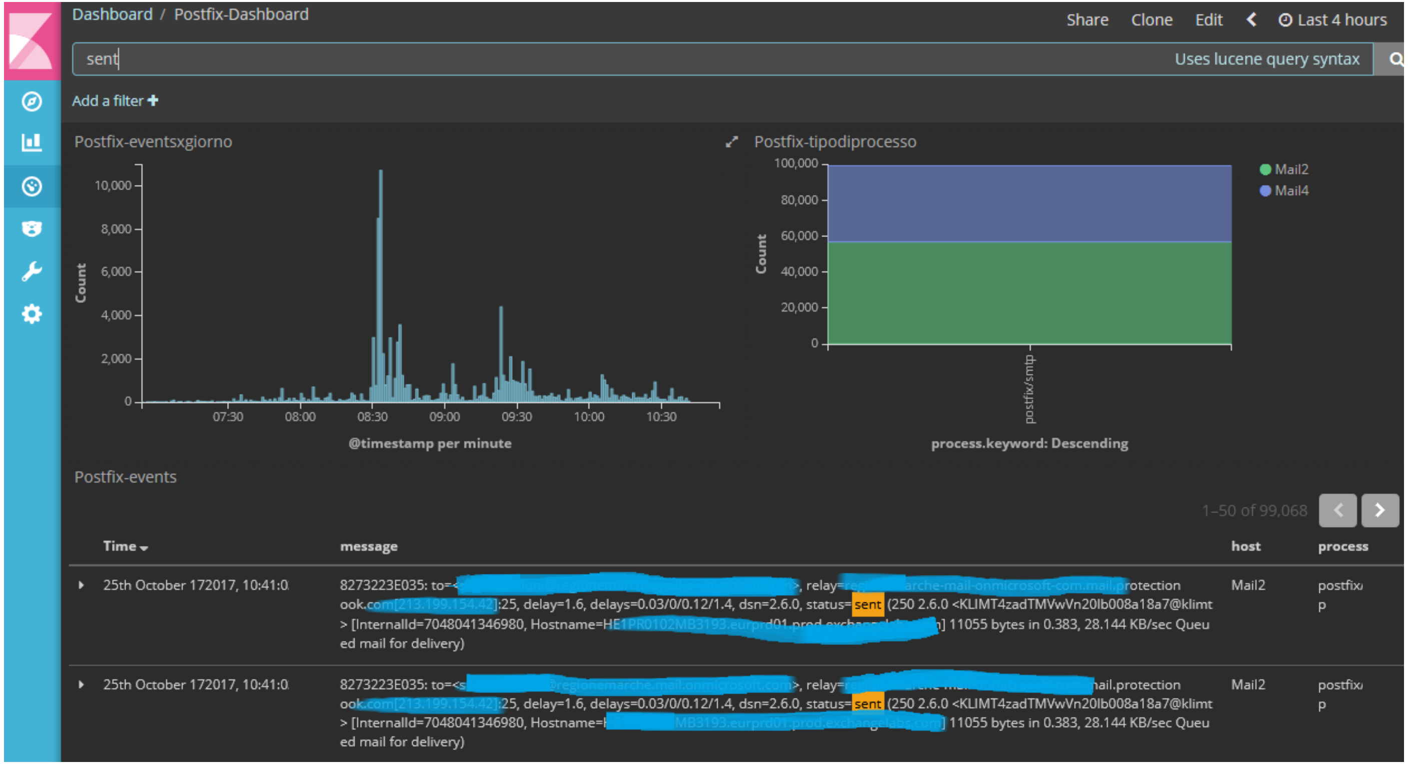
Analysis &Alert

Graphing numbers of occurrences at several levels



Analysis &Alert

Email server example: Trending emails sent over time



Time	message	host	process
25th October 172017, 10:41:0	8273223E035: to=[redacted], relay=[redacted]protection ook.[redacted]:25, delay=1.6, delays=0.03/0/0.12/1.4, dsn=2.6.0, status=sent (250 2.6.0 <KLIMT4zadTMVwVn20lb008a18a7@klimt > [InternalId=7048041346980, Hostname=[redacted]] 11055 bytes in 0.383, 28.144 KB/sec Queued mail for delivery)	Mail2	postfix/p
25th October 172017, 10:41:0	8273223E035: to=<s[redacted]>, relay=[redacted]ail.protection ook[redacted]:25, delay=1.6, delays=0.03/0/0.12/1.4, dsn=2.6.0, status=sent (250 2.6.0 <KLIMT4zadTMVwVn20lb008a18a7@klimt > [InternalId=7048041346980, Hostname=[redacted]] 11055 bytes in 0.383, 28.144 KB/sec Queued mail for delivery)	Mail2	postfix/p

Analysis &Alert - Alerting

Alerting is included in the commercial X-Pack, Watcher module
Alternative open source project using the elastic API:
https://github.com/Yelp/elastalert
http://elastalert.readthedocs.io

What’s next? Some interesting new features

- Interesting developments are being released constantly, some recent:
- Netflow module, for easier traffic analysis
- Logstash Jquery for importing SQL data
- Artificial intelligence features, unfortunately as commercial add-on

Thank you !

For any additional questions, please send me a note: o.melendres@fastnet.it