elastic ON TOUR

From Ingestion to Analysis

How to make the Elastic Stack work for you

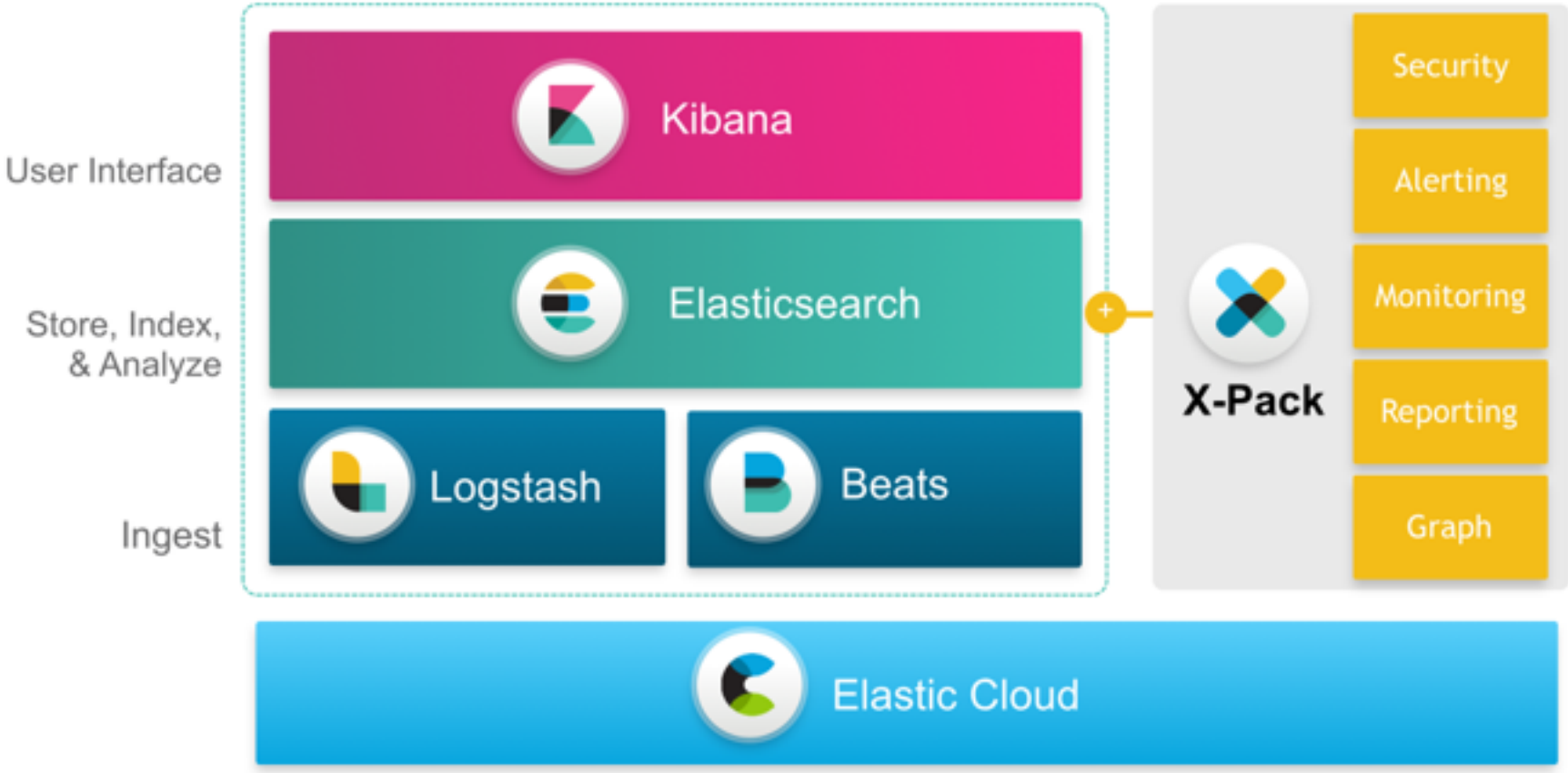
Christian Zumbiehl, Professional Services Consultant

David Pilato, Developer | Evangelist

November 9th 2016

Agenda:

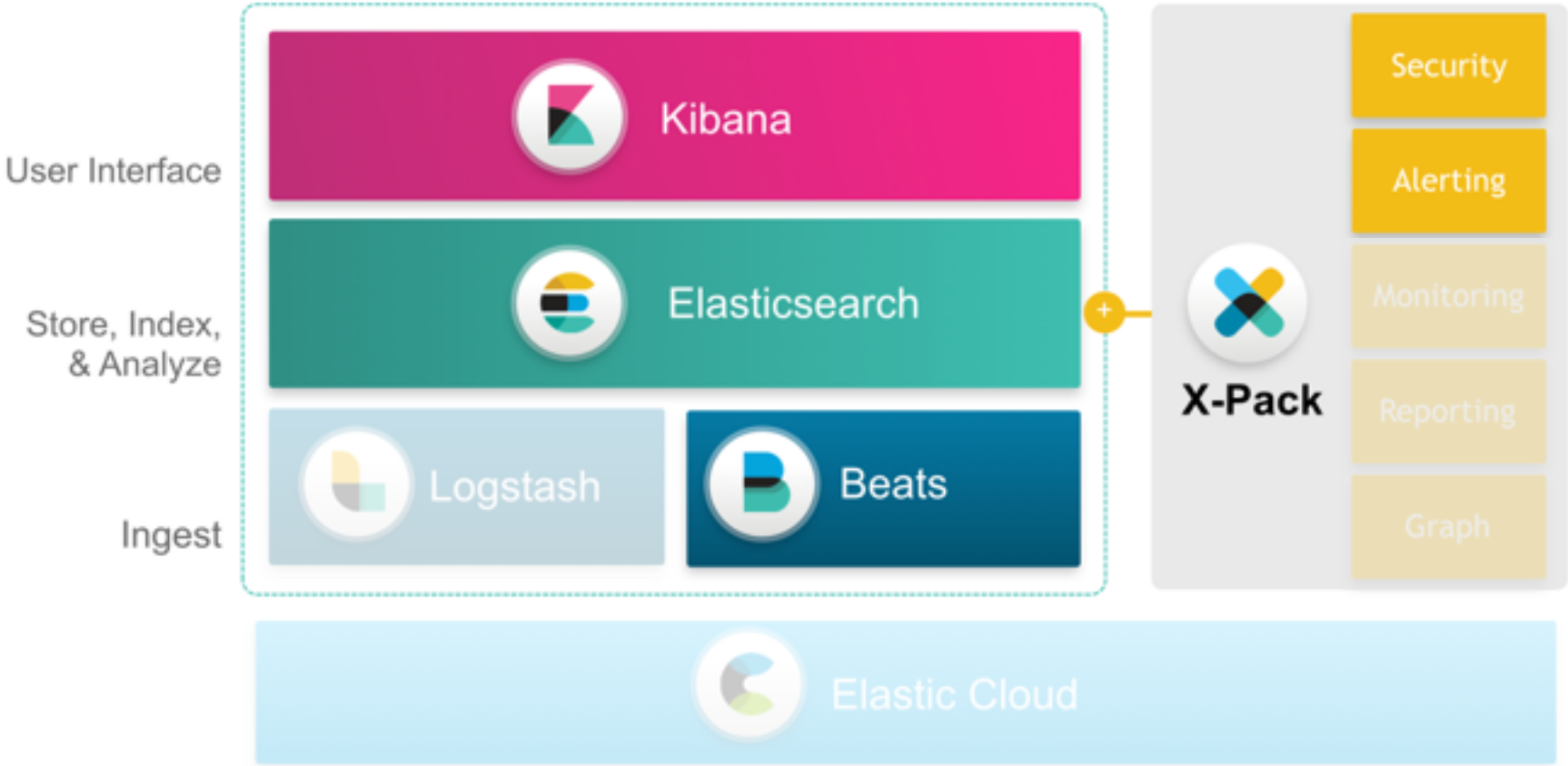
- Detecting intrusion through failed logins
- Q&A
- Alerting on known threats
- Q&A


User Interface
Store, Index, & Analyze
Ingest
Kibana
Elasticsearch
Logstash
Beats
X-Pack
Security
Alerting
Monitoring
Reporting
Graph
Elastic Cloud

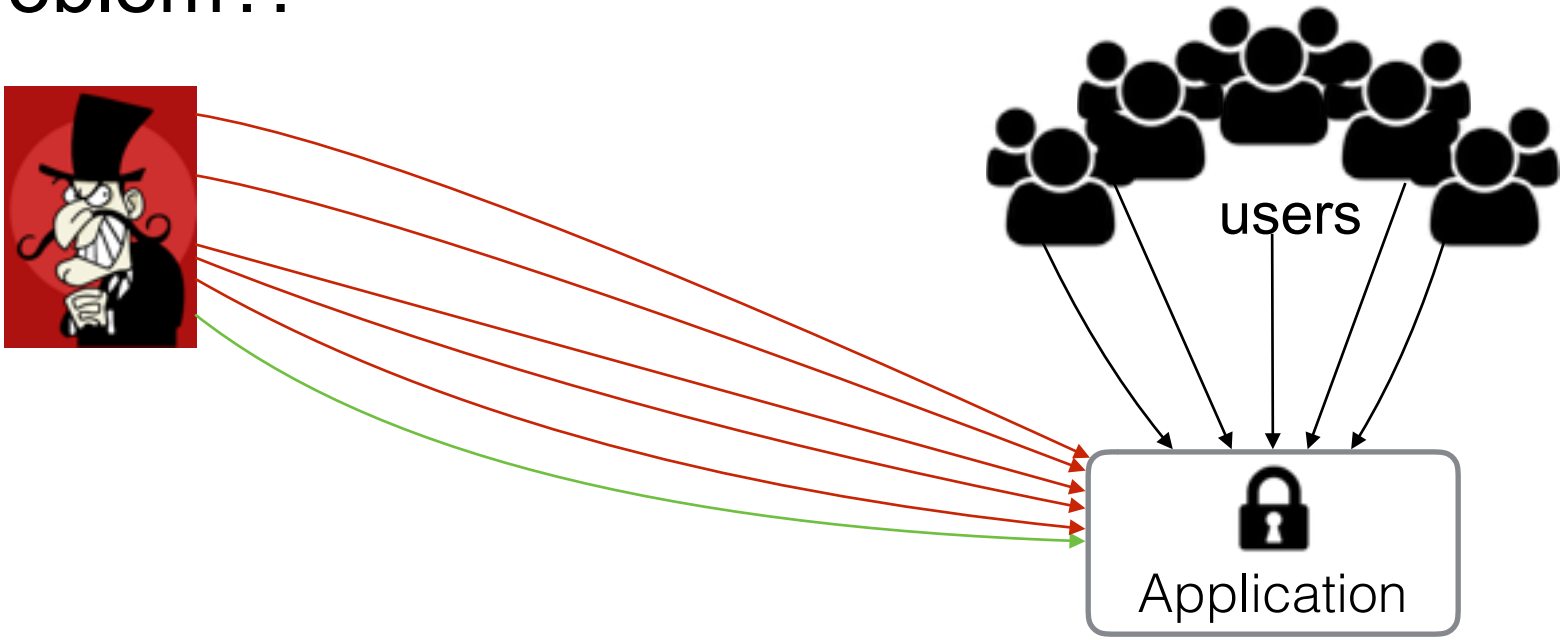
Detecting intrusion through failed logins

Detecting intrusion through failed logins

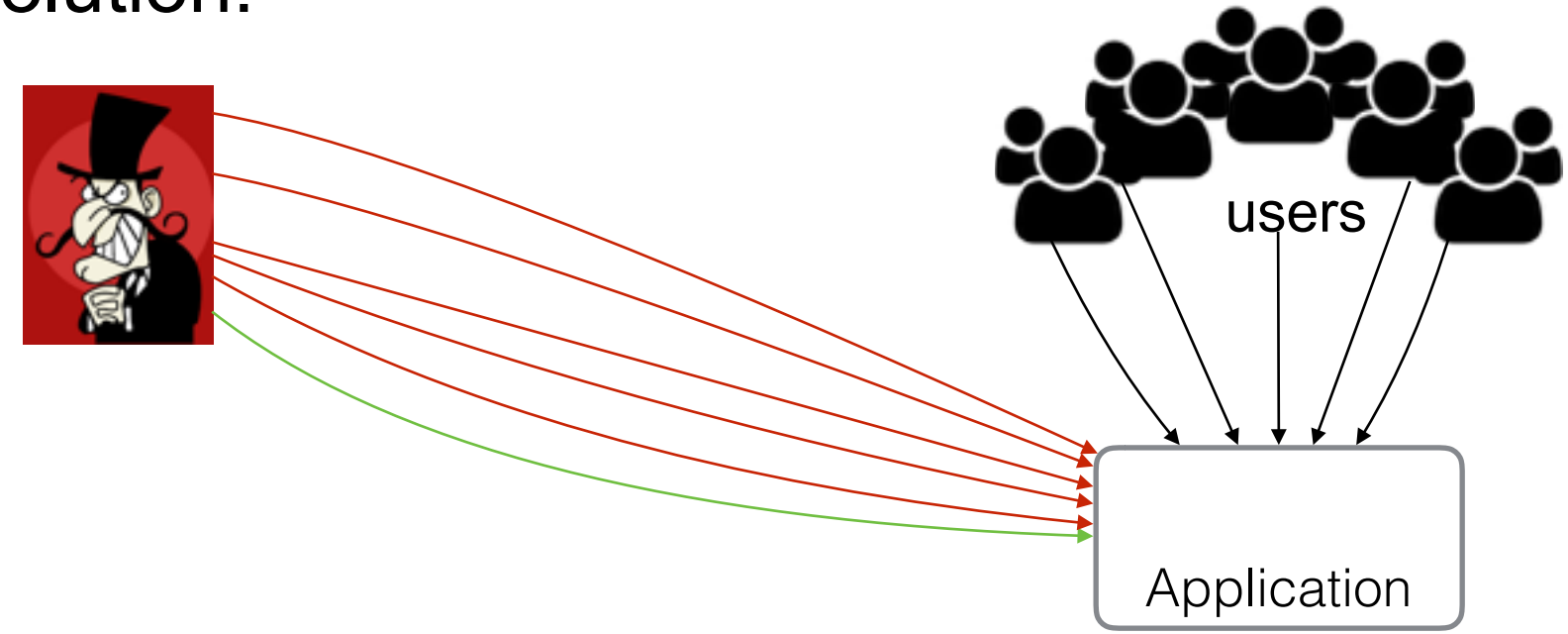
- Detecting a login pattern which is suspicious:
    - Website, SSH, Elasticsearch, ...
- Alert if a user fails to login more than X times in a row, followed by a successful login.
- Products used: Filebeat, Elasticsearch, Kibana
- Features used: Ingest Node, Alerting, Pipeline Aggregations, Custom Visualization


User Interface
Store, Index, & Analyze
Ingest
Kibana
Elasticsearch
Logstash
Beats
X-Pack
Security
Alerting
Monitoring
Reporting
Graph
Elastic Cloud

Problem?!

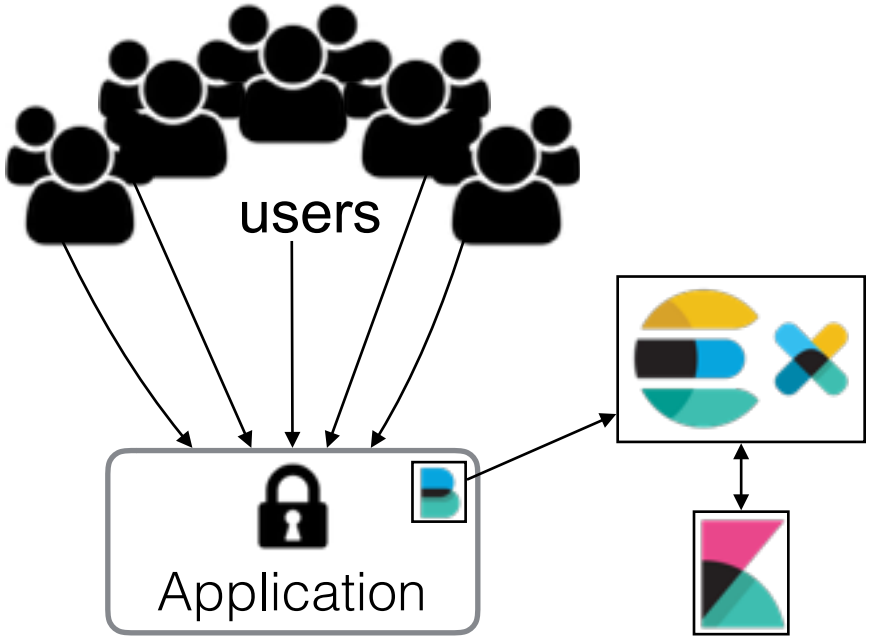


Solution!



Alerting right away so we can "restore" security!

Solution architecture



- Auth Log Shipper (**Filebeat**)
- Data Enrichment (**Ingest Node**)
- Analytics Engine (**Aggregations**)
- Alerting Plugin (**Alerting**)
- Visualization Tool (**Kibana**)

Alerting

- **Schedule** How often?
- **Query** Which documents or values?
- **Condition** Do they meet the criteria?
- **Action** If yes, what should we do?

Alerting on intrusions

- **Schedule** Every 5 seconds.
- **Query** The number of logins per *user_host* in the last 5 minutes.
- **Condition** Does the *user_host* have more than 3 failures followed by a success?
- **Action** If yes, create a new document confirming the attack to *user_host*.

Demo:

- SSH login attacks
- 1 Linux machine with Filebeat collecting auth logs
- 1 Elasticsearch host receiving auth logs and executing Alerting
- 1 Kibana host to serve visualizations and dashboards
- Script that logs in with different users (95% success and 5% failure)
- Script that simulates a brute force until success (random number of retries)

Indices

- *auth-yyyy-MM-dd*, all login attempts
- *auth-detection*, successful logins after X failures (probably an attack)

```
green  open auth-detection                1 0     18      0  45.2kb  45.2kb
green  open auth-2016.09.25               1 0    637      0   227kb   227kb
green  open auth-2016.09.09               1 0    278      0 101.4kb 101.4kb
green  open auth-2016.09.21               1 0    147      0 100.4kb 100.4kb
green  open auth-2016.09.22               1 0     52      0    73kb    73kb
green  open auth-2016.09.08               1 0    278      0  85.7kb  85.7kb
green  open auth-2016.09.24               1 0    909      0   300kb   300kb
green  open auth-2016.09.26               1 0    250      0 135.7kb 135.7kb
green  open auth-2016.09.28               1 0     14      0  43.2kb  43.2kb
green  open auth-2016.09.27               1 0   1290      0 358.8kb 358.8kb
```

Login dashboard brainstorm

Demo Time!

Data model tricks

- Concatenate fields *"user"* and *"host"* into a new field *"user_host"*
  - Uses more storage but makes aggregations easier to write and **faster**

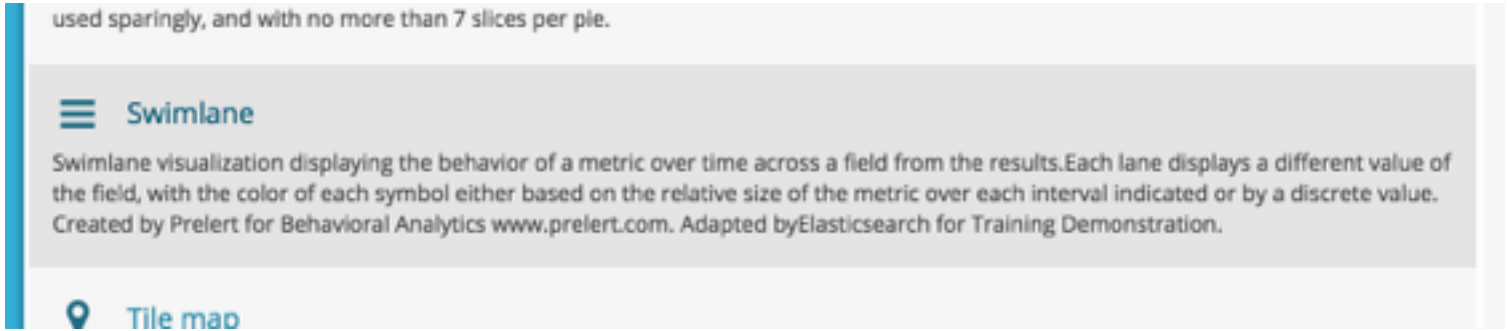
- Add a new field "threat_score" during ingestion (Ingest Node) based on the login
  - *accepted -> 0*, *invalid_user -> 1*, *failed -> 2*
  - *accepted_attack_confirmed -> 4*, added by Alerting

Data model tricks

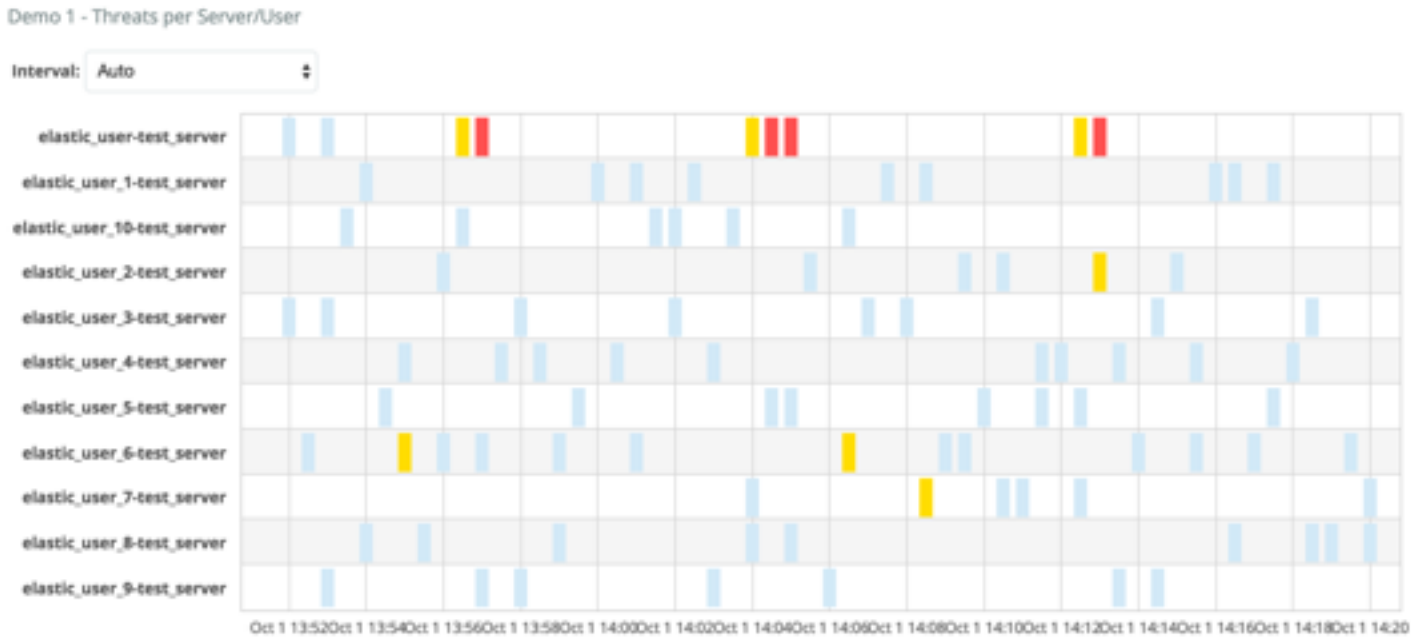
- Attack overlap
  - attack documents can be generated many times for the same attack (Alerting freq)
  - we set the attack document timestamp to the successful login timestamp
  - no duplicates in the visualization
  - easier than setting the same document id

Extra Visualization

- Swimlane (https://github.com/prelert/kibana-swimlane-vis)
- Created by Prelert, available on May 18th 2016
- Installed as a plugin in Kibana and available as a visualization



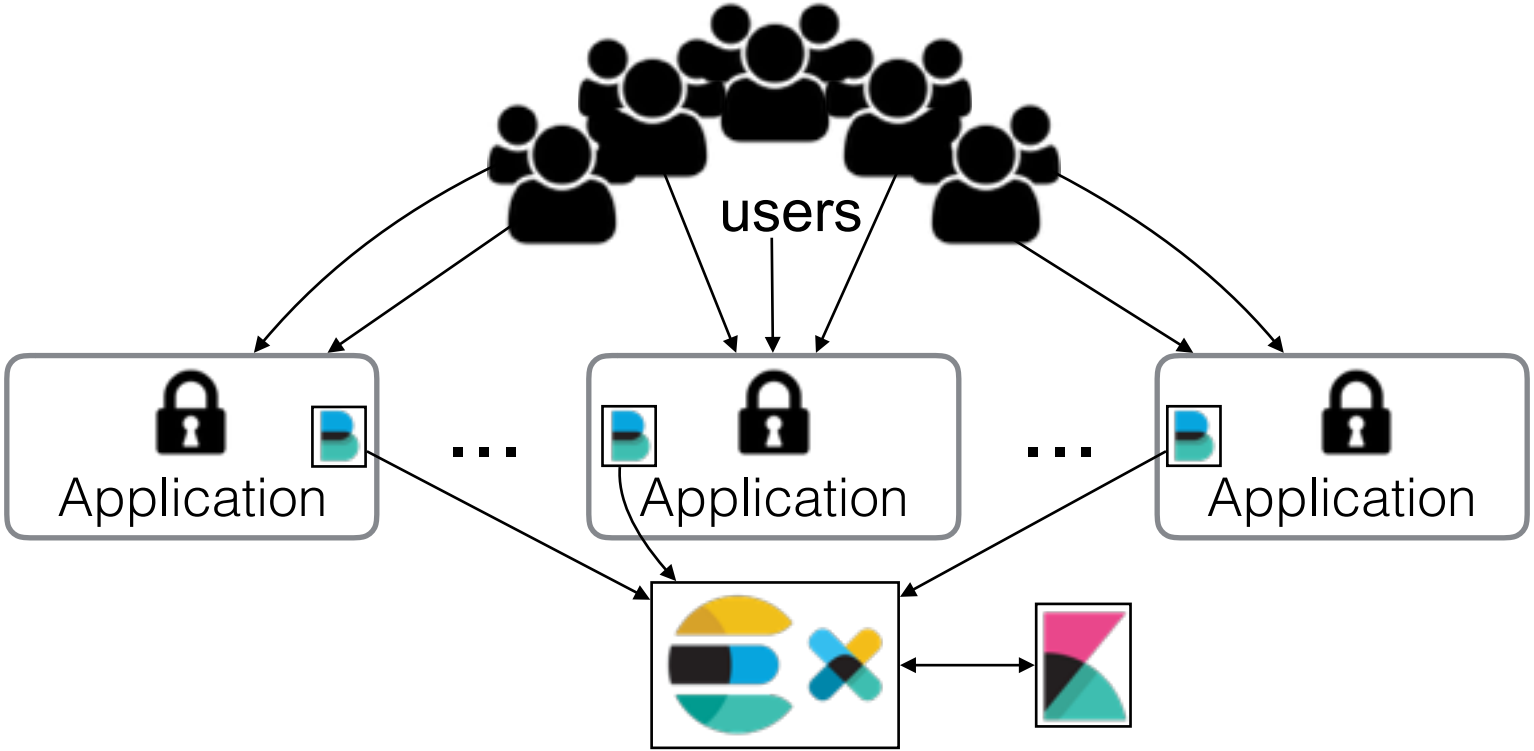
Swimlane


Demo 1 - Threats per Server/User
Interval:
Auto
elastic_user-test_server
elastic_user_1-test_server
elastic_user_10-test_server
elastic_user_2-test_server
elastic_user_3-test_server
elastic_user_4-test_server
elastic_user_5-test_server
elastic_user_6-test_server
elastic_user_7-test_server
elastic_user_8-test_server
elastic_user_9-test_server
Oct 1 13:52
Oct 1 13:54
Oct 1 13:56
Oct 1 13:58
Oct 1 14:00
Oct 1 14:02
Oct 1 14:04
Oct 1 14:06
Oct 1 14:08
Oct 1 14:10
Oct 1 14:12
Oct 1 14:14
Oct 1 14:16
Oct 1 14:18
Oct 1 14:20

What if the application grows?

- From one application server to multiple servers:
  - add one filebeat per application server
  - alert based on attempts to the same server (current)
  - alert based on all attempts, regardless of the server (small aggregation change)

Scaling the application monitoring



Scaling the Elastic Stack

- Ingest node vs. Logstash
- Dedicated ingest node
- Alerting vs. Cronjob
- Alerting frequency: 5 vs. 60 seconds

Ingest Node versus **Logstash**

- Ingest Node
  - simple architecture (beats -> Elasticsearch)
  - sub-set of processors (filters)
- Logstash
  - no cluster resource overhead (CPU and RAM)
  - multiple inputs and outputs

Dedicated Ingest Node?

- No performance overhead on data nodes
- Tightly coupled to Elasticsearch
- Centralized configuration
- Not as flexible and powerful as Logstash (archiving, e.g. S3)

Alerting versus Cronjob

- No separate process/system to setup
- Highly Available
- Built-in actions
- Execution History
- API Driven
- Maintained by Elastic

Alerting frequency: 5 or 60 seconds

- What is your hard requirement?
- How expensive is the Alerting aggregation?
- How many new documents are indexed per execution?
- How expensive is the Alerting *query*?

How expensive is the Alerting query?

```
GET auth-*/auth_log/_search
{
  "query": {
    "bool": {
      "filter": [
        { "range": { "@timestamp": { "gte": "now-5m"}}},
        { "terms": { "access": [ "accepted", "failed" ]}},
        { "exists": { "field": "user_host" }}
      ]
    }
  }
}
```

How expensive is the Alerting query?

Range filter for the last 5 minutes

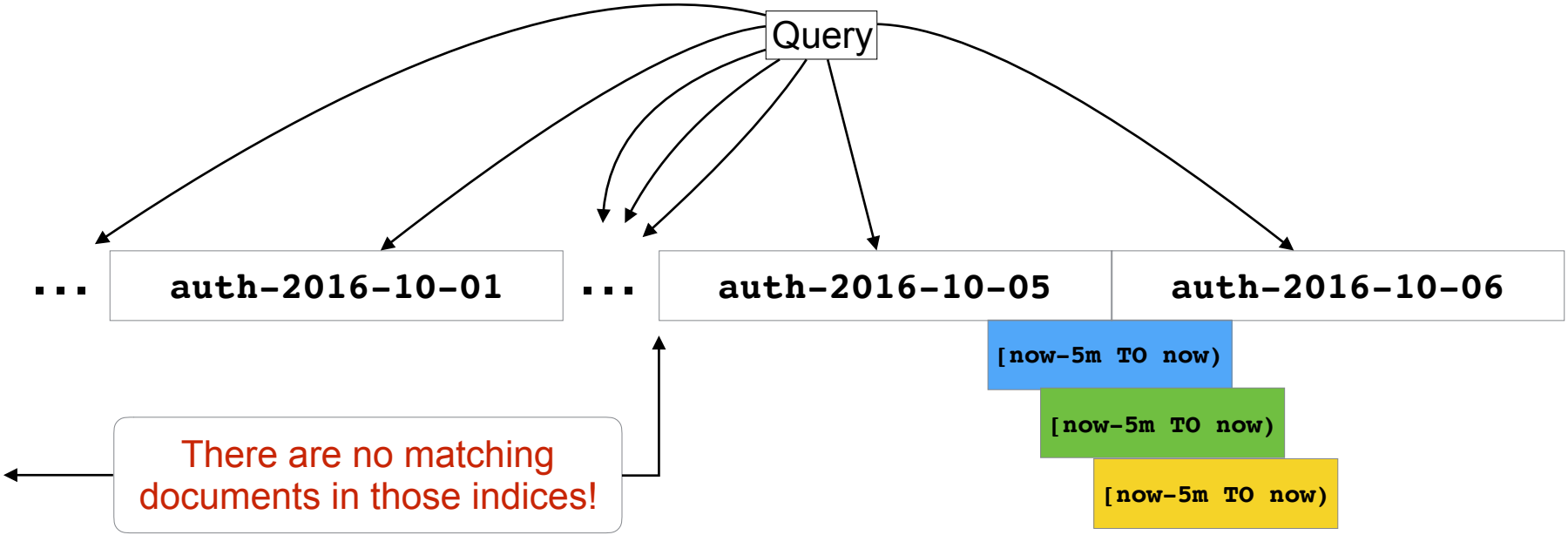
```
GET auth-*/auth_log/_search

{ "range": { "@timestamp": { "gte": "now-5m"}}},
```

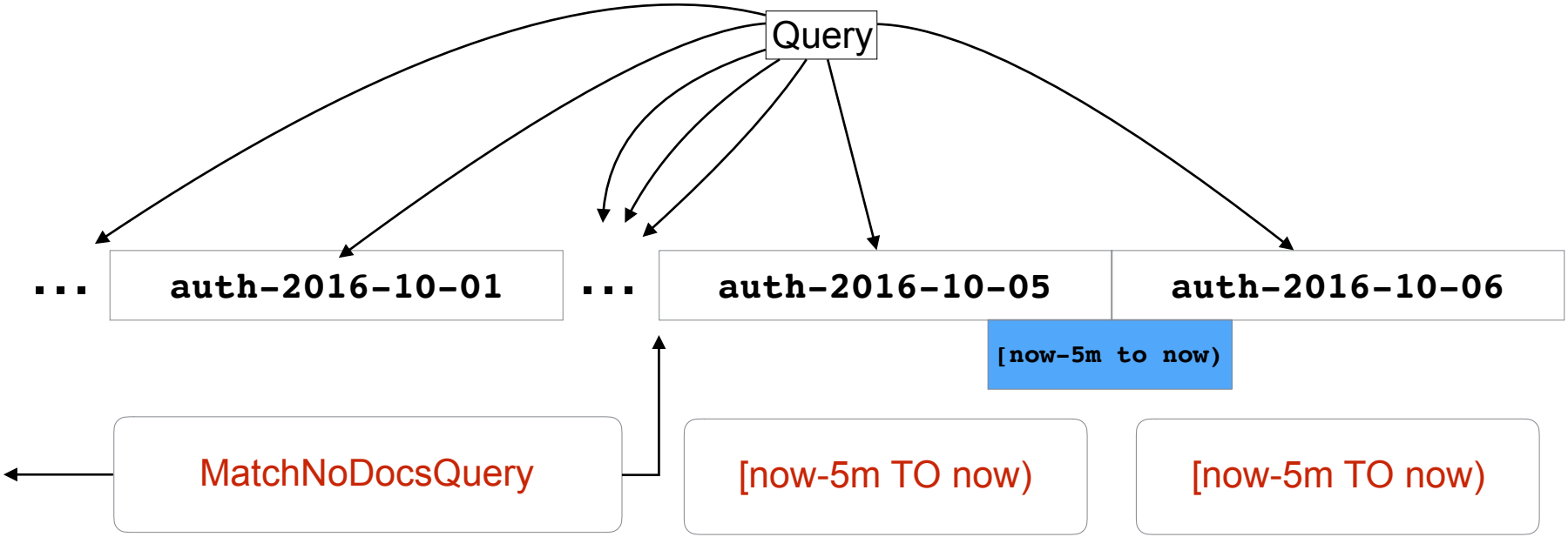
Every shard of every index that starts with *auth-*

But only events in the **last five minutes**

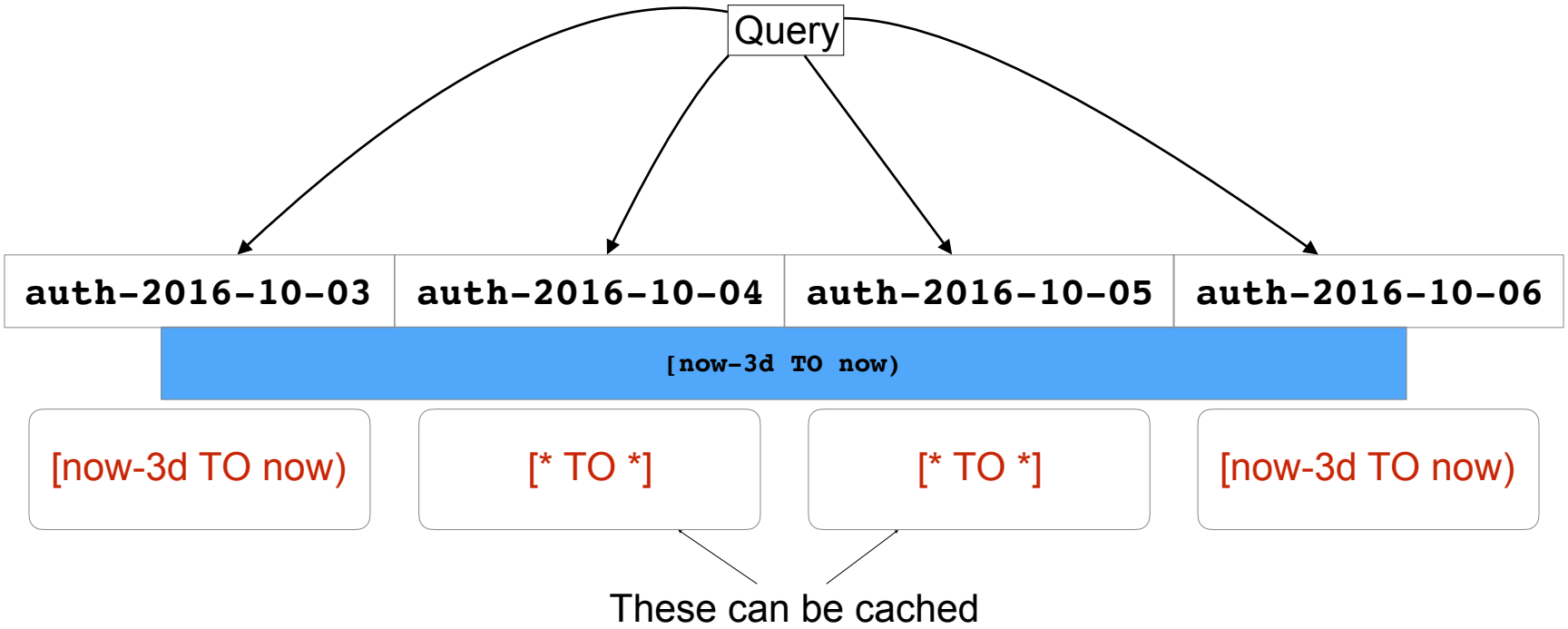
Query executes in all indices



Elasticsearch re-writes it to optimize execution



This is also used in queries from Kibana



What if you have too many indices?

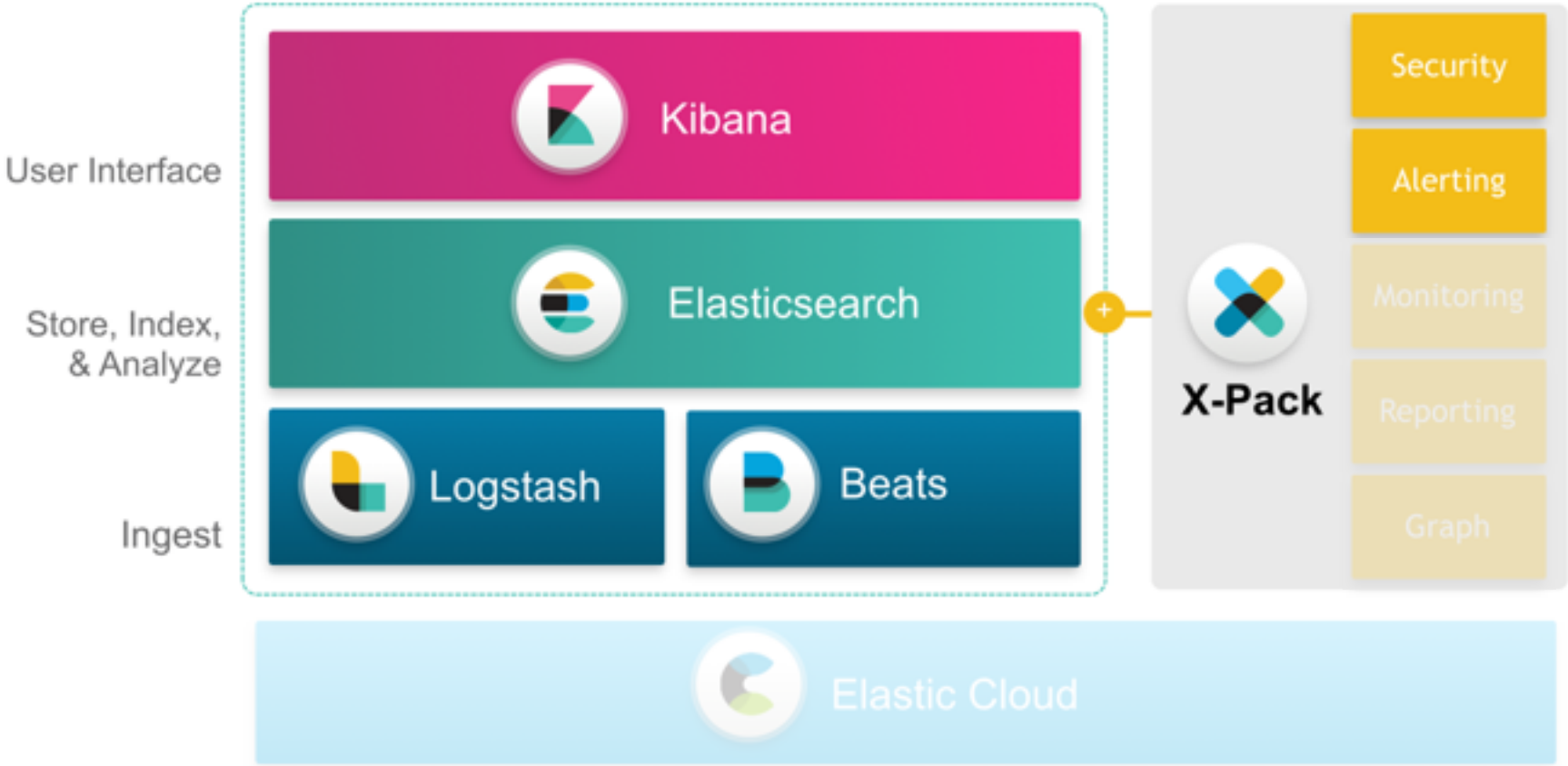
- 1 year of data with 1 index per day
- One should **not** go into 365 indices if only 2 are needed
- Better ways to solve:
  - Aliases
    - create an alias pointing to the last two days
  - Date math support in index names
    - `/<auth-{now/d-1d}>,<auth-{now/d}>/_search`

Questions?

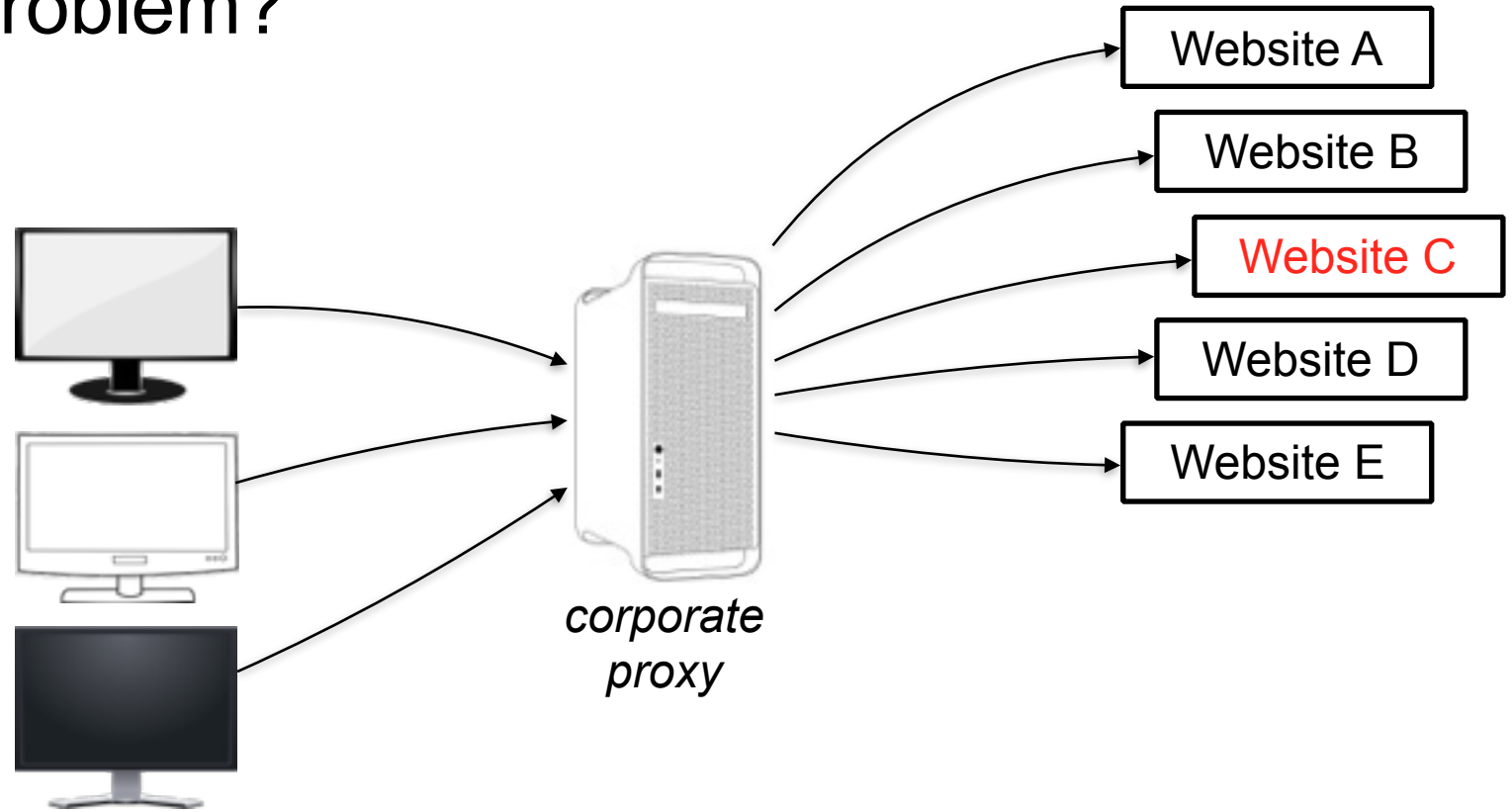
Alerting on known threats

Alerting on known threats

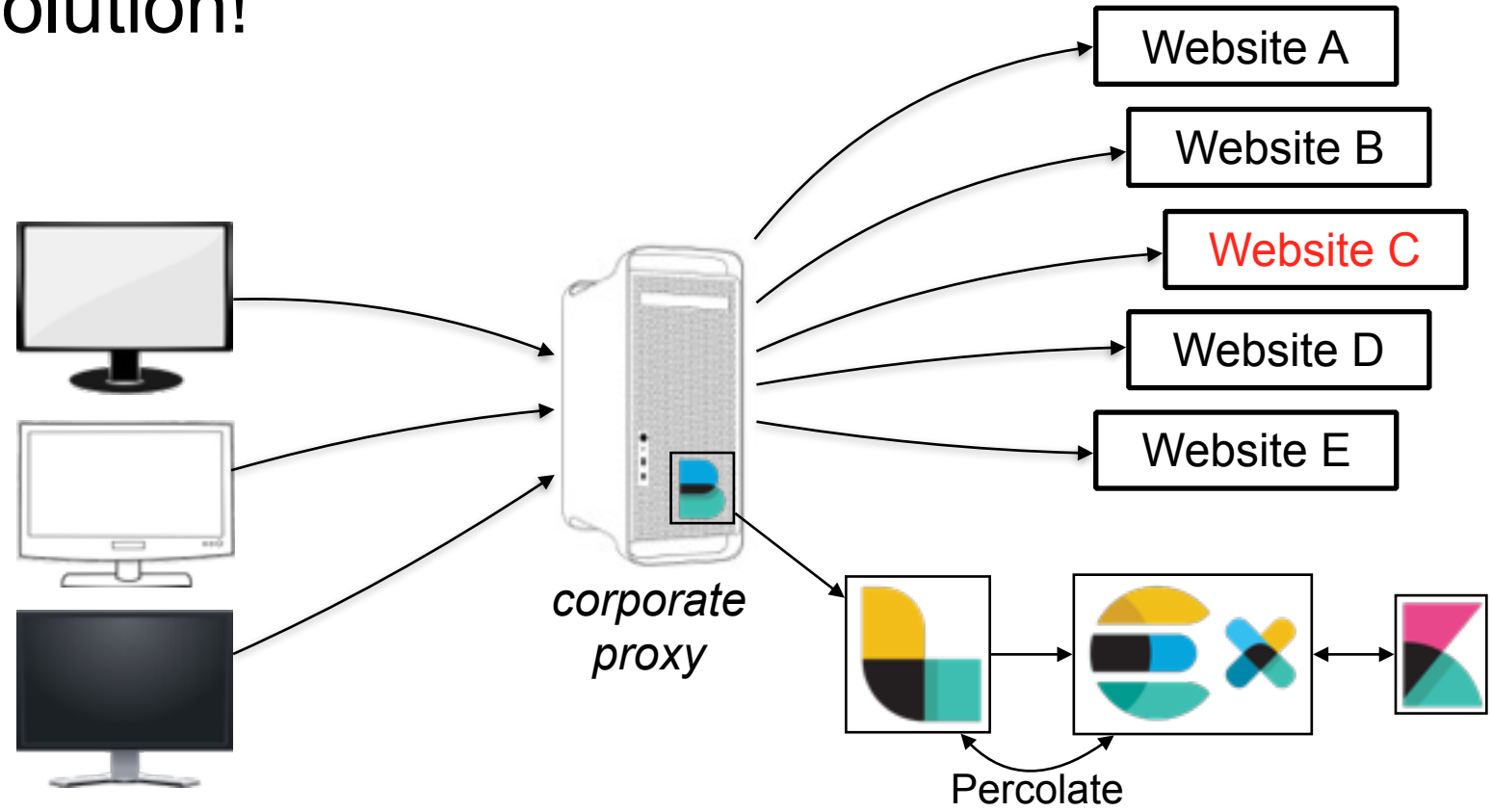
- Detect when a desktop/server connects to a known threat
- Alert on current and retrospective events as threat list is updated
- Products used: Packetbeat, Logstash, Elasticsearch, Kibana
- Features: Alerting, Aggregations, Percolator, Elasticsearch filter for Logstash


User Interface
Store, Index, & Analyze
Ingest
Kibana
Elasticsearch
Logstash
Beats
X-Pack
Security
Alerting
Monitoring
Reporting
Graph
Elastic Cloud

Problem?

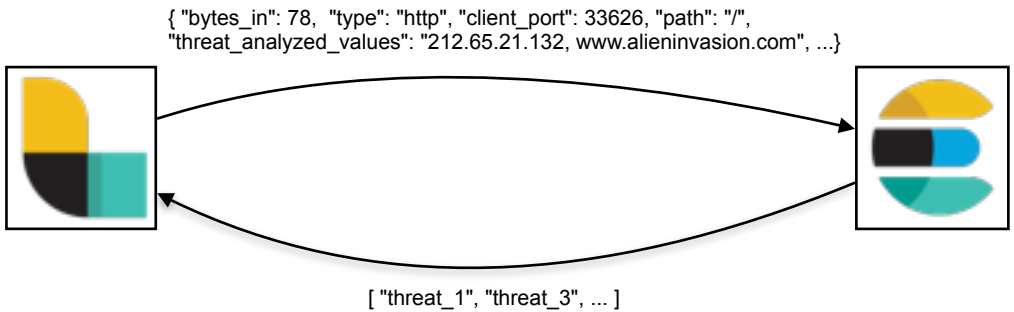

Website A
Website B
Website C
Website D
Website E
corporate
proxy

Solution!


Website A
Website B
Website C
Website D
Website E
corporate
proxy
Percolate

Percolator

- "Search Reversed"
  - stores queries and verifies if a given document matches any stored query
- Used to classify a request as threat or not



Demo:

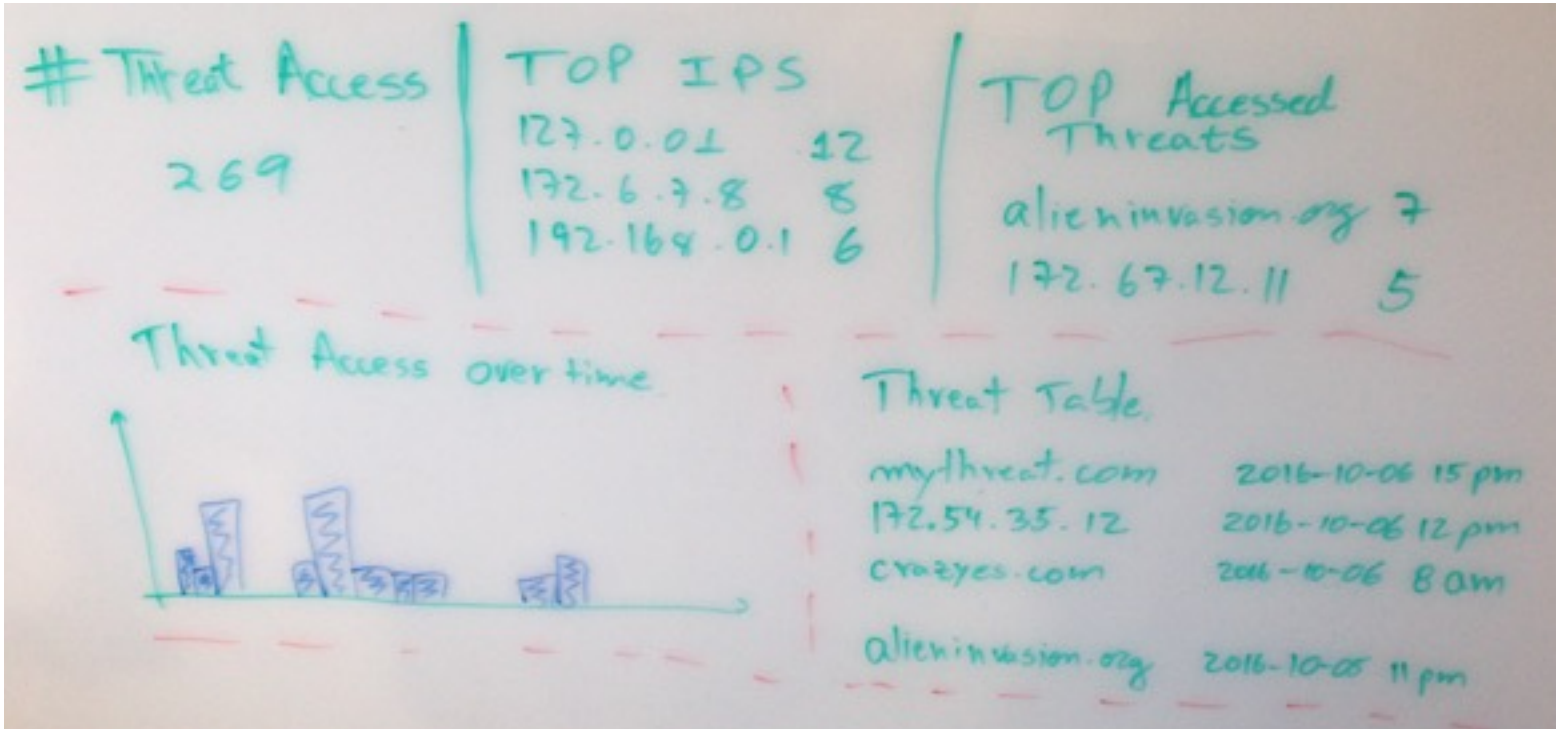
- SSH login attacks
- 1 Linux machine with Packetbeat collecting tcp requests
- 1 Logstash host receiving Packetbeat info, enriching and sending to Elasticsearch
- 1 Elasticsearch host receiving requests and executing Alerting
- 1 Kibana host to serve visualizations and dashboards
- Scripts that access a known and a new threat

Indices

- *threats-yyyy-MM-dd*, all known threats as percolate queries
- *packetbeat-yyyy-MM-dd*, requests to hosts

```
green  open threats-2016.08.09              1 0     15     0    40kb    40kb
green  open threats-2016.07.30              1 0     50     0  49.1kb  49.1kb
green  open threats-2016.07.05              1 0     25     0  39.7kb  39.7kb
green  open threats-2016.08.07              1 0     34     0  46.4kb  46.4kb
green  open threats-2016.07.04              1 0     30     0  43.3kb  43.3kb
green  open packetbeat-2016.09.28           1 0     16     0  69.1kb  69.1kb
green  open packetbeat-2016.09.25           1 0      8     0  28.1kb  28.1kb
green  open packetbeat-2016.09.21           1 0     87     0  73.7kb  73.7kb
green  open packetbeat-2016.10.02           1 0    134     0 149.4kb 149.4kb
```

Threat dashboard brainstorm



Demo Time!

Alerting on new threats

- **Schedule** Every 60 seconds.
- **Query** All new threats in the last 60 seconds.
- **Condition** Is there a threat?
- **Action** If yes, *update_by_query* old matching requests.

Elasticsearch filter for Logstash

- Allows Elasticsearch queries during Logstash filtering
- Enhanced for this training:
    - https://github.com/logstash-plugins/logstash-filter-elasticsearch/pull/42
- Allows the use of percolators to classify documents before indexing

Percolator strategies

- Percolator indices in the main Elasticsearch cluster
  - one single cluster to manage
  - scales on its own
- Percolator indices in a dedicated cluster in Logstash machines
  - local lookups
  - scales with Logstash

IPv6

- Multi-dimensional points
  - uses a 1-dimensional kd-tree data structure
  - index and search speedup, and more space-efficient
- All IP addresses are now represented as a 128-bits IPv6 address
  - IPv4s will be translated to an IPv4-mapped IPv6s at index time and converted back
  - https://www.elastic.co/blog/indexing-ipv6-addresses-in-elasticsearch

Questions?

www.elastic.co