Kibana- security analysis

Tuesday, November 21, 2023 2:32 PM

Packet vs Network Events

Packet Events
(Source/Destination)

10.0.0.1 (S)	→	200.1.1.6 (D)
10.0.0.1 (D)	←	200.1.1.6 (S)
10.0.0.1 (S)	→	200.1.1.6 (D)
10.0.0.1 (D)	←	200.1.1.6 (S)

- Based on individual packets
- Source and destination change based on the **direction** of the **communication**

TCPdump
Wireshark
Suricata

Network Events
(Client/Server)

Client (Originator)		Server (Responder)
10.0.0.1	→	200.1.1.6
10.0.0.1	←	200.1.1.6
10.0.0.1	→	200.1.1.6
10.0.0.1	←	200.1.1.6

- Based on who **initiated** the conversation
- Client and Server **do not change** over the course of the connection

Zeek

Event Fields

- Used to provide context information about log events
- "Where it's from"
  - Which data source created the event?

```
event.module: zeek
```

  - Which logtype created the event?

```
event.dataset: http
```

- "What it is"
  - Organize ECS events into categories

```
event.category: network
```

  - Did the event succeed or fail?

```
event.outcome: success
```

  - What type of information is in this event?

```
event.kind: alert
```

Event.Dataset

conn.log

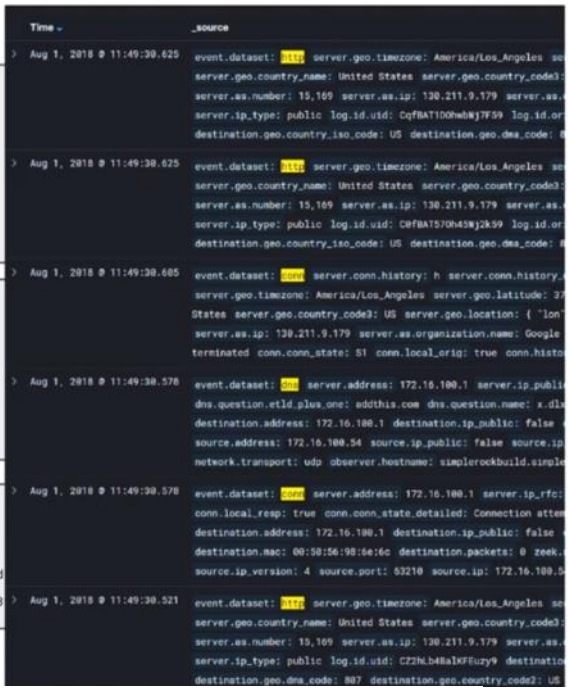
uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	proto	service
CeL4Co1LEBEv58VC91	145.254.160.237	3009	145.253.2.203	53	udp	dns
CjcWi91kXgad8GwL0l	145.254.160.237	3371	216.239.59.99	80	tcp	-
CZg3zn1yvvhuvcV26h	145.254.160.237	3372	65.208.228.223	80	tcp	http

http.log

uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	method	host	uri
CZg3zn1yvvhuvcV26h	145.254.160.237	3372	65.208.228.223	80	GET	www.ethereal.com	/download.html
C3iDqa4TCwUggp8xek	172.16.100.52	3372	173.241.244.22	80	GET	www.reuters.com	/bootstrap.js
CxlYs13eVfsp2Si1vb	172.16.100.52	3372	173.241.244.22	80	GET	www.reuters.com	/index.html

dns.log

uid	id.orig_h	id.orig_p	id.resp_h	id.resp_p	proto	query	answers
CGYQ9z1nbIhdx1NGv6	145.254.160.237	3372	65.208.228.223	53	tcp	www.hockeypads.com	staticad
CGYQ9z1nbIhdx1NGv6	145.254.160.237	3372	65.208.228.223	53	tcp	www.reddit.com	174.21.3



Field Matching [Examples]

- Entries from the Zeek conn.log

```
event.dataset: conn
```

- Log Entries where the server/responding port is 5353

```
server.port: 5353
```

- Limit results to entries associated with the Zeek UID CqfBAT1DOhwbWj7F59

```
zeek.session_id: CqfBAT1DOhwbWj7F59
```

Search Basics: IP Address

Lucene

```
client.ip: 172.16.100.0\/24
```

- Fields must be mapped as IP
- Can search with CIDR notation!
- Exact IP Match

```
client.ip: 172.16.100.54
```

- Match on CIDR range

```
client.ip: 172.16.100.0/24
```

- Internal to External Traffic

```
client.ip: 172.16.100.0/24 AND NOT server.ip: 172.16.100.0/24
```

Boolean Searches [Examples]

- Entries from the Zeek conn.log or http.log

```
event.dataset: (conn OR http)
```

```
event.dataset:conn OR event.dataset:http
```

- Log Entries where the server port is 5353 but the transport protocol is not TCP

```
server.port: 5353 AND NOT network.transport: tcp
```

- Limit results to entries from the Zeek http.log that have http methods other than GET or POST

```
event.dataset: http AND NOT http.request.method: (POST OR GET)
```

Field Exists

Lucene:

```
_exists_: <FIELD>
```

- Search for the field, not the value
- Should a field be present? Anomaly!

```
<FIELD>: *
```

- Results where and the HTTP referrer field exists

```
http.request.referer: *
```

- Results from dns.log where the DNS answers field is empty

```
event.dataset: dns AND NOT dns.answers.name: *
```

Wildcards

Lucene:

```
dns.question.name: google*
```

```
dns.\*: *google*
```

- Basic wildcard searching (not regex)
- Not efficient, but powerful!
- Allows for a “contains” function
- Can be using with the field or value of a query
- Find all connections where a responder replied with an ACK

```
event.dataset: conn AND conn.history: *a*
```

- Search “url.domain” for a value that ends with “.com”

```
url.domain: *.com
```


Wildcards

Lucene:

```
dns.question.name: google*
```

```
dns.\*: *google*
```

- Basic wildcard searching (not regex)
- Not efficient, but powerful!
- Allows for a “contains” function
- Can be using with the field or value of a query
- Find all connections where a responder replied with an ACK

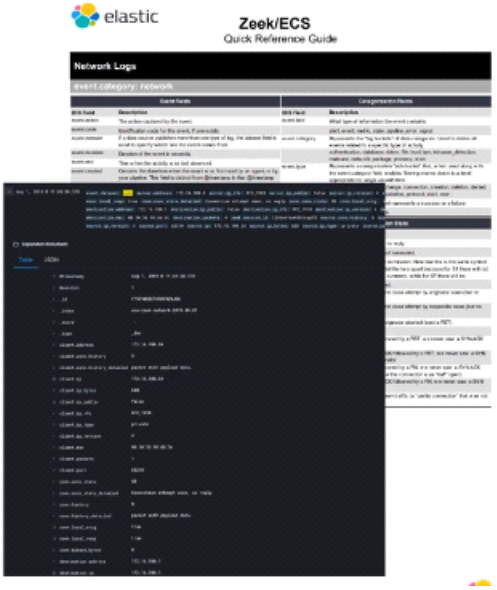
```
event.dataset: conn AND conn.history: *a*
```

- Search “url.domain” for a value that ends with “.com”

```
url.domain: *.com
```


Harvesting Fields

- Methods to find field names:
  - Zeek/ECS Cheat Sheet
  - Expand a single doc in Discover
  - Use a raw text query, dig into resulting document



Advanced Search: Numbers

- Search exact match

```
client.port: 56689
```

- Greater/Less than or equal to

```
client.port >56689
```

```
client.port <=56689
```

Lucene:

```
client.port: >56689
```

```
client.port: [56689 TO 56692]
```

```
client.port: [56689 TO *]
```

- Define a numerical range

```
client.port >20 AND client.port <100
```

Advanced Search: Regular Expressions

- Lucene only!
- Apache Lucene regex
  - similar to PCRE
- Encapsulate the query in forward slashes

```
http.request.method: /[Gg][Ee][Tt]/
```

```
dns.answers.name.text: /([0-9]{1,3}\.){3}[0-9]{1,3}/
```

Full-text search

- Search for any phrase
- Not efficient, but powerful!
- Handy for when you aren't sure which field to search
- Use sparingly, can provide unexpected results depending on ES/Kibana Engineering
- Search for the term "amazon" across all fields*

```
amazon
```

- Search all fields* in the http log that don't contain the term "google"

```
event.dataset: http AND NOT google
```

Advanced Search: Fuzzy

- Lucene only!
- Search for terms that are “similar”

```
user_agent.original.text: Mozilla~
```

```
user_agent.original.text: Mozilla~ AND
NOT user_agent.original.text: Mozilla*
```

- “.text” for keyword mapped fields
  - user_agent.original (keyword = exact match)
  - user_agent.original.text (tokenized!)

Advanced Search: Proximity

- Lucene only!
- Search for terms within a defined distance of each other
  - distance is # of tokenized terms after the first hit

```
user_agent.original.text: “Windows x64”~5
```

- Order is important

```
user_agent.original.text: “Windows x64 Win64 KHTML”~4
```

Searching - Final Notes

- Saved searches
- Sharing is caring!
  - Don’t be a monster, use the short URL
- Not every document will have every field, understanding your underlying data is the best way to build effective searches
- The search bar can be used to search all the data inside Elasticsearch

Overview

- Data is often complex and involves many dimensions
- Often, we want summarized insights:
  - Slices based on specific attributes
  - Calculations based on specific attributes
- In the Elastic Stack we call this an **aggregation**
- All aggregations are performed at Elasticsearch, Kibana just renders the results
- Kibana displays aggregations in the form of **visualizations**

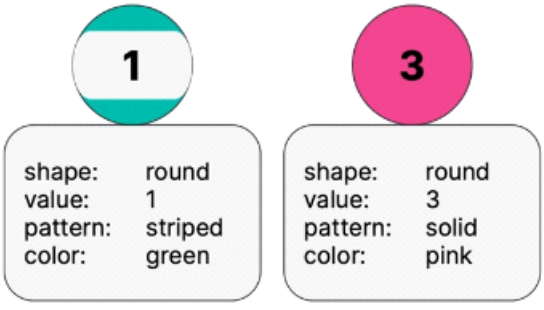
Metrics Aggregation

- Calculates numerical values over a set of documents
- Similar to how **values** are **summarized** in a **pivot table** for a specific column
- Mathematical operation that outputs
  - a single value (eg., **avg, sum, min, max, unique count**)
  - or multiple values (eg., **percentiles, percentile_ranks**)

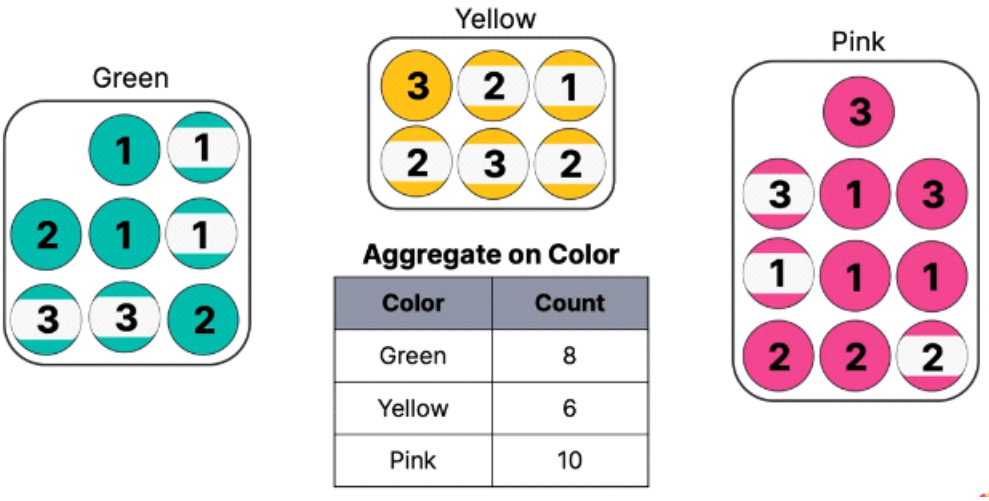
Bucket Aggregation

- A way of **slicing** data
- Similar to grouping by values in **rows** or **columns** in a **pivot table**
- Creates **buckets**
  - collection of documents that share a common criterion
  - can have one or more metrics associated with it
  - number of documents (doc count) per bucket is default metric

Data Attributes Scenario



Bucket Aggregation - Aggregate on Color

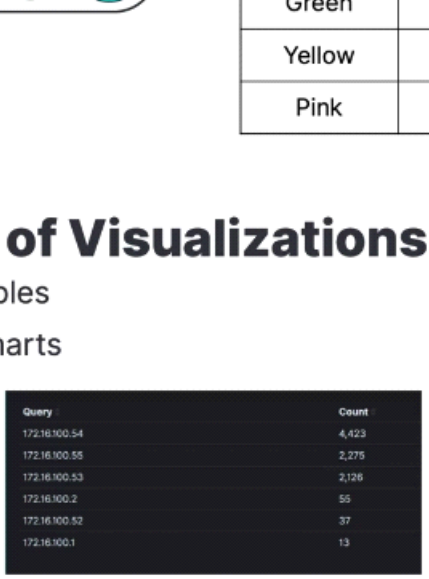


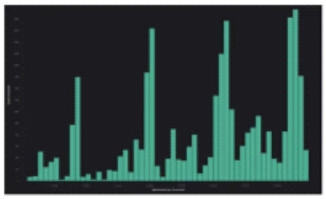
Aggregate on Color

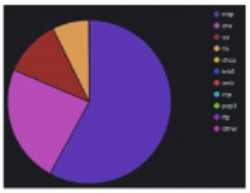
Color	Count
Green	8
Yellow	6
Pink	10

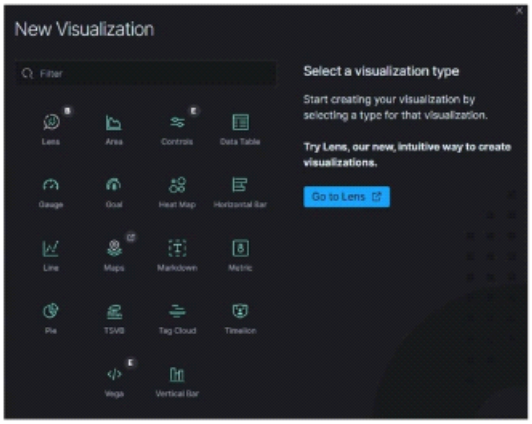
Types of Visualizations

- Data Tables
- Basic Charts



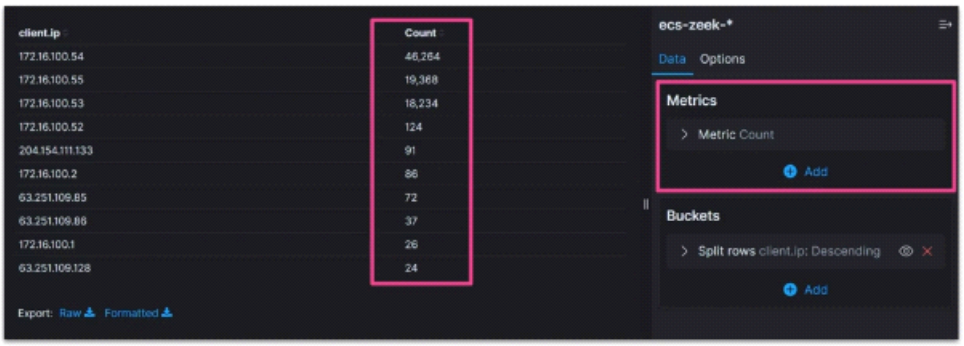






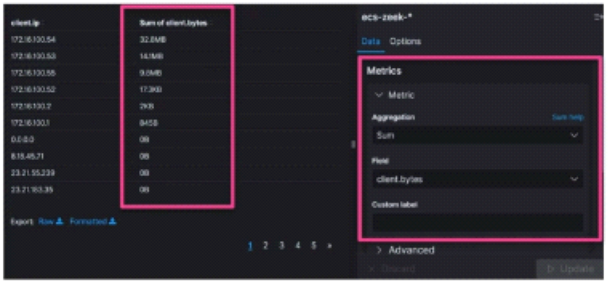
Metric Aggregation - Count

- Default metric for visualizations
- Counts the total number of records/documents



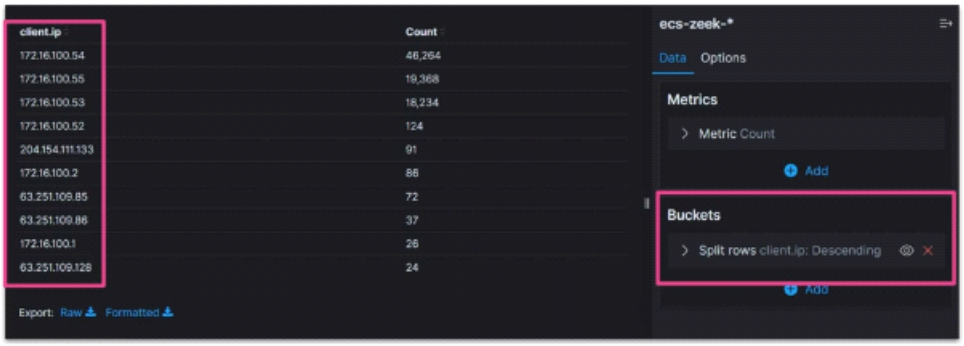
Metric Aggregation - Math Functions

- More than just counting records
- Math functions against the values of certain fields
- User has to specify the field to run calculations against



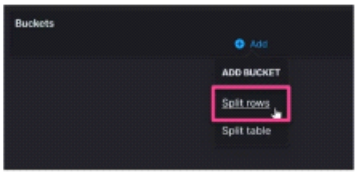
Bucket Aggregation

- Buckets allow you to split your data based on the value in a field

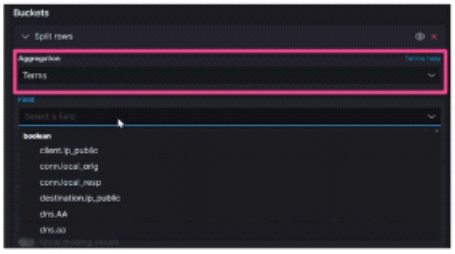


Bucket Creation

- When adding a bucket, select “Split Rows” most of the time, unless you need multiple tables

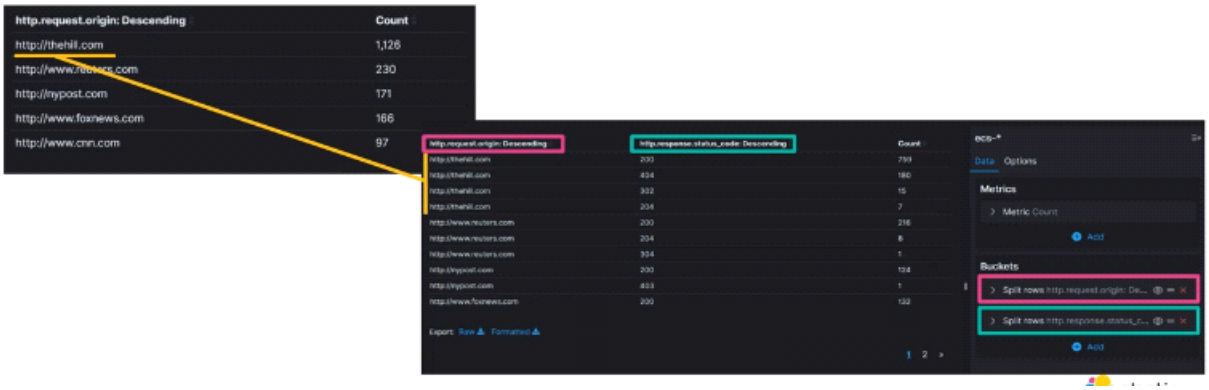


- Selecting the “terms” option under aggregation will open a sub-menu that allows you to choose a field



Sub-Bucket Aggregation

- Subdivides buckets into small groupings
- Groups records based on different combinations of values in specified fields

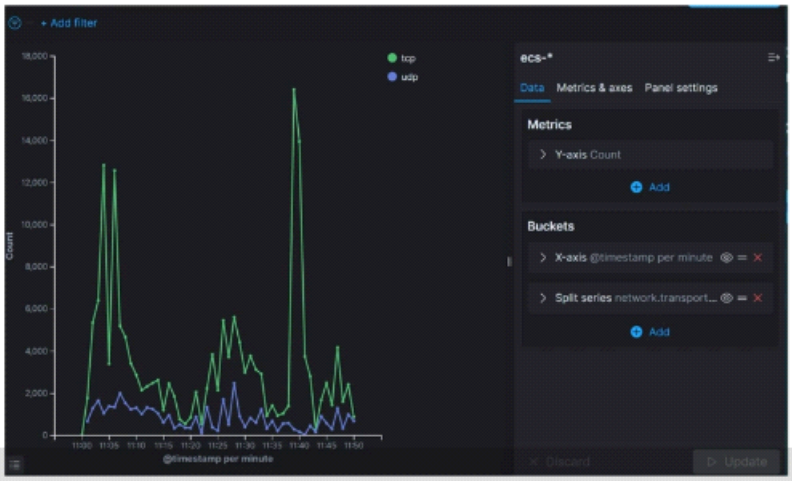


Graphical Visualization

==========

Types of Graphical Visualizations

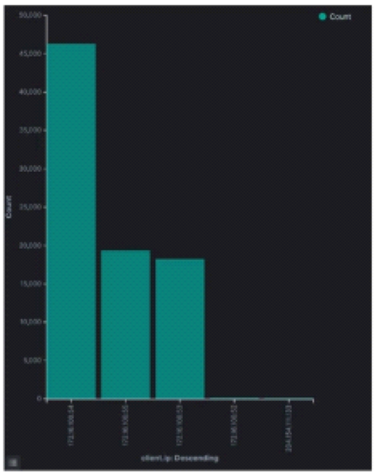
- Basic Charts
- Bar
- Line
- Pie
- Area



Data Tables vs Graphical Visualizations

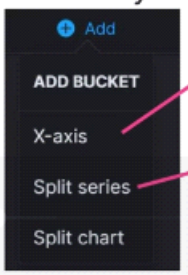
- Different ways to view the same data
- Each view has pros and cons

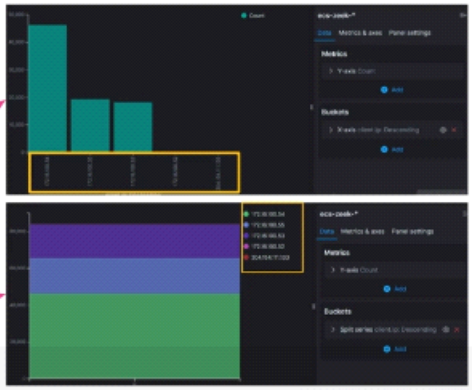
client.ip: Descending	Count
172.16.100.54	46,264
172.16.100.55	19,368
172.16.100.53	18,234
172.16.100.52	124
204.154.111.133	91



Bucket Aggregations for Graph-based Visualizations

- Graph Visualizations approach buckets differently
- X-Axis will specify what buckets run along the bottom of the table
- Split-Series will subdivide existing elements and usually add to the legend





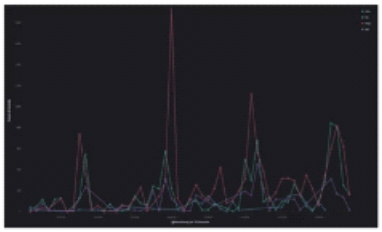
elastic

Bucket Aggregations for Time

- Select Bucket > X-axis > Aggregation > Date Histogram

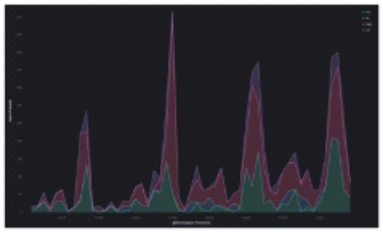
Types of Visualizations - Time based

Line



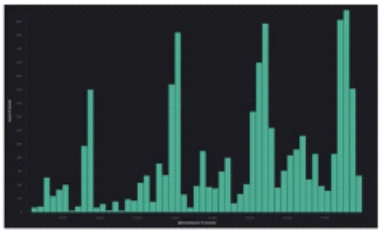
- Compare one value to another
- Difficult to determine cumulative totals

Area



- Combines cumulative and granular views of data
- Can be misleading if read incorrectly

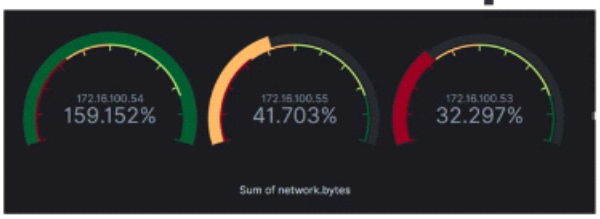
Vertical Bar

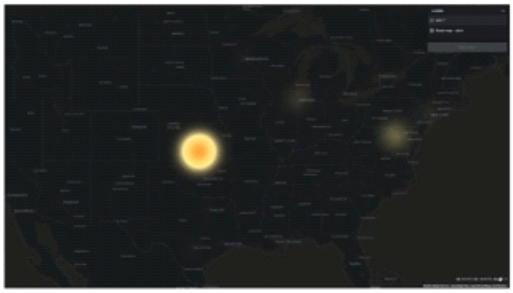


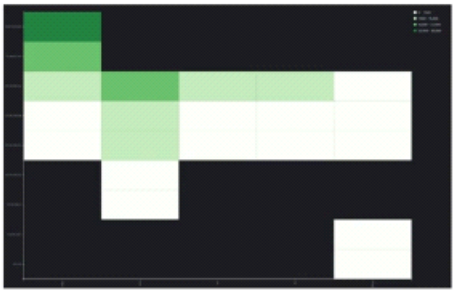
- Best used with a single value/bucket

Types of Visualizations - Out of Scope

- Maps
- Heatmaps
- Goals/Gauges



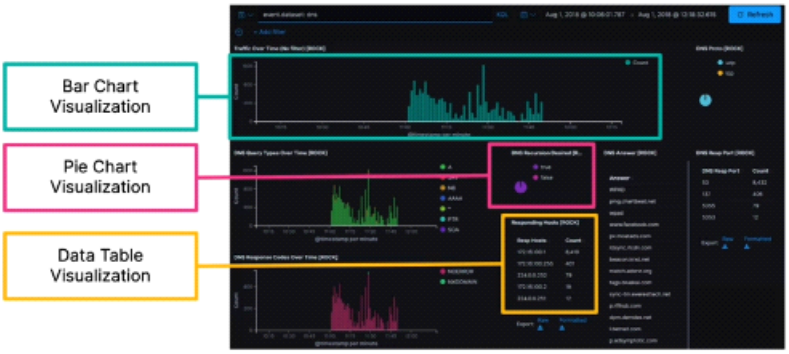




Dashboards
==========

Dashboards

- Dashboards are simply a collection of visualizations
- Focus on one data type/activity at a time
- Avoid making "one dashboard to rule them all"



Dashboards - Can you find the conn.log data?

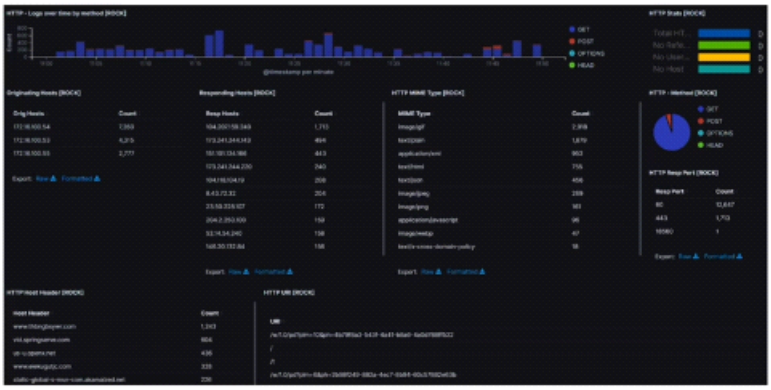
- Conn Dashboard

Fields from the conn.log

- id.orig_h
- id.orig_p
- id.resp_h
- id.resp_p
- protocol

Dashboards - Can you find the http.log data?

- HTTP Dashboard

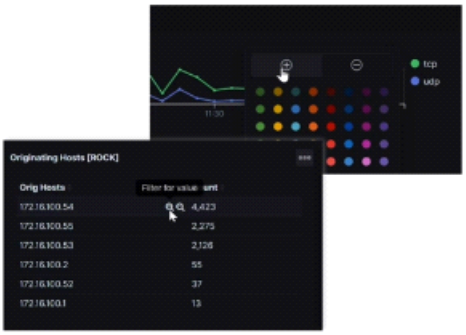


Fields from the http.log

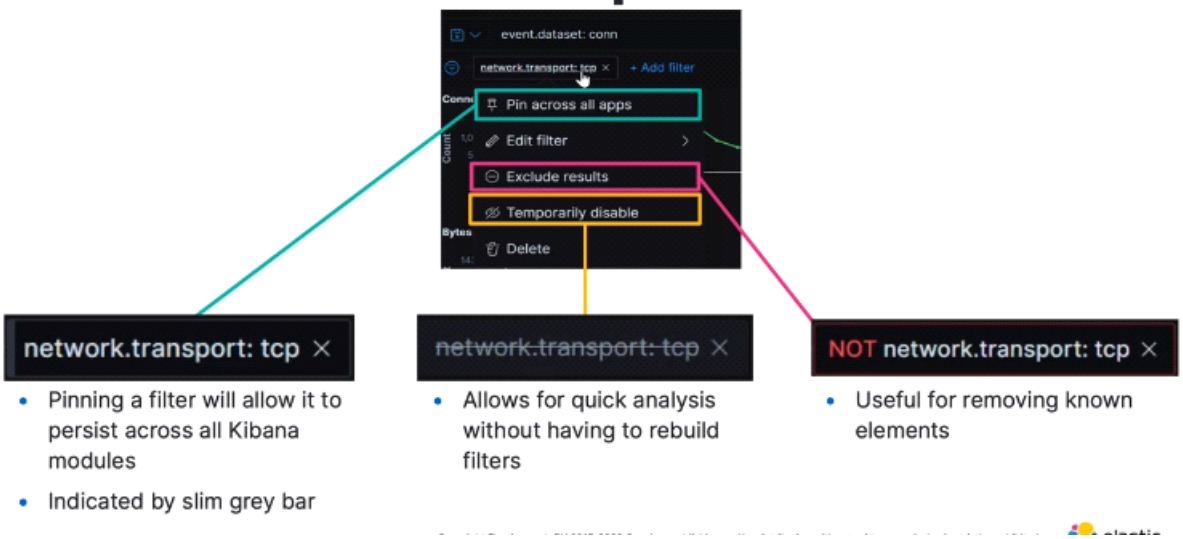
- id.orig_h
- id.orig_p
- id.resp_p
- method
- host
- uri
- resp_mime_types

Filters

- Filter for any value from any visualization
- Impact all visualizations/data
- Filters appear right below the search bar
- Can be renamed for clarity



Filters - Modification Options

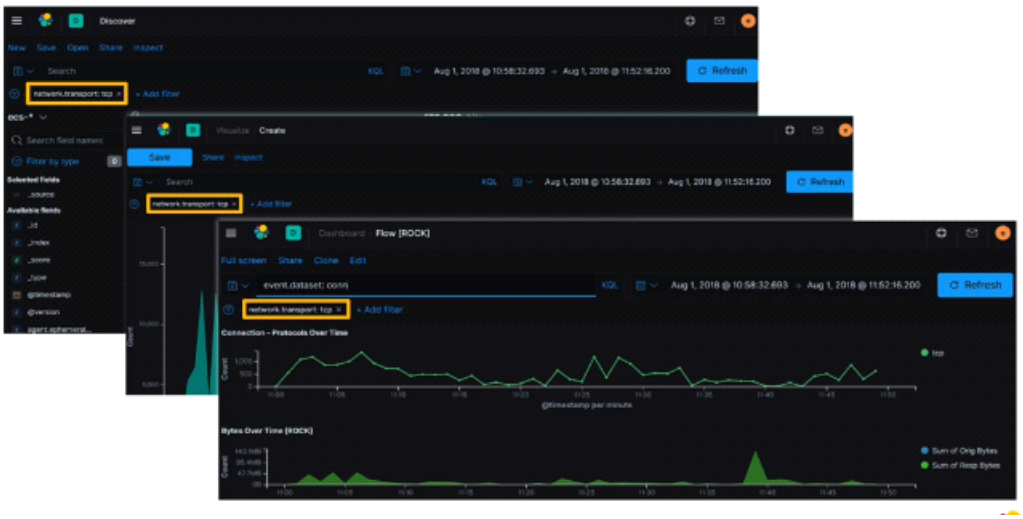


- Pinning a filter will allow it to persist across all Kibana modules
- Indicated by slim grey bar

- Allows for quick analysis without having to rebuild filters

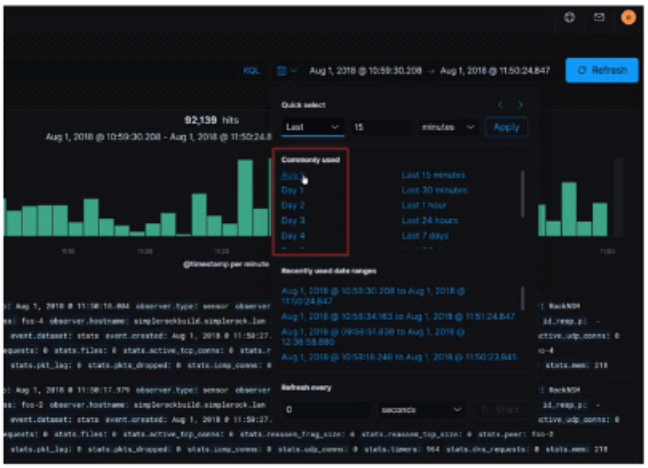
- Useful for removing known elements

Filters - Pin to Pivot Across Kibana Modules



Time Range

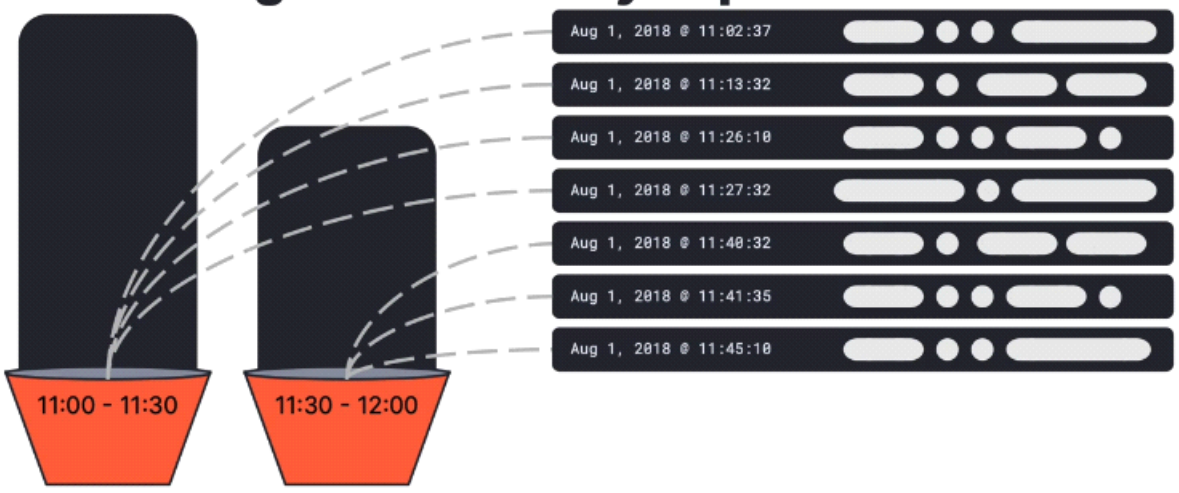
- Use the pre-build date ranges we have provided
- Be careful to reset your time/date picker after you have exhausted a lead



Time Range - Granularity

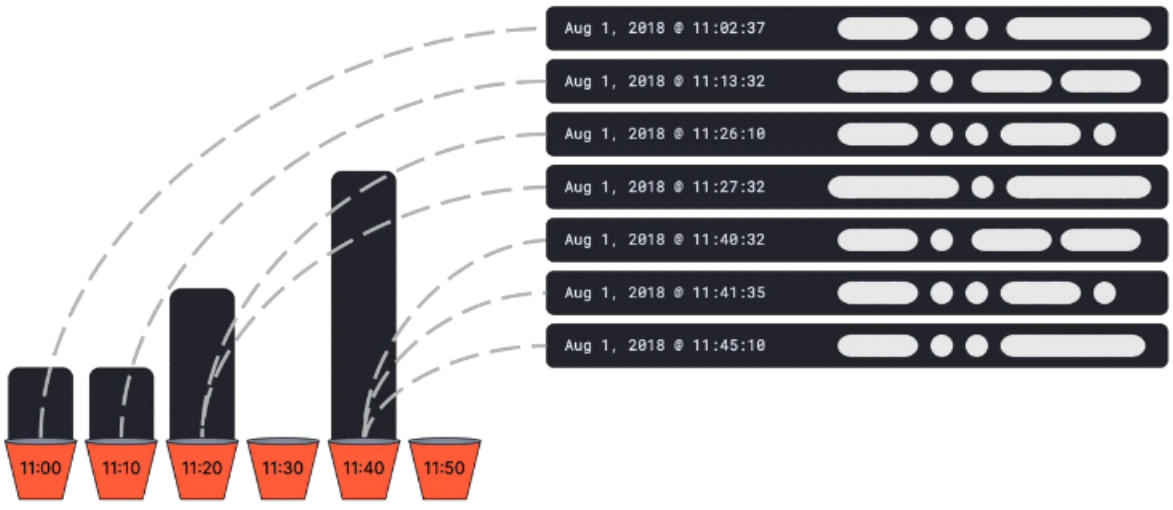
- Changing the date/time range will impact the granularity of the Histogram
- The smaller the time range, the more detail you get
- Use caution when trying to determine when a spike occurred

Time Range - Granularity Impact



@Timestamp per 30 Minutes

Time Range - Granularity Impact



@Timestamp per 10 Minutes

elastic