**Workshop WiFi**
**SSID: BSides-W**
**Pass: BSidesLondon1999?**

BSides London 2024 Workshop

# Taking the garbage out!

By Guy Kramer & Kyle Pearson

## Agenda

- Workshop Introduction
- Usecase Lifecycle
- Documentation
- Common log sources and collection methods
- Log source collection (inc practical exercise)
- Best practices to identify
    - Use case definition
    - Log verbosity
- RegEx Introduction (inc practical exercise)
- Review parsed example log source

## Workshop Introduction

## Workshop Introduction

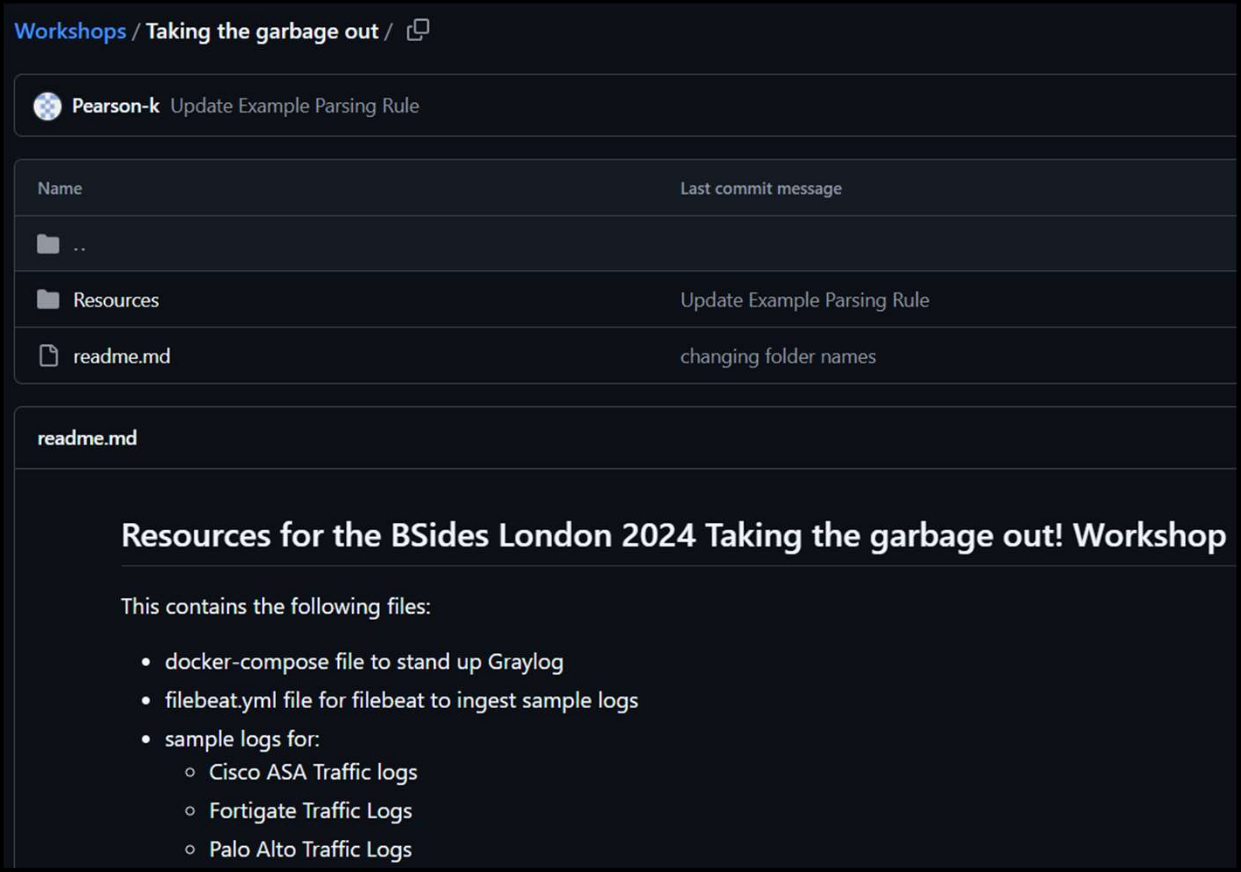
**What this is**
This is a high-level overview of technical and business considerations, while trying to balance collect vs only collect the events that matter.

**What this is not**
In-depth zero to hero

Docker image and files used in the exercises are located on a GitHub Repo, go to link below
https://github.com/cia-uk/Workshops/

**Requirements**
Docker Desktop or equivalent
Filebeat

Please download if you don't have either



Workshops / Taking the garbage out /
Pearson-k Update Example Parsing Rule

| Name | Last commit message |
|---|---|
| .. | |
| Resources | Update Example Parsing Rule |
| readme.md | changing folder names |

readme.md

Resources for the BSides London 2024 Taking the garbage out! Workshop

This contains the following files:

- docker-compose file to stand up Graylog
- filebeat.yml file for filebeat to ingest sample logs
- sample logs for:
  - Cisco ASA Traffic logs
  - Fortigate Traffic Logs
  - Palo Alto Traffic Logs

## Whoami?

Mr Guy Kramer

**BCS** (British Computer Society)
**CIISec** (Chartered Institute of Information Security)
**FIRST** (Forum of Incident Response and Security Teams)
**NCSC CyberFirst**

**LinkedIN:** www.linkedin.com/in/guy-kramer/

**Email:** gkramer@ciauk.ltd

## Whoami?

Mr Kyle Pearson

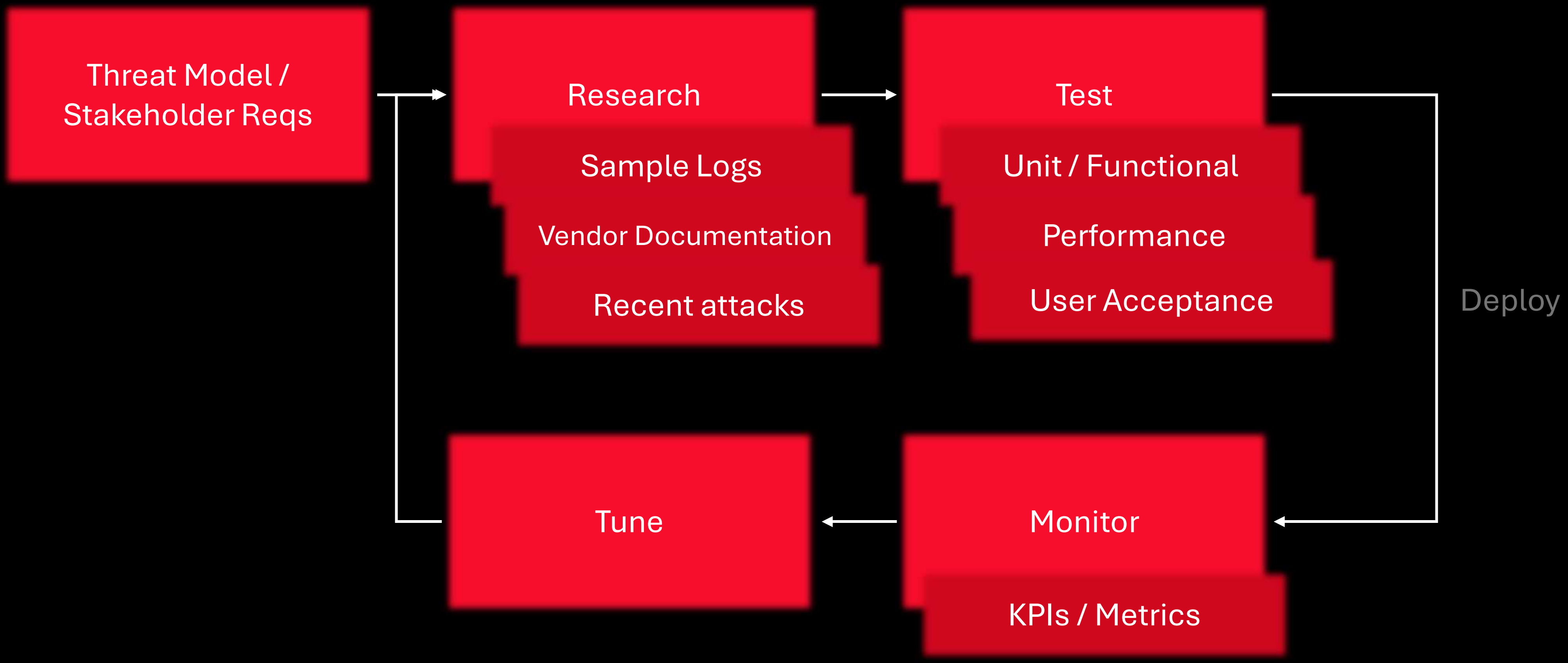
SIEM Wizard
Hand Holder
Problem Solver
Line and Box Drawer

LinkedIn: https://www.linkedin.com/in/kylepearsondf/

Email: kyle.pearson@graylog.com

## Usecase Lifecycle

## Usecase Lifecycle



## Documentation

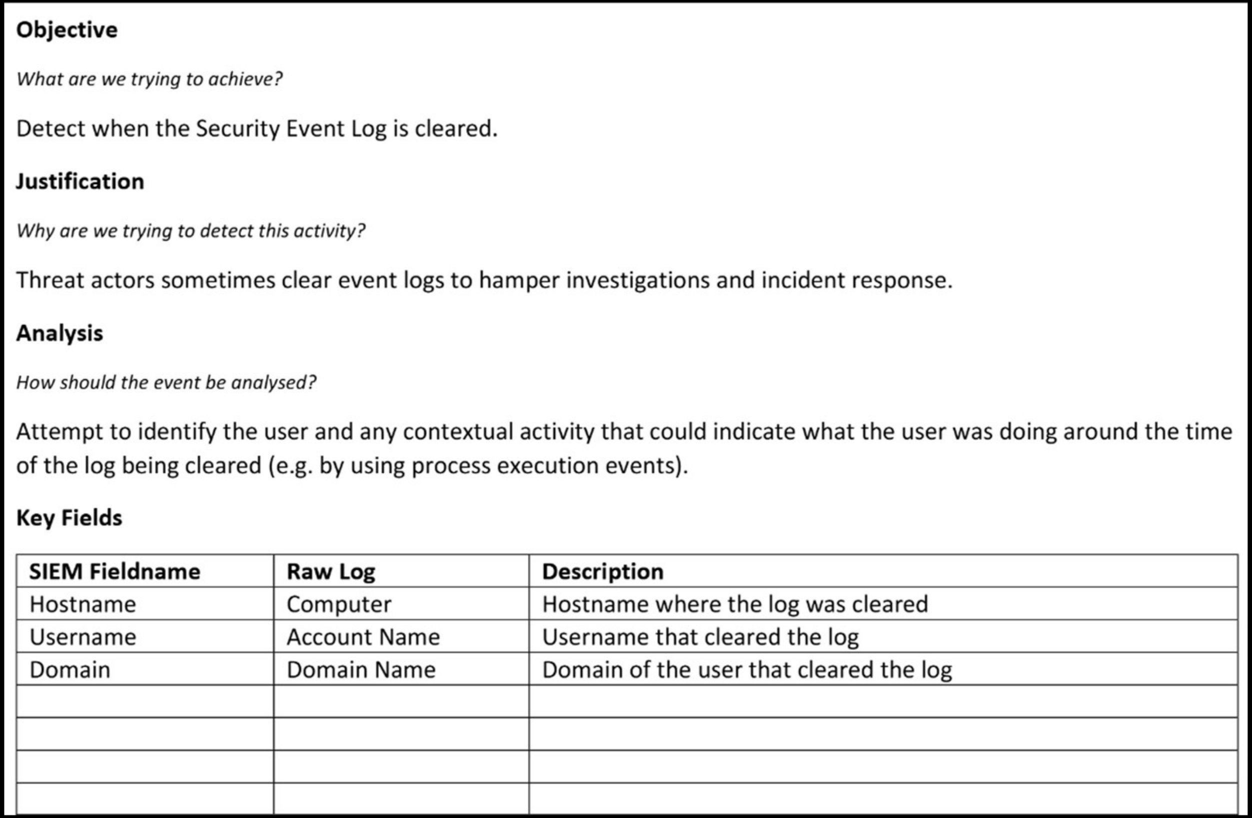
# Documentation

Documentation is a key asset to and maintain, that nobody like to do.

- Business and technical issue the usecase is mitigating
- What was the logic
- What thresholds?
- False-Positive rate?
- Engineering time

**Note**

- Package into a bundle and keep for reference or troubleshooting



**Objective**

*What are we trying to achieve?*

Detect when the Security Event Log is cleared.

**Justification**

*Why are we trying to detect this activity?*

Threat actors sometimes clear event logs to hamper investigations and incident response.

**Analysis**

*How should the event be analysed?*

Attempt to identify the user and any contextual activity that could indicate what the user was doing around the time of the log being cleared (e.g. by using process execution events).

**Key Fields**

| **SIEM Fieldname** | **Raw Log** | **Description** |
|---|---|---|
| Hostname | Computer | Hostname where the log was cleared |
| Username | Account Name | Username that cleared the log |
| Domain | Domain Name | Domain of the user that cleared the log |
| | | |
| | | |
| | | |
| | | |

## Common log source and Collection methods

## Common log source and Collection methods

Not all log sources are created equal, some are easy, some are not!

You must think about where the device logs, how it logs and how to collect it

Your log source may log to:
- Console (just displayed on screen)
- File, line by line or multi-line
- Database

You need to think about how you collect it, agent/agent-less

- Where that data is coming from?
- is it in house
- is it SaaS
- can you access the logs directly
- is it via an API

## Common log source and Collection methods

Some Examples:

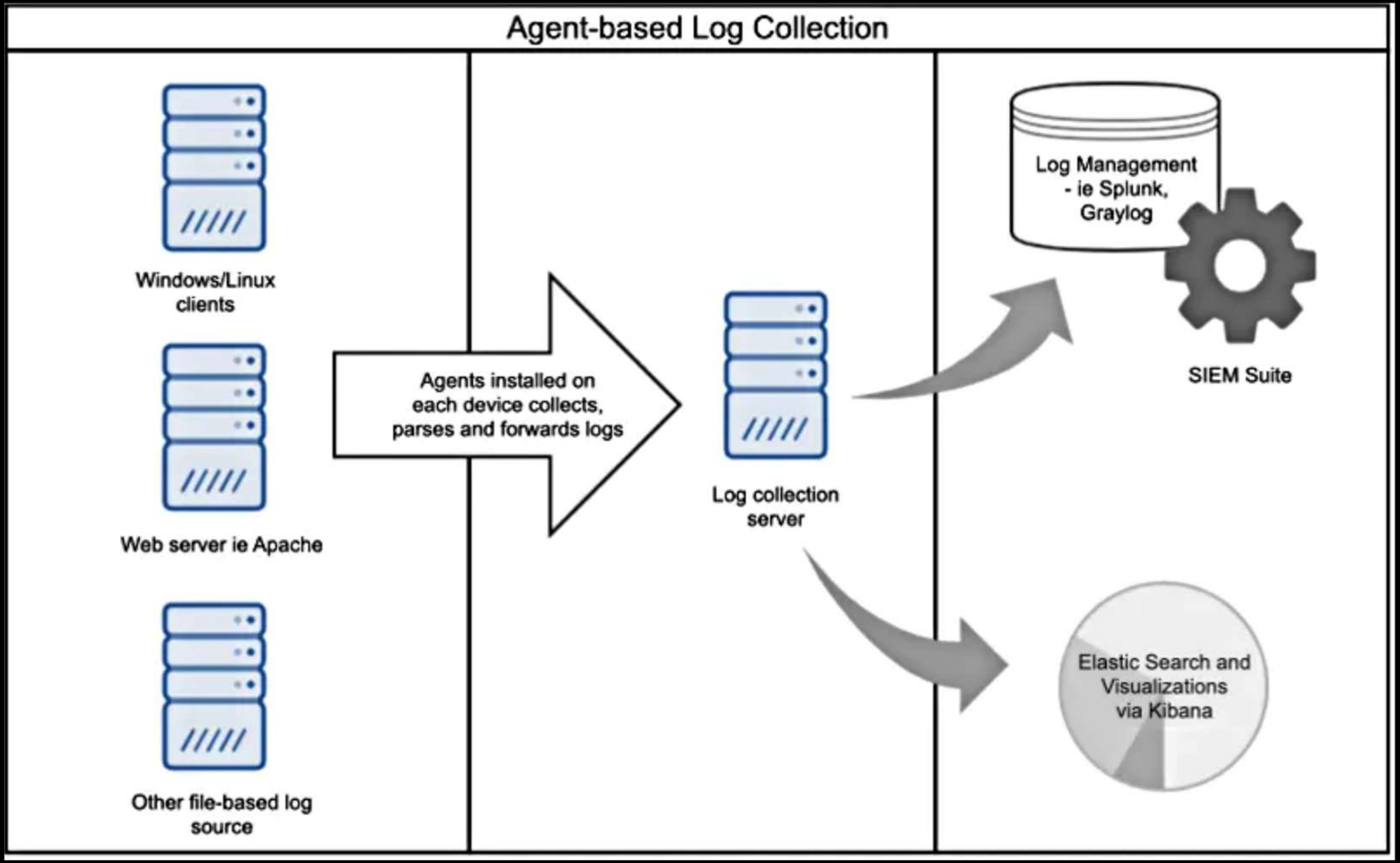
- Windows Event Log – requires agent to interact with windows API (for most part)
- EDR Products (Sentinel, CrowdStrike etc) - mainly require pulling from the API.
- Network equipment (Cisco, Palo Alto, Fortinet etc) mainly utilise Syslog and push logs.
- AWS – s3 buckets, consumed directly via API or via SQS/SNS for queuing.
- GCP – logsinks, can be pushed to BigQuery and consumed
- Azure – can be read via Azure monitor, or Azure Event Hubs.

## Common log source and Collection methods

Some good free or opensource log shippers are:

- Beats
- Logstash
- NXLog
- Fluentd
- Vector

If you are using a paid solution, you may have a vendor specific shipper such as Splunk Universal / Heavy Forwarders.


Agent-based Log Collection
Windows/Linux clients
Web server ie Apache
Other file-based log source
Agents installed on each device collects, parses and forwards logs
Log collection server
Log Management - ie Splunk, Graylog
SIEM Suite
Elastic Search and Visualizations via Kibana

## Common log source and Collection methods

The log you collect may have a wide variety of formats.

Common formats include:

- Plaintext
  - network connection from 1.1.1.1 to 2.2.2.2

- Key value pairs
  - event_type=Network connection source_ip=1.1.1.1 destination_ip=2.2.2.2

- Delimited (CSV etc)
  - network connection,1.1.1.1,2.2.2.2

- JSON

  -
    ```
    {
        "event_type": "Network Connection",
        "source_ip": "1.1.1.1",
        "destination_ip": "2.2.2.2"
    }
    ```

- XML

## Common log source and Collection methods

Within the sample logs provided for thw workshop, you have been given some logs from three major firewall vendors, Cisco, Palo Alto and Fortinet have a look at the formats.

Palo Alto:

-
  ```
  1,2024/12/09 18:04:50,007200002538,TRAFFIC,start,2305,2024/12/09 18:04:50,192.168.92.15,192.168.92.15,,,Allowed
  Traffic,,,Teams,vsys2,Internal,Internal,ethernet1/2,ethernet1/2,,2024/12/09
  18:04:50,63393,1,65001,8080,0,0,0x8000000000000000,UDP,allow,111,92,19,11,2024/12/09
  18:04:50,35,Allowed,0,60144,0x8000000000000000,United Kingdom,United Kingdom,0,2,9,aged-out,31,12,0,0,,palo01,from-
  policy,,,0,,0,,N/A,0,0,0,0,28b956cb-00f7-4508-a3eb-04b5e8c4ab1d,0,0,,,,,,
  ```

Fortinet:

-
  ```
  DEV01 date=2024-12-09 time=18:07:01 logid="11030201" type="traffic" subtype="forward" level="notice" vd="root"
  eventtime=1635361817042248679 tz="-0700" srcip=192.168.92.6 srcport=45181 srcintf="port4" srcintfrole="undefined"
  dstip=213.154.64.1 dstport=80 dstintf="port2" dstintfrole="wan" poluuid="707a0d88-c972-51e7-bbc7-4d421660557b"
  srccountry="Reserved" dstcountry="Australia" sessionid=771 proto=6 action="close" policyid=1 policytype="policy"
  service="HTTPS" trandisp="snat" transip=172.16.200.45 transport=45181 appid=40568 app="HTTPS.BROWSER"
  appcat="Web.Client" apprisk="medium" duration=2 sentbyte=25 rcvdbyte=129 sentpkt=0 rcvdpkt=0 appcat="unscanned"
  crscore=30 craction=131072 crlevel="high" user="asheppard"
  ```

Cisco Asa:

-
  ```
  %ASA-6-302015 Built inbound UDP connection 555555 for inside:192.168.92.24/43314 (192.168.92.24/43314) to
  outside:8.8.8.8/8080 (8.8.8.8/8080)
  ```

## Log Source Collection: Practical Exercise 1

## Log Source Collection: Practical Exercise 1

Today we are going to use **Graylog** as our log aggregator ran within docker and filebeat as an agent to collect some log files.

We have provided the docker-compose file and sample logs for the workshop, you need a way to run the docker-compose, the easiest way is to install docker desktop (Mac/Windows/Linux)

**https://docs.docker.com/desktop/**

You will also need to download filebeat for your operating system, we will run this in the foreground so no need to install as a service.

**https://www.elastic.co/downloads/beats/filebeat**

In the Github repo link you will find all the instructions:

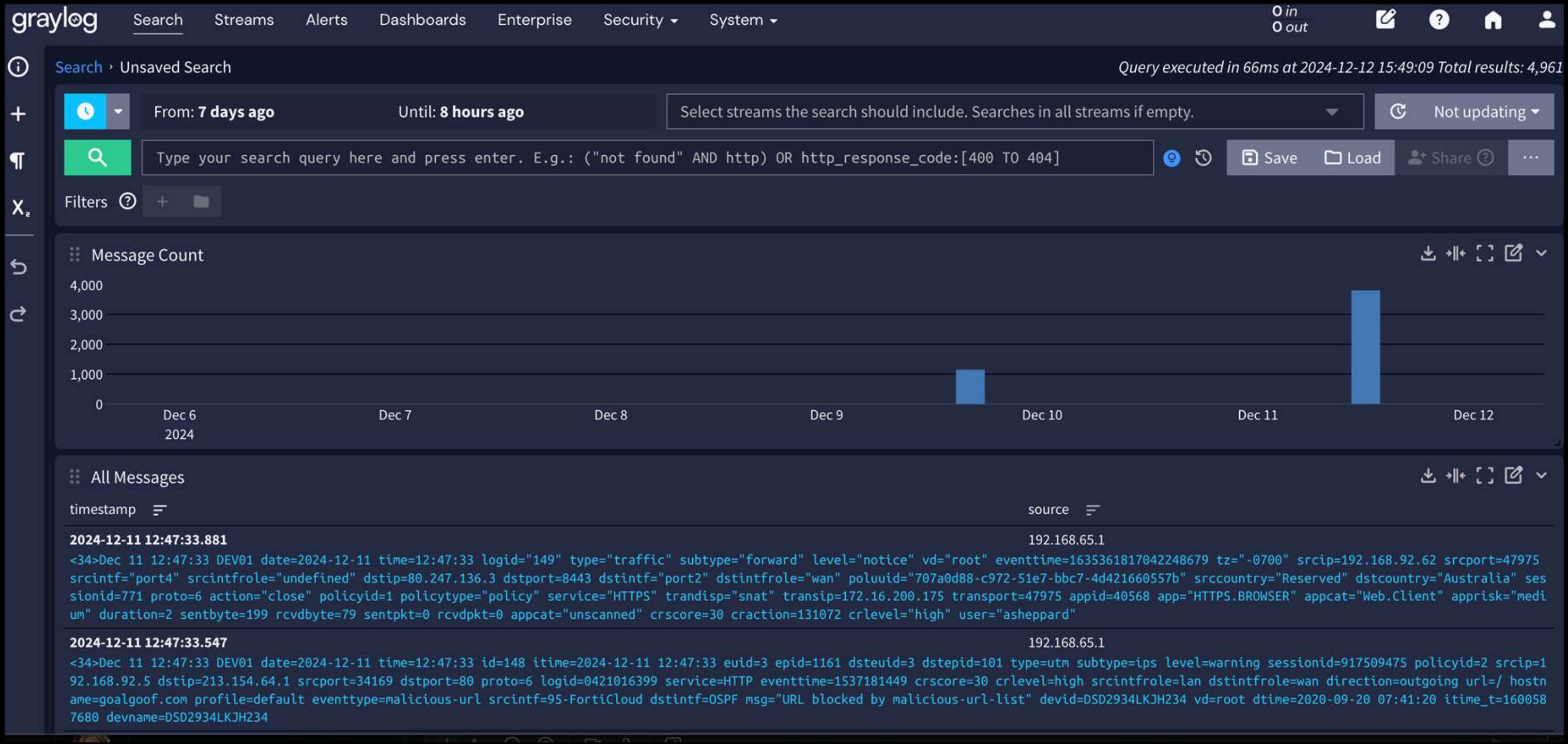
**https://github.com/cia-uk/Workshops/tree/master/Taking%20the%20garbage%20out**

**Target objective**

Get logs into Greylog via Filebeat, validate the within the platform and run searches to interrogate the logs

## **Log Source Collection:** Practical Exercise 1

If you can see your logs when you hit search you have succeeded.

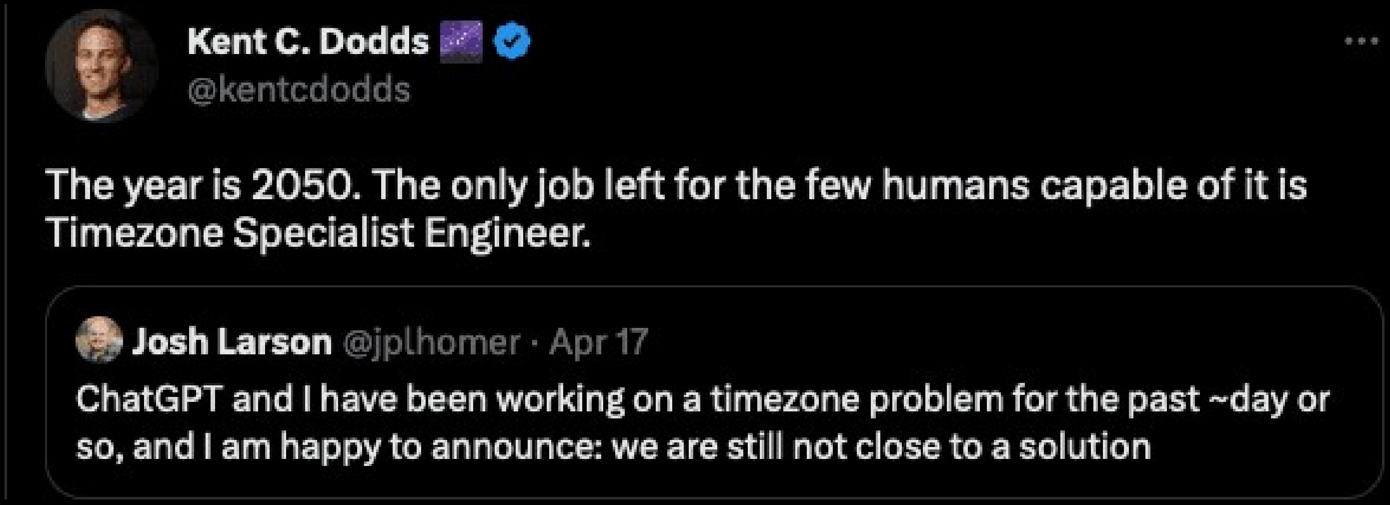


*Query executed in 66ms at 2024-12-12 15:49:09 Total results: 4,961*

```
timestamp                                                                      source
2024-12-11 12:47:33.881                                                        192.168.65.1
<34>Dec 11 12:47:33 DEV01 date=2024-12-11 time=12:47:33 logid="149" type="traffic" subtype="forward" level="notice" vd="root" eventtime=1635361817042248679 tz="-0700" srcip=192.168.92.62 srcport=47975
srcintf="port4" srcintfrole="undefined" dstip=80.247.136.3 dstport=8443 dstintf="port2" dstintfrole="wan" poluuid="707a0d88-c972-51e7-bbc7-4d421660557b" srccountry="Reserved" dstcountry="Australia" ses
sionid=771 proto=6 action="close" policyid=1 policytype="policy" service="HTTPS" trandisp="snat" transip=172.16.200.175 transport=47975 appid=40568 app="HTTPS.BROWSER" appcat="Web.Client" apprisk="medi
um" duration=2 sentbyte=199 rcvdbyte=79 sentpkt=0 rcvdpkt=0 appcat="unscanned" crscore=30 craction=131072 crlevel="high" user="asheppard"
2024-12-11 12:47:33.547                                                        192.168.65.1
<34>Dec 11 12:47:33 DEV01 date=2024-12-11 time=12:47:33 id=148 itime=2024-12-11 12:47:33 euid=3 epid=1161 dsteuid=3 dstepid=101 type=utm subtype=ips level=warning sessionid=917509475 policyid=2 srcip=1
92.168.92.5 dstip=213.154.64.1 srcport=34169 dstport=80 proto=6 logid=0421016399 service=HTTP eventtime=1537181449 crscore=30 crlevel=high srcintfrole=lan dstintfrole=wan direction=outgoing url=/ hostn
ame=goalgoof.com profile=default eventtype=malicious-url srcintf=95-FortiCloud dstintf=OSPF msg="URL blocked by malicious-url-list" devid=DSD2934LKJH234 vd=root dtime=2020-09-20 07:41:20 itime_t=160058
7680 devname=DSD2934LKJH234
```

## Log Source Collection: Practical Exercise 1

**Note:** All the logs have had their timestamps stripped and will just use the filebeat timestamp. In log management timestamps are the hardest thing to deal with so we will not put you through this pain today.



**Kent C. Dodds** @kentcdodds

The year is 2050. The only job left for the few humans capable of it is Timezone Specialist Engineer.

**Josh Larson** @jplhomer · Apr 17

ChatGPT and I have been working on a timezone problem for the past ~day or so, and I am happy to announce: we are still not close to a solution

## RegEx: Practical Exercise 2

## RegEx: Practical Exercise 2

Ok so everyone hates it, but regex is still widely used to parse log messages.

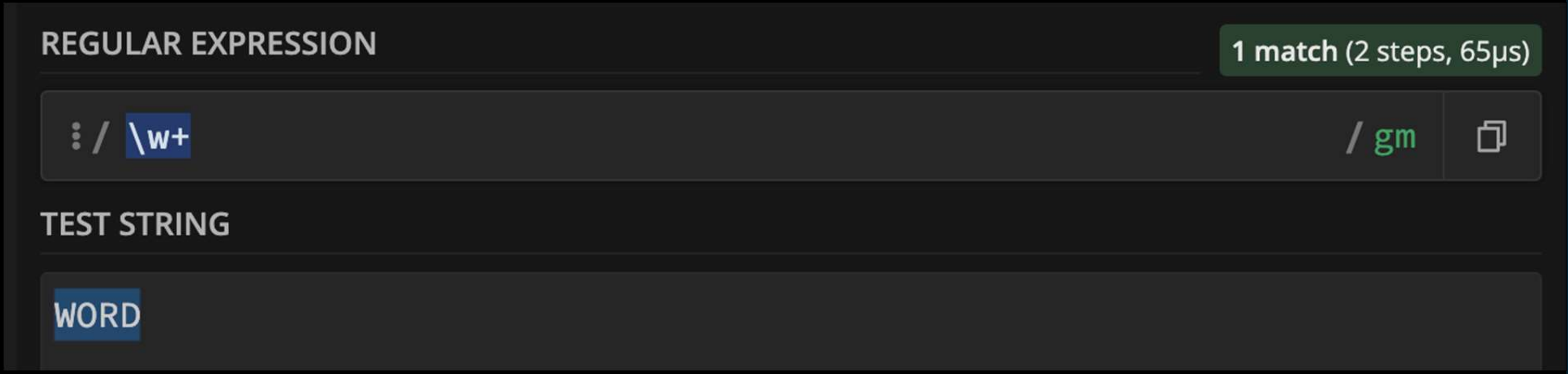
So what is reges anyway?

To put it simply, it's a way of expressing text in a standardised expression format

For example any word, lets use "WORD" can be expressed as "\w+"

To explain -

- \w
  - This means any word character so a-z or A-Z
- +
  - 1 or more of these characters



```
/ \w+ / gm
```

```
WORD
```

## RegEx: Practical Exercise 2

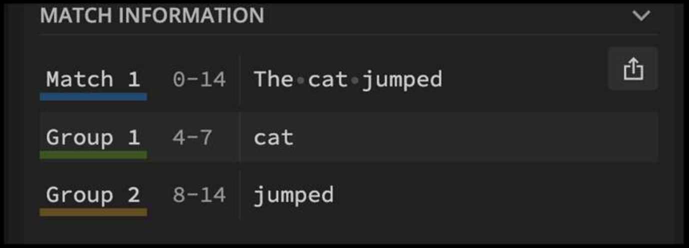
When using regex on logs, we group values that we want to store, and ignore values we do not.

If you want to capture something you can use parentheses ()

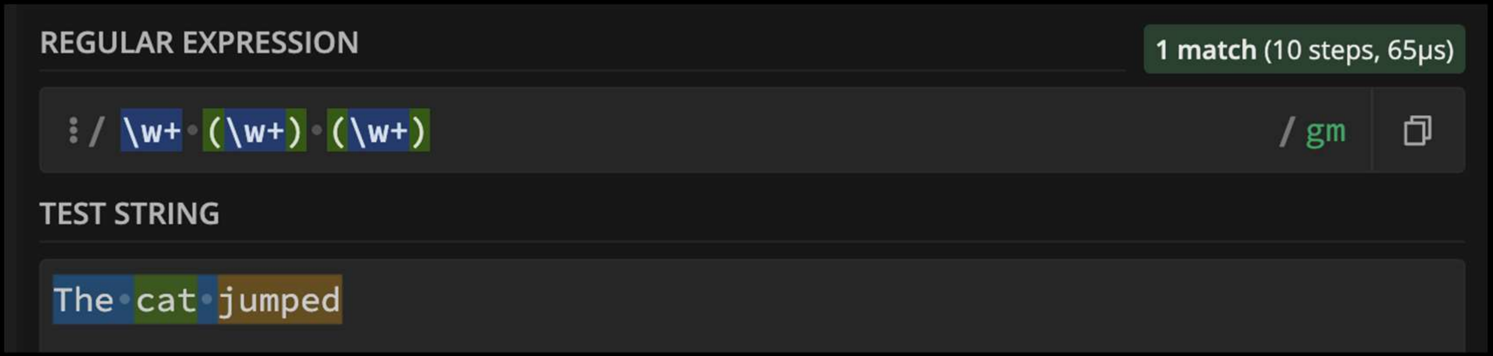
for example in the string "The cat jumped" we may not care about the word "The"

We can express this like "\w+ (\w+)"

If we want "cat" and "jumped" to be two separate captures we'd extend it further to "\w+ (\w+) (\w+)"


MATCH INFORMATION

```
Match 1   0-14   The cat jumped
Group 1   4-7    cat
Group 2   8-14   jumped
```


REGULAR EXPRESSION

1 match (10 steps, 65µs)

```
/ \w+ (\w+) (\w+) / gm
```

TEST STRING

```
The cat jumped
```

## RegEx: Practical Exercise 2

When writing regex, you can always use some free online tools to help, even those of us who have written a lot of regex in our time still sometimes use these, good ones are:

https://regex101.com/

https://regexr.com/

The regex cheat sheet of your choice

Or you can just steal someone else's patterns off stackoverflow (or ask chatGPT)

## RegEx: Practical Exercise 2

Let's start with a simple exercise just to get you familiar with regex, or refresh your mind before tackling our big challenge for this part of the workshop

Let's use the string "Task1 pickup 2 2.5 litre cans of paint from B&Q, maybe buy 50% extra for re-painting later"

It's a mix of letters. Numbers and punctuation so it's a good intro string.

Put the string into regex101 or regexr and write an expression that matches the string and captures the key bits of information.

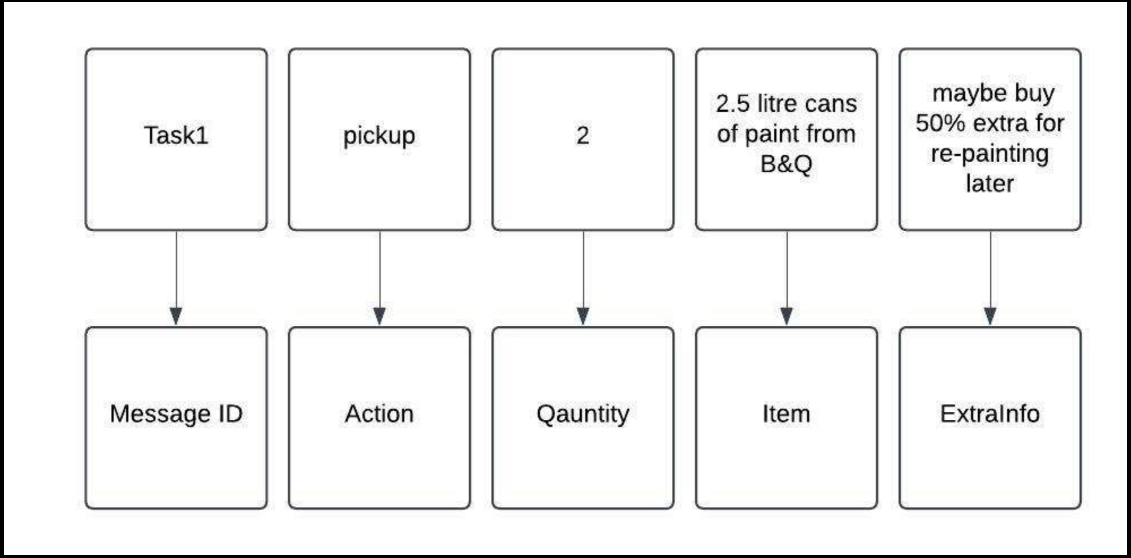
Don't use (.*)

## RegEx: Practical Exercise 2

What did you come up with?

Hopefully you grouped your logs like this, there is no reason to over complicate your captures, it's ok if you went all in, but this simple expression works fine.

```
(\w+\d+) (\w+) (\d+) (.+), (.*)
```


Task1
pickup
2
2.5 litre cans of paint from B&Q
maybe buy 50% extra for re-painting later
Message ID
Action
Qauntity
Item
ExtraInfo

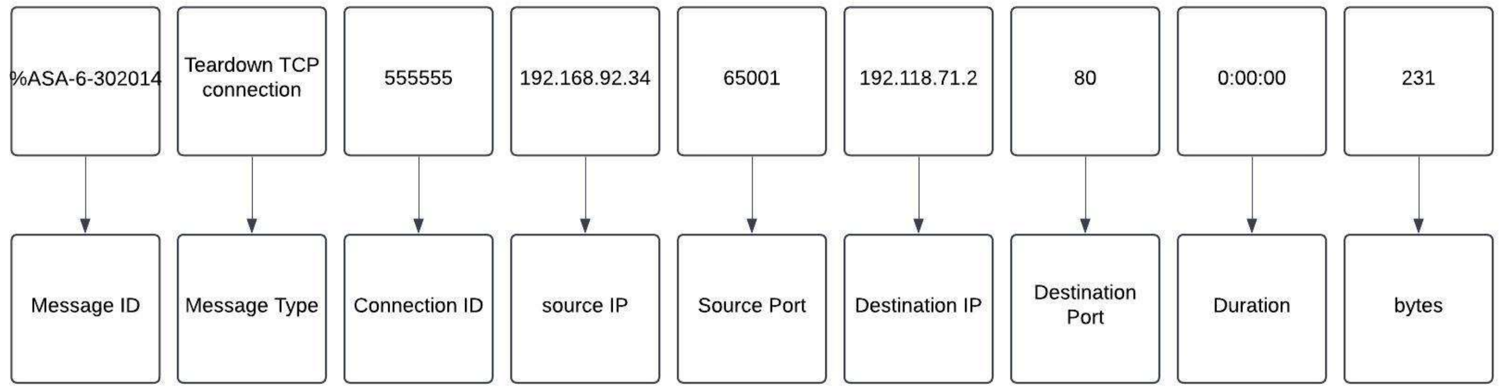
## RegEx: Practical Exercise 2

For this exercise  we are going parse a simple cisco ASA log message 302014 – You may wish to look this up in the documentation but here is what a sample message looks like:

- %ASA-6-302014 Teardown TCP connection 555555 for inside:192.168.92.34/65001 to outside:192.118.71.2/80 duration 0:00:00 bytes 231 TCP FINs

Note the cisco_asa log set that was given to you contains a variety of messages, if you are a regex wizard, feel free to try some of the other messages such as:

- %ASA-6-106100 access-list dmz_acl denied UDP inside/192.168.92.20(51413) -> outside/8.8.8.8(8080) hit-cnt 1 first hit [0x4373afa0, 0x0]



## RegEx: Practical Exercise 2

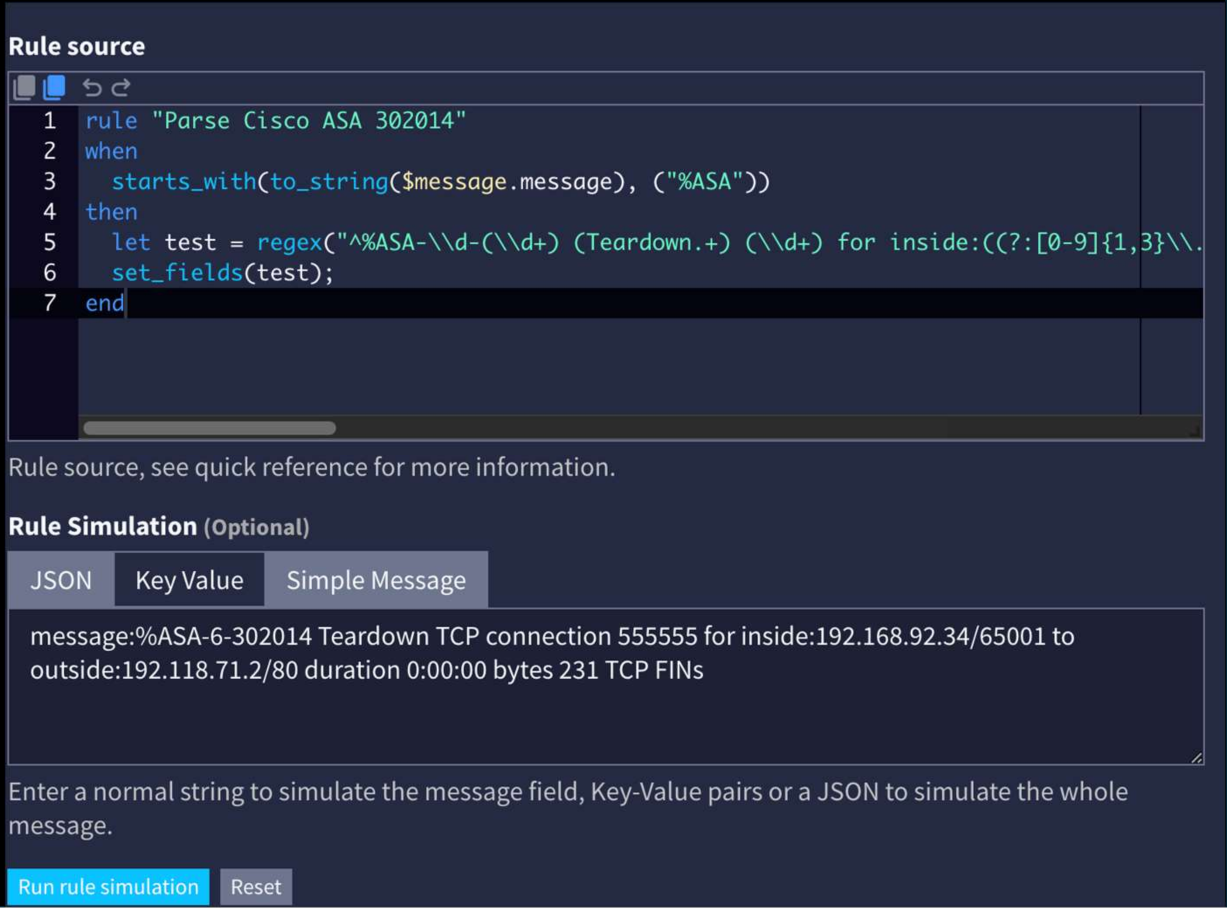
Now you have the regex figured out, lets put it to use in a practical way.

Graylog uses "pipeline rules" to parse logs, lets put our regex into one of these now, inside your Graylog UI, go to system>pipelines we want to create a rule first, not a pipeline.

You must use the code UI as the regex function does not allow you add custom field names in the regex function, I will give you the base pipeline syntax, just plop your regex in.

I have placed this example rule, in the Github repo.

NOTE: Graylog uses Java regex, so turn those slashes into double slashes.



**Rule source**

```
rule "Parse Cisco ASA 302014"
when
  starts_with(to_string($message.message), ("%ASA"))
then
  let test = regex("^%ASA-\\d-(\\d+) (Teardown.+) (\\d+) for inside:((?:[0-9]{1,3}\\.
  set_fields(test);
end
```

Rule source, see quick reference for more information.

**Rule Simulation** (Optional)

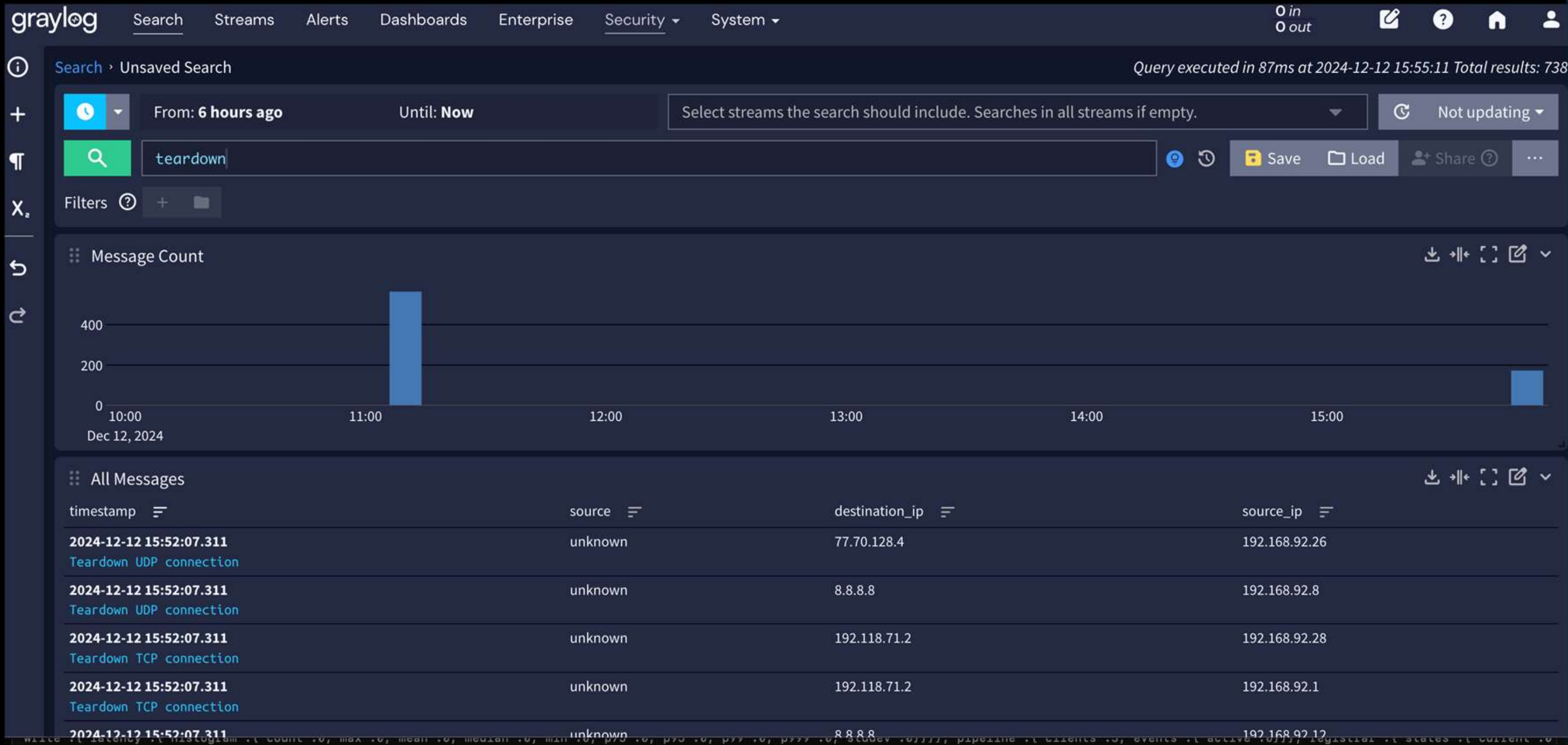
```
message:%ASA-6-302014 Teardown TCP connection 555555 for inside:192.168.92.34/65001 to
outside:192.118.71.2/80 duration 0:00:00 bytes 231 TCP FINs
```

Enter a normal string to simulate the message field, Key-Value pairs or a JSON to simulate the whole message.

# RegEx: Practical Exercise 2

Now you have created your rule, make a pipeline, attach it to the "default stream" and add your rule to stage 0 or stage 1.

Now clear the data directory from filebeat and re-ingest the logs, see that your logs are now parsed.



Query executed in 87ms at 2024-12-12 15:55:11 Total results: 738

```
teardown
```

| timestamp | source | destination_ip | source_ip |
|---|---|---|---|
| 2024-12-12 15:52:07.311<br>Teardown UDP connection | unknown | 77.70.128.4 | 192.168.92.26 |
| 2024-12-12 15:52:07.311<br>Teardown UDP connection | unknown | 8.8.8.8 | 192.168.92.8 |
| 2024-12-12 15:52:07.311<br>Teardown TCP connection | unknown | 192.118.71.2 | 192.168.92.28 |
| 2024-12-12 15:52:07.311<br>Teardown TCP connection | unknown | 192.118.71.2 | 192.168.92.1 |
| 2024-12-12 15:52:07.311 | unknown | 8.8.8.8 | 192.168.92.12 |

## Log Source Parsing: Best Practices

## Log Source Parsing: Best Practices

- Use the best tool for the job, both for collecting the logs but also for parsing the logs.
- Most tools whether they are Open Source, Free or paid have support for common log formats to make your life easier, key value pairs, xml, json, syslog should all be able to be parsed without the need for complicated regex etc.
- Only feed logs that are relevant to the parser, there is no reason to run all logs through all parsers.
- Drop logs if not required as close to the source as possible to avoid introducing unnecessary load on the systems/network.
- Use building blocks when parsing, parse out core parts of the logs, then run additional parsing as needed.
- Think about re-use across different log sources, if you build a parsing rule that extracts IP addresses, can that be used in other parsers?
- Ask for help – Lots of community forums for different tools, ChatGPT etc can also be used.

## Workshop: Wrap-up

## Workshop: Wrap-up

**What have you learned?**

Log Collection
Using Regular Expressions to parse logs
Best practices

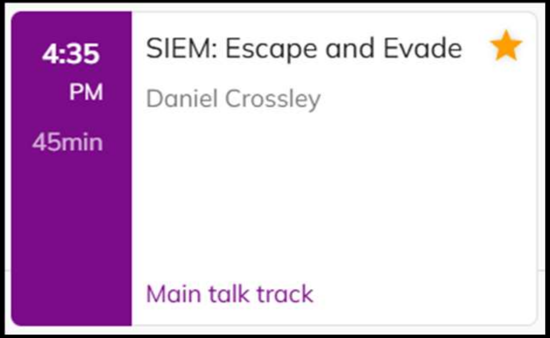
NOTE
- You can work through the practical exercises at your own pace from the Github Repro
- If useful I can create a Discord group for folks to collaborate and

## BSides Talk - SIEM: Escape and Evade

Make sure to catch the SIEM: Escape and Evade on the **main track** at **16:35 GMT**

Speakers:

**Dan Crossley**
**Niall Errity**
**Guy Kramer**



4:35
PM
45min

SIEM: Escape and Evade

Daniel Crossley

Main talk track

# Thank You