RSAConference2018
San Francisco | April 16–20 | Moscone Center

SESSION ID: CSV-R14

FIM AND SYSTEM CALL AUDITING AT SCALE IN A LARGE CONTAINER DEPLOYMENT

Ravi Honnavalli

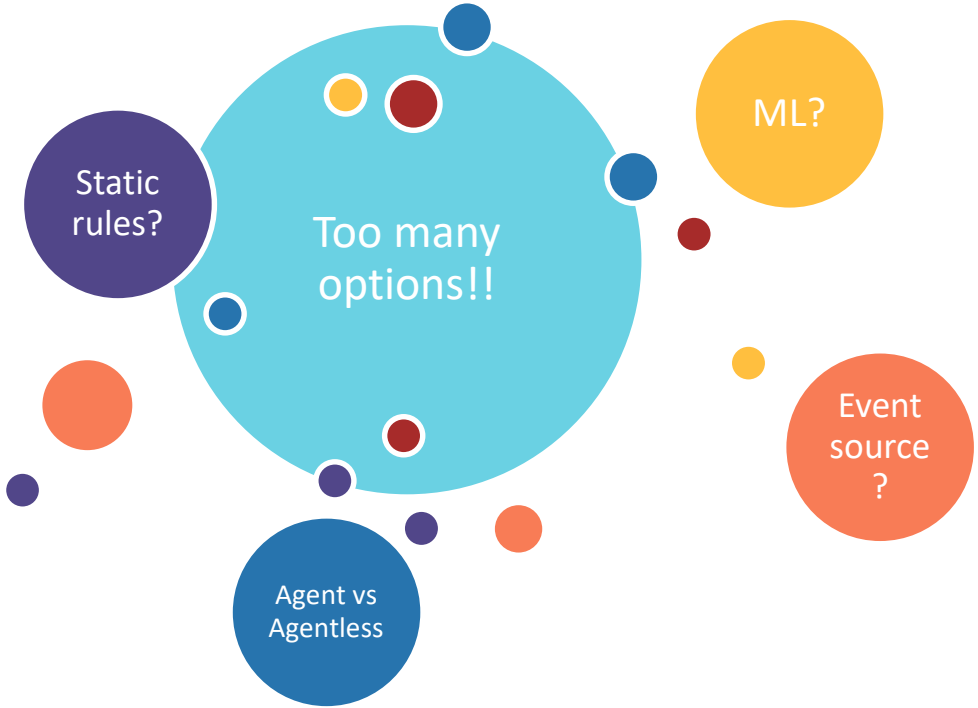
Staff Engineer
Walmart
Twitter handle: @ravi_honnavalli

Disclaimer

NOTE: All content discussed here are out of self learning and not related to the work I do at Walmart.

Ever increasing amount of logs

Overwhelming amount of choices



Flood of OSS tools

fluentd

Kolide

go-audit

SIEM tools

Auditd

Doorman

Elasticsearch

Logstash

Beats

Zentral

osquery

Grafana

Kibana

journald

ElastAlert

syslog

Unstructured datastores

TensorFlow

GOAL: Demystifying the choices we have

Understanding types of event sources	•Classifying event sources •Understand event source type •Evaluate open source stacks
Build our own stack based on insight needed	•Understand the insights we are looking for •Build a stack based on the event classification •If needed customize existing open source tools •Build adaptors / tools that join the whole chain
Make an informed decision	• The stacks discussed in this presentation are by no means the only stack available

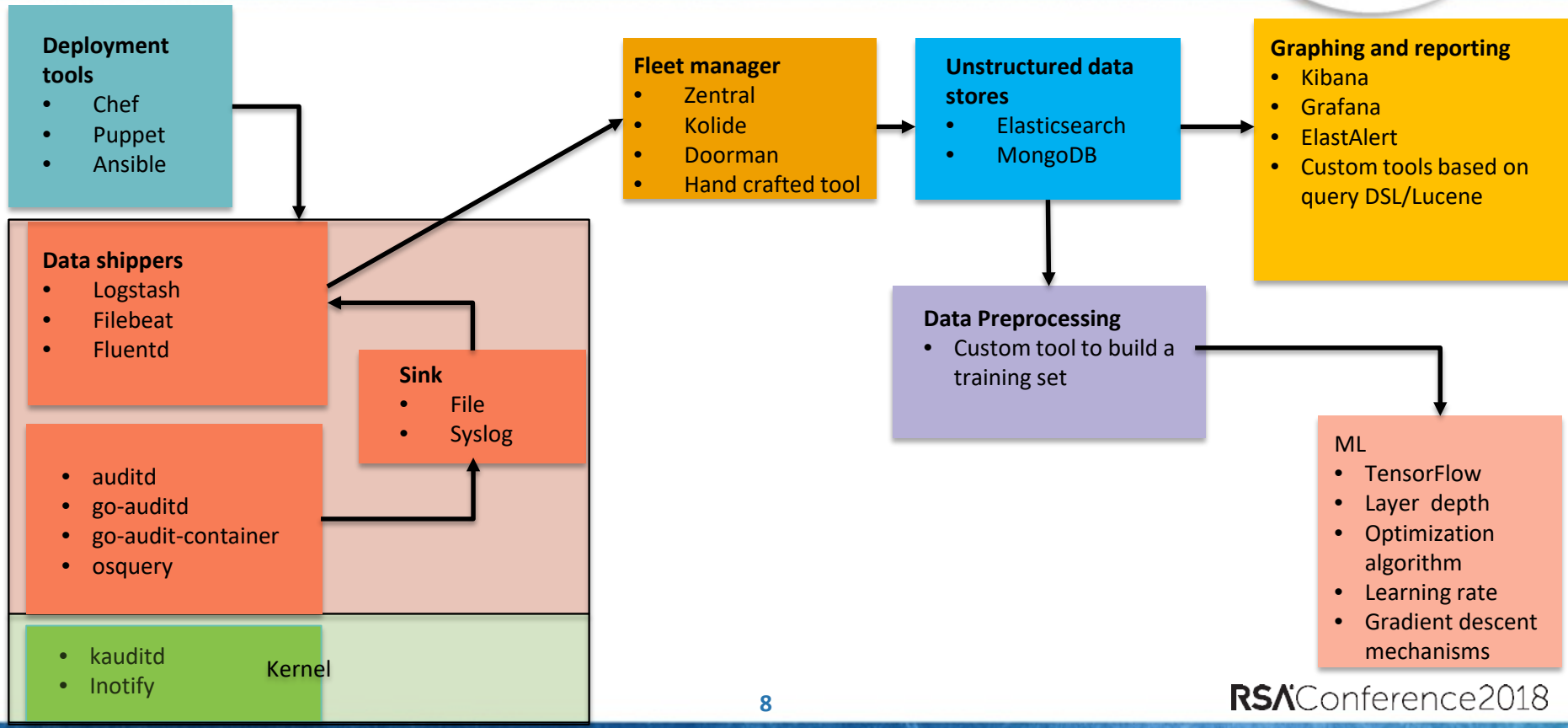
Quick poll

Use audit logging to detect anomalies?

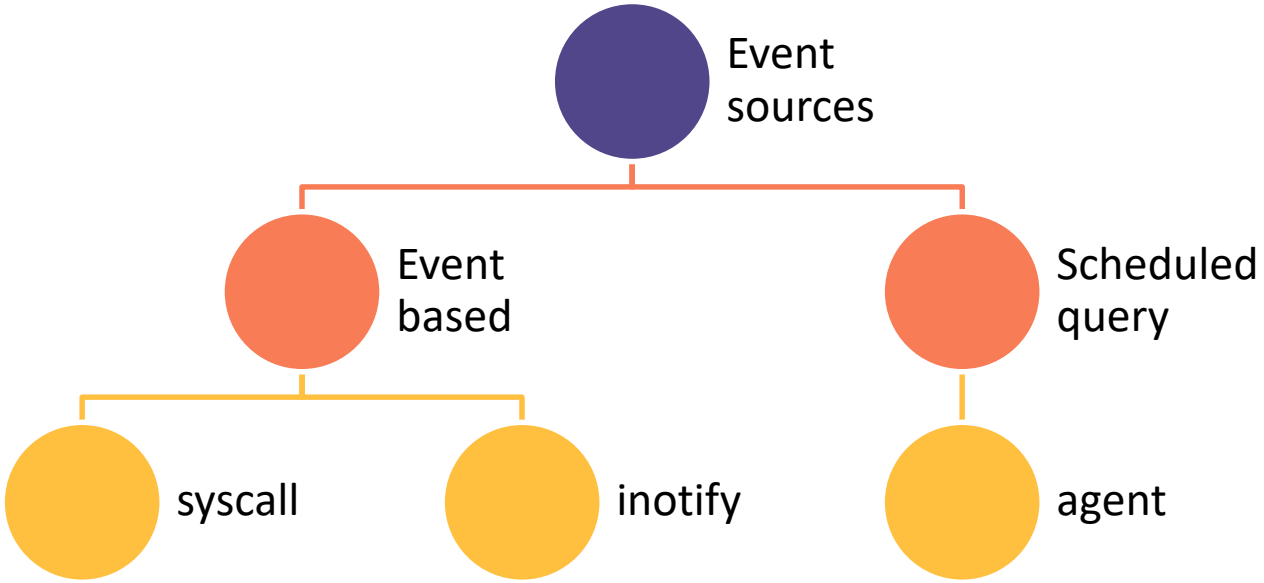
Take it further to use machine learning techniques?

How may implement it only to meet compliance?

Possibilities of tools evaluated



Classification of event sources



Security insight based on event source type

syscall

Looking for specific outliers among mostly normal dynamic events.

- Like identifying outliers
- Monitoring constantly for a specific malicious system call along with other criteria (uid, etc)

inotify

Safe-guarding specific sensitive files / area in the file system

- Watch for CREATE/ACCESS/MODIFY/ DELETE events on specific files

agent

Scheduled activities for static information

- OS patch level queries, vulnerable kernel modules, mis-configuration

SYSTEM CALLS

Why syscall?

Fundamental transit points between user land and kernel

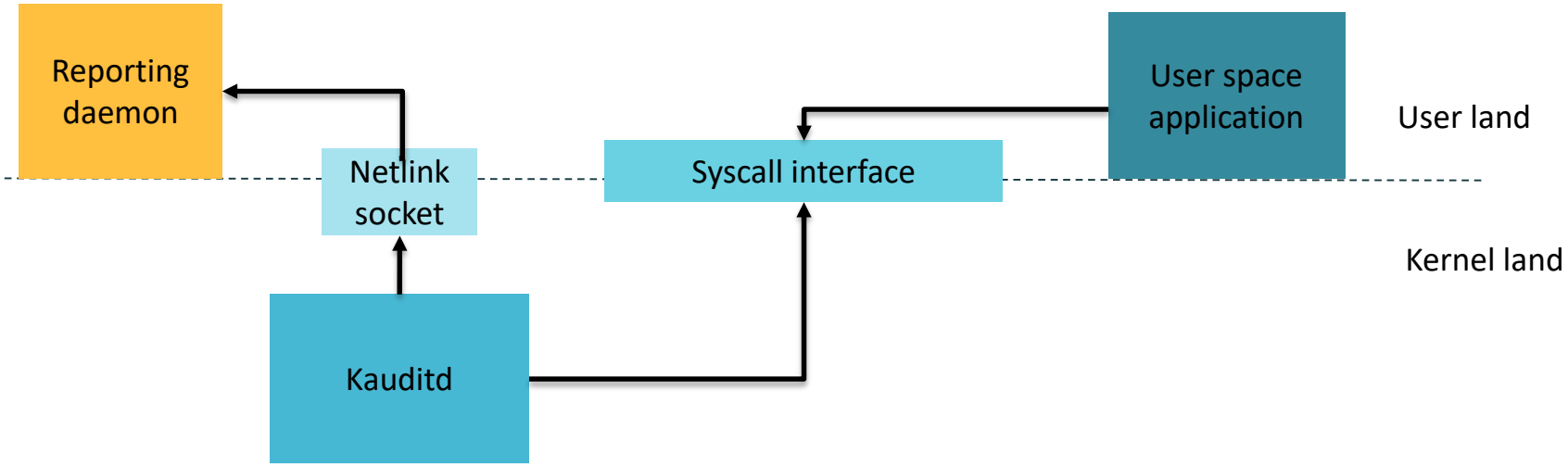
Every process makes system calls disclosing information of its activity

Several user space tools that send audit information (auditd, go-audit, go-audit-container)

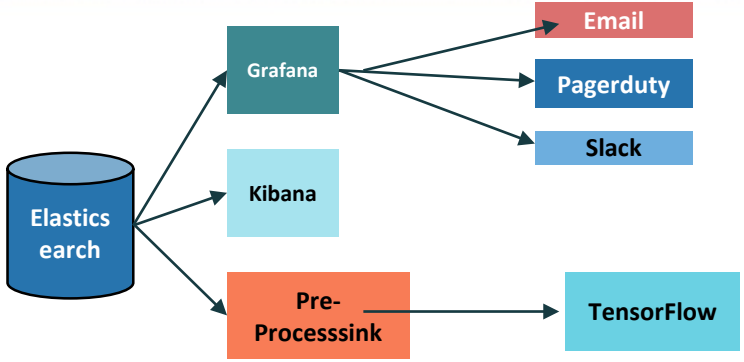
Can provide deep insight when aggregated and drilled down

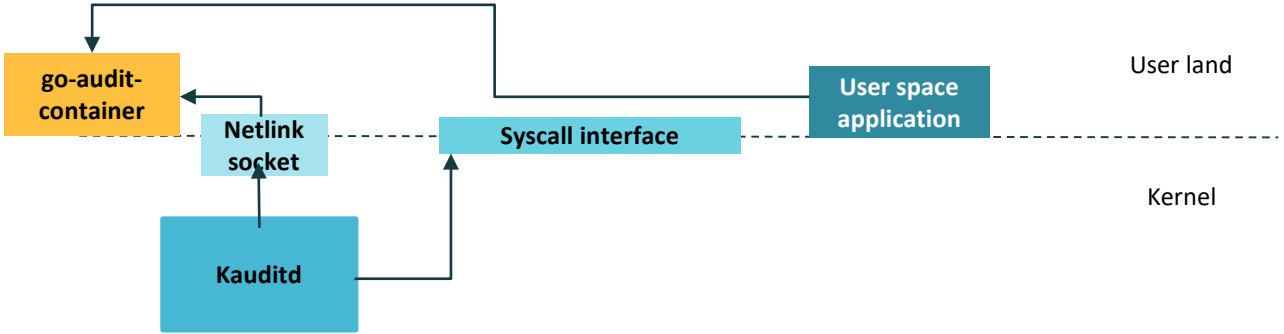
Ideal candidate to build a machine learning training set as the volume of data is huge

Audit component



Audit log to gain insights at scale

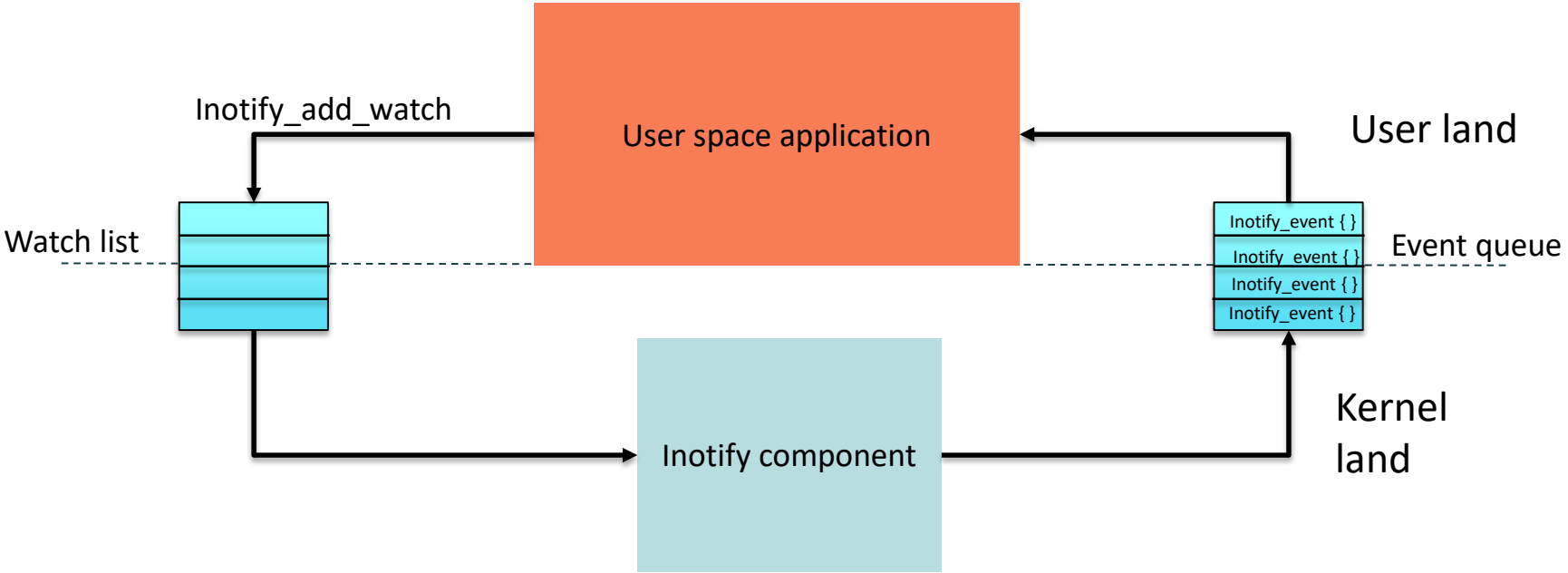




Demo

INOTIFY

Inotify component



Why inotify?

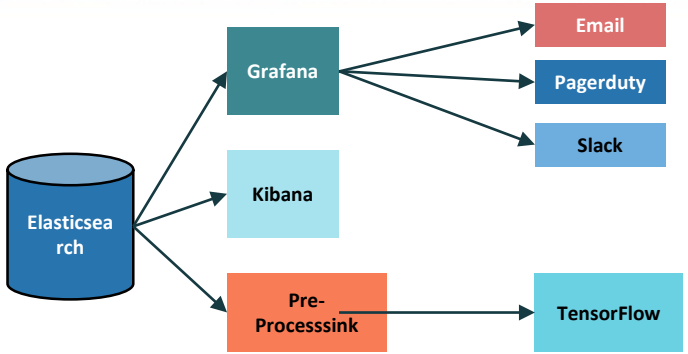
Lesser CPU consumption on an average

Another parallel stack, a new and exciting stack to explore with osquery

Missing details in the reports

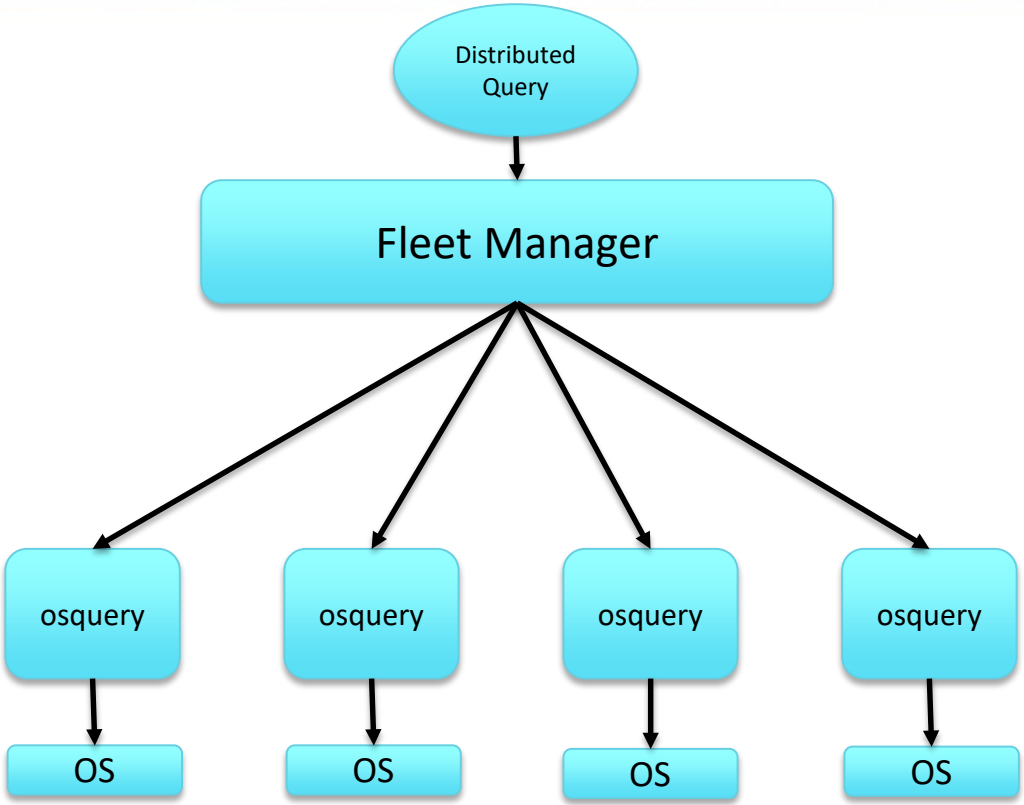
Demo

inotify based stack for FIM

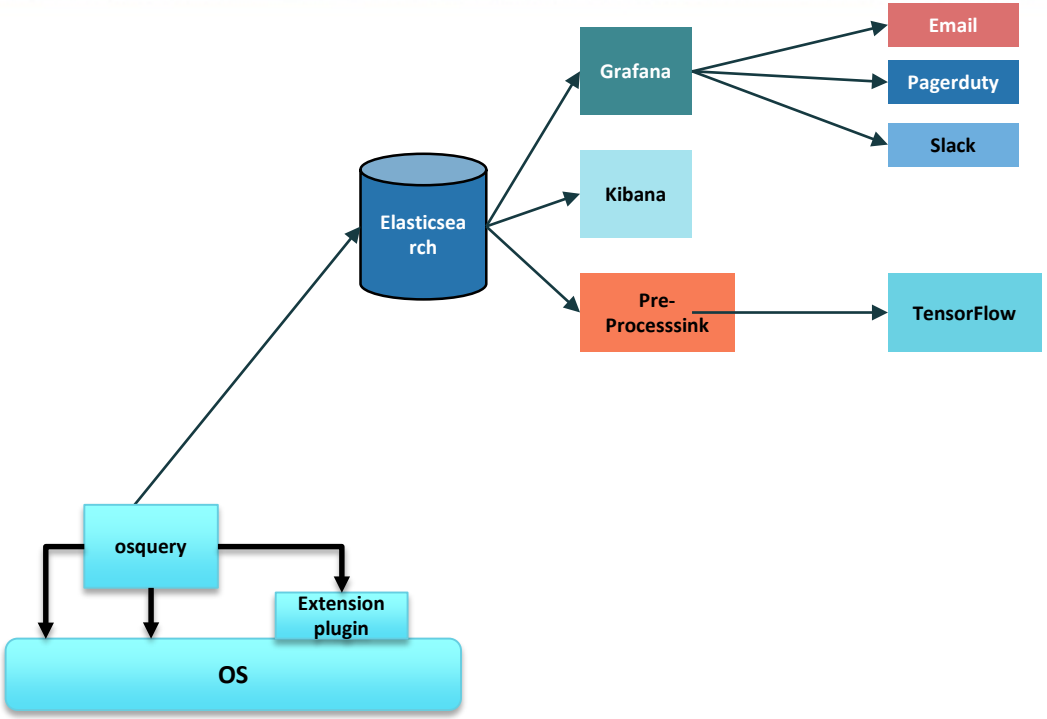


AGENTS

osquery


Distributed Query
Fleet Manager
osquery
osquery
osquery
osquery
OS
OS
OS
OS

Osquery stack to get insights at scale



Demo of tracing a ‘Dirty COW’ exploit

Learning from using fixed queries in Kibana, Grafana and custom tools

Fixed queries are good but only goes so far

Robust rules need a lot of queries

Any small variation of the rules is a false negative

Using DNN based machine learning helps improve our ability to detect anomalies

MACHINE LEARNING

High level differences

Unsupervised

- Elasticsearch ML
- Anomaly detection
- Time series data

Supervised

- Pre-processink-osquery
- Explicit labelling and pre-processing
- Explicit data classification on disparate info

Elasticsearch ML: Detecting outliers

Picture credit: https://unsplash.com/@ripato

Demo of ElasticSearch ML

Use case: Classifying disparate data

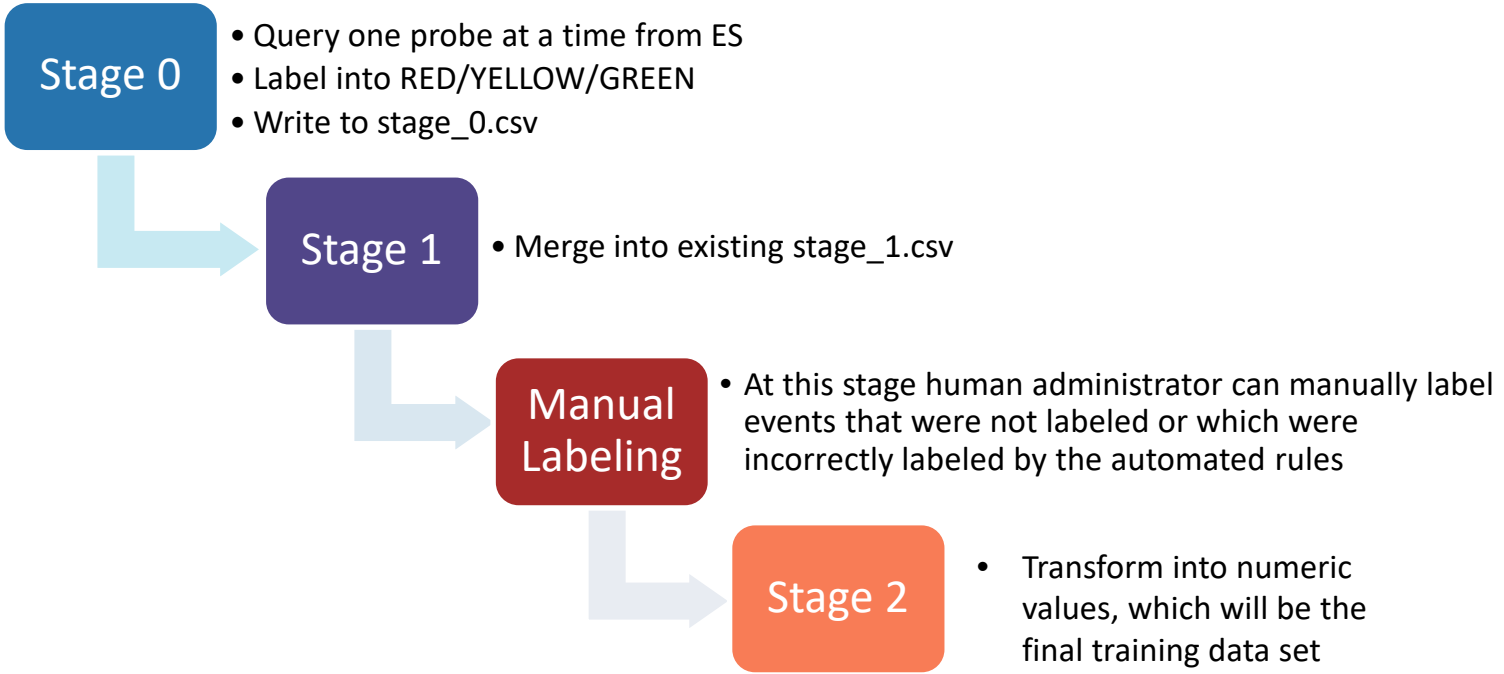
Classifying data from different event sources

Broad classification into RED/YELLOW/GREEN

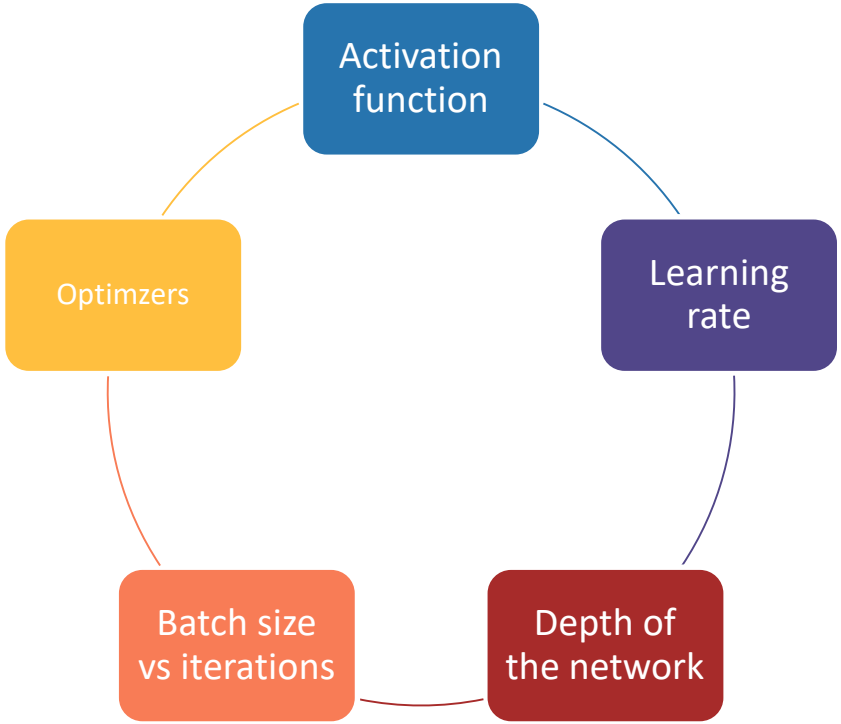
Classifying to have a big picture of the security posture of the organization

Supervised learning: Building a training set is key

Pre-processink-osquery stages



ML choices if you are building your own solution



Results of our experiment

ReLU for hidden layers and softmax for activation

Epoch vs Batch Size vs Iterations

Adam optimizer

Lower the learning rate the better

Lessons learnt

With system call auditing, inotify, os level querying agents as foundations, combined with ability to aggregate at scale in Elasticsearch, we can achieve very deep security insights on the production environment.

With anomaly detection discussed in this talk, we are just scratching the surface. Given that system calls are fundamental, possibilities are enormous.

With static threshold and reporting configuration it is very easy to miss security insights

DNN based machine learning helps in getting intelligent insights

Based on parameters like platform support, CPU utilization, memory and disk space foot print, etc we looked at different choices of stack

Apply

Start with a simple File Integrity monitoring implementation using audit log. Observe load of FIM events on the infrastructure

Grow the solution to more detailed monitoring

Try applying ML based on the rule based and manual labels

Think of possibilities outside of what is discussed here today

AuditNG suite

https://github.com/auditNG/preprocessink-osquery

https://github.com/auditNG/go-audit-container

Questions?

You can also reach out later:

Twitter handle: @ravi_Honnavalli

LinkedIn:
https://www.linkedin.com/in/ravi-honnavalli-0535163/