Threat detection on Kubernetes using GNN embeddings

CAMLIS 2023

Arjun Chakraborty
Staff Engineer, Databricks
About me

Security ML
And
Threat detection
(Twitter/X: @sfrosagulla)

©2022 Databricks Inc. — All rights reserved
Agenda

• Problem statement
• A primer on Kubernetes (K8s) logs
• K8s logs as graphs: The how and the why
• Using GNNs to build out embeddings: A walk through
• Threat hunting with GNN embeddings
• Challenges and looking ahead
• QnA
Problem statement

Static detection rules need constant tuning

Low Signal to noise ratio when detection rule complexity increases

Static detection rules do not have structure or context encoded

No labeled data to build out a supervised learning problem
Monitoring the Kubernetes (K8s) API server

- The core of the Kubernetes control plane is the API server.
- The API server exposes an HTTP API that lets end users and different parts of the cluster, and external components communicate with each other.
- Kubernetes API server generates audit logs that are used for security monitoring.

- What user/system is generating what traffic?
- Which users access production frequently and why?
- What requests are being rejected and why?
- How do we monitor for malicious activity within the Kubernetes cluster?

Source: https://medium.com/pareture/monitor-kubernetes-api-server-audit-in-eks-3e8e6e18e7fb

©2022 Databricks Inc. — All rights reserved
A primer on K8s audit logs

**Request information**
- Originating source of request
- User or system
- Action/Verb type (GET/POST/DELETE/PATCH)
- Requesting resource and subresource

**Response information**
- Outcome of the request
- HTTP status code
- Returned object or details associated with resource

**Metadata components**
- Timestamp
- Audit ID
- Annotations or additional information provided by auditor

**Logging stages**
- Request Received
- Response Started
- Response Completed

Source: https://medium.com/pareture/monitor-kubernetes-api-server-audit-in-eks-3e8e6e18e7fb
What is a K8s session?

K8s API server audit logs details action on a resource

Dependency between each of the components associated with an action/verb (such as GET/PATCH/DELETE)

Inter-dependencies between different actions based on IPs, user groups and resources.

Aggregate multiple such records to create a defined K8s session
Using GNNs to generate embeddings

- Generate embeddings based on classifying graphs into different verb types
- This is a multi label graph classification problem
- Embeddings generated can be used for multiple downstream tasks associated with anomaly detection.
- Size of embeddings can vary depends on the dataset and some level of experimentation
Graph formulation for K8s audit logs

**Graph definition**
- \( G = (V,E,X,Y) \)
- \( V \) = Set of nodes in the graph
- \( X \) = Node feature matrix
- \( E \) = adjacency/connecting matrix
- \( Y \) = labels

**Set of nodes**
- IP
- User group
- Request object
- Resource
- Decision

**Feature matrix and labels**
- 40 dim feature matrix
- Feature hashing based on string
- \( Y \) = Type of verb
- Classification task

**Aggregation**
- Aggregating K8 audit logs based on 1 minute time intervals to define a “session”
- Each session to be considered its own graph.
- Each session to be associated with verb
Model type and architecture

A GCN is a specific instance of a graph neural network that modifies the idea of a CNN to work with graph data and generate node embeddings.

Model specifications

- 3 layer GCN with a linear layer attached at the end.
- Node embeddings are passed through a global_mean_pool to generate graph embeddings.
- The input is associated with the node features.

Source: https://towardsdatascience.com/understanding-graph-convolutional-networks-for-node-classification-a2bfdb7aba7b
TSNE based distribution of embeddings

BEFORE training

Training data size: 600K graphs
Time period: 2 months
Average number of nodes: 206

AFTER training
Model results

No of evaluation samples : 9392
F1-score : 0.804

No of evaluation samples : 9911
F1-score : 0.840

Two different evaluation samples from different regions
Threat detection with embeddings

From the POV of an analyst: What can we actually detect?

- Lot of the MITRE use cases listed above can be detected by writing **precise heuristic based detection rules**.

- Some common use cases:
  - Listing of K8s secrets
  - Kubernetes cronjob
  - Kubecfg file

- Embedding based models in this case are “umbrella” models.

- Best way to use these models is by correlating their results with heuristic based detection rules.

- Infrastructure is prone to malicious attacks. But it’s also prone to bad hygiene → expect benign true positives.

source: https://attack.mitre.org/matrices/enterprise/containers/
Using embeddings downstream

- Trained an isolation forest on the trained model embeddings with a low contamination factor (~ 0.001 - 0.006)
- Created an evaluation dataset of 420 graphs to see if we can find unusual activity patterns.
- Use trained isolation forest on evaluation set embeddings and extract anomalies

How do we evaluate if anomalies these are legitimate?

- Unusual number of nodes
- Set of source IPs associated with the K8s session
- Namespaces associated with the K8s session
- Usernames associated with the K8s session

3 data points that are considered anomalous in this case
After some digging...

Why were the anomalous points anomalous?

<table>
<thead>
<tr>
<th>Threat hunt</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>What was the action/verb associated with the anomalous graph?</td>
<td>● The action/verb was “CREATE”</td>
</tr>
<tr>
<td>What was the size of the anomalous graphs?</td>
<td>● The size of the anomalous graphs were the lowest in that K8s session for the verb “CREATE”</td>
</tr>
<tr>
<td>Were there any new namespaces associated with the anomalous graph?</td>
<td>● There was one new namespace associated with the anomalous graph</td>
</tr>
<tr>
<td></td>
<td>● The namespace was associated with a new container hardening process introduced by engineering</td>
</tr>
<tr>
<td>Were there any new usernames associated with the anomalous graph?</td>
<td>● New username associated with container hardening process</td>
</tr>
<tr>
<td>Were there any new IPs associated with the anomalous graph?</td>
<td>● New IPs associated with the hardening process</td>
</tr>
</tbody>
</table>

Anomalous process  ➔  Benign True positive
Challenges and looking ahead

Challenges

• Interpretability
• An anomalous event is not always malicious

Looking ahead

• Threat hunting will continue to remain an important part of detection
• Specific detection rules + Models still the way to go for threat detection
• Red teaming to make sure adversary simulation is realistic
• Work with your IR/SOC team to tune models at a consistent cadence.
Shoutout to the Databricks Detection and Response teams!

Questions?
References/Acknowledgements

- https://attack.mitre.org/matrices/enterprise/containers/
- https://medium.com/stanford-cs224w/task/home
- https://medium.com/pareture/monitor-kubernetes-api-server-audit-in-eks-3e8e6e18e7fb
- https://mlabonnie.github.io/blog/posts/2022_02_20_Graph_Convolution_Network.html
- https://docs.lacework.net/onboarding/kubernetes-audit-logs-overview
- https://docs.aws.amazon.com/eks/latest/userguide/control-plane-logs.html