EdgeLake - KubeArmor Integration

Overview

KubeArmor analyzes telemetry data to understand application behavior for container/node forensics. With thousands of nodes deployed (using Open Horizon), sending events streams to a centralized node is not a viable option.

To address the need, Open Horizon functionality was extended by deploying on each serviced edge node an EdgeLake agent. The agent pulls the telemetry data from KubeArmor and hosts it locally (EdgeLake appears on each edge node as a local service).

The EdgeLake instances form a decentralized network of nodes that service the distributed edge data as a unified collection of data (whereas the physical data remains distributed at the edge).

With this setup, KubeArmor users and applications are able to query the distributed data. This approach distributes each query to the edge nodes with relevant data and aggregates the individual replies to form a unified and complete result set equivalent to a reply from a cloud based database. A more detailed information on how EdgeLake Operates is available with this link - Value Proposition.

Users deploying EdgeLake to manage the KubeArmor's event data are able to extract real time insight from their data, enable real-time alerts and monitoring and service the data to analysis and AI applications, all of that without cloud contracts and costs.

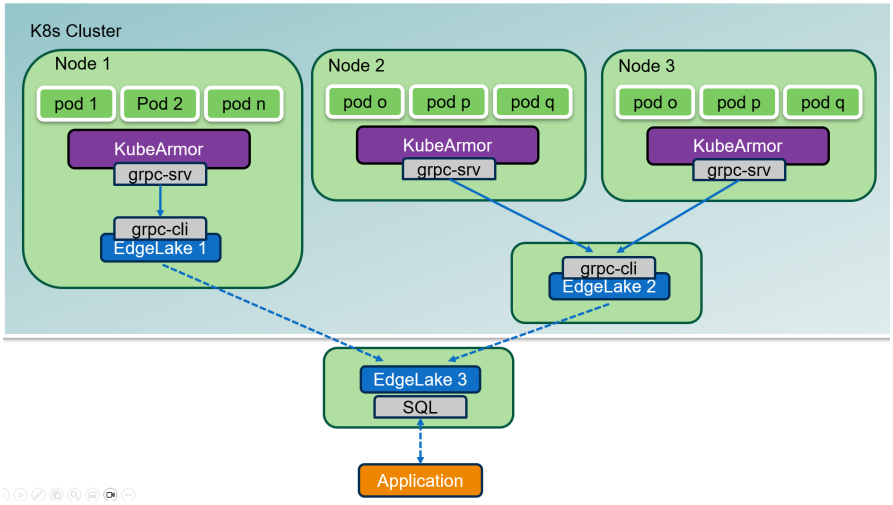
Deployment Architecture

Each EdgeLake is instance is configured to pull data from one or more KubeArmor instances. Users have full discretion where to deploy the EdgeLake instances - on the same physical node as KubeArmor or at a remote node.

EdgeLake can be deployed as an independent instance or as a background process on a machine shared with other instances (physical or virtual).

To host the KubeArmor data, EdgeLake is using a gRPC client service connector (details are available here) to pull the data and host it locally on the EdgeLake node. As each EdgeLake node is a member of the EdgeLake Network, the distributed data is available through the EdgeLake Network services as if the data is centralized.

The overall architecture is shown in the diagram below:



Using this architecture two EdgeLake instances are hosting the KubeArmor event data: EdgeLake 1 is deployed on the same node with KubeArmor. It pulls the KubeArmor event data and hosts it locally.

EdgeLake 2 is deployed on a dedicated node and is pulling data from two KubeArmor instances. EdgeLake 3 is configured to service SQL requests from applications that query the data.

In the same way that applications interact with a relational database, EdgeLake 3 presents to the applications a list of databases, a list of tables for each database and a list of columns per each table.

Using this metadata, applications and users formulate and issue a query to EdgeLake 3. Using the EdgeLake Network protocol and a shared metadata layer (the shared metadata is transparent to the applications and not shown in the diagram), EdgeLake 3 will identify the EdgeLake nodes that host the relevant data (in this example EdgeLake 1, or EdgeLake 2, or both), deliver the query to the participating nodes, and unify the results returned from all the participating nodes.

This process allows to return a complete reply to the application (as if the KubeArmor event data is centralized).

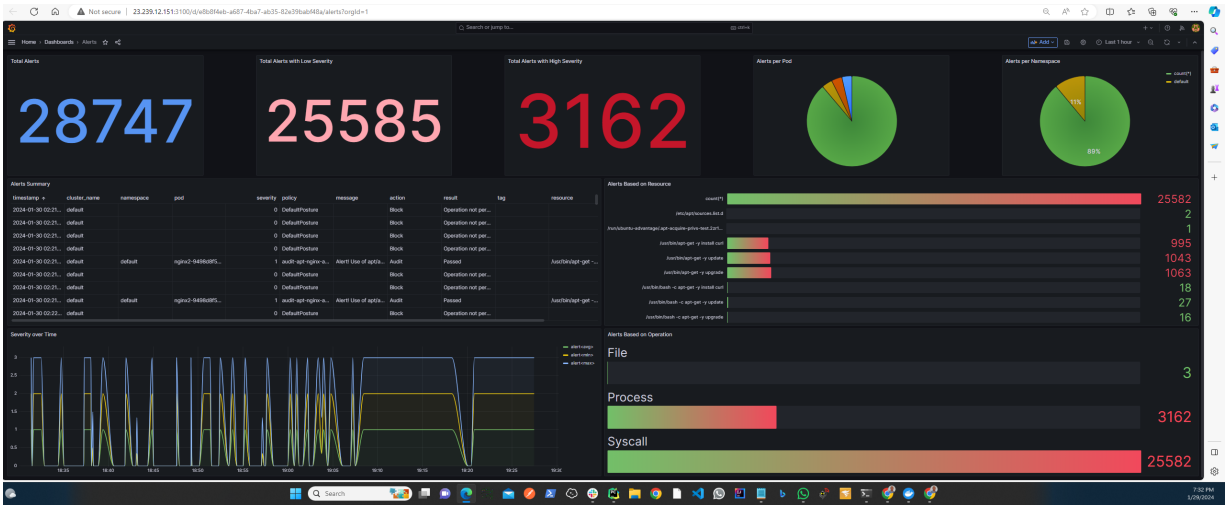
- This process is efficient as only queries and result sets are being transferred over the network and the core data remains in place.

- This process is cost-effective as it is not using cloud services.

Servicing the KubeArmor data to applications

Using this approach, applications connect to a single node in the EdgeLake Network, without the need to know which are the nodes that host the needed data. However, the queries are distributed transparently to the nodes that host the data that needs to be considered, and a complete result set is returned to the query process.

The following Dashboard represents a result set returned to a query that was issued to an EdgeLake Network that services multiple KubeArmor instances (the dashboard is using Grafana, but users can leverage PowerBI or any other tool of their choice).



Deployment Options

EdgeLake nodes can be configured to satisfy different deployment and setup requirements. Some of the commonly used options that are used to support KubeArmor event data are listed below:

- The number of EdgeLake instances in a network.
- The number of KubeArmor instances that are supported by each EdgeLake node.
- The volumes of data to keep on each EdgeLake node.
- Data Archive options.
- High Availability Options.
- Rules on each of the deployed nodes.
- Monitoring active KubeArmor Processes
- Alerts based on data and resource status.
- Policies to identify and tag nodes and events. The tagging can be leveraged in the query and monitoring processes.