corelight

Kubernetes :: Untapped

Vijit Nair & Stan Kiefer

Ninth EU MITRE ATT&CK® Community Workshop

2 June 2022

K8s security monitoring needs network visibility

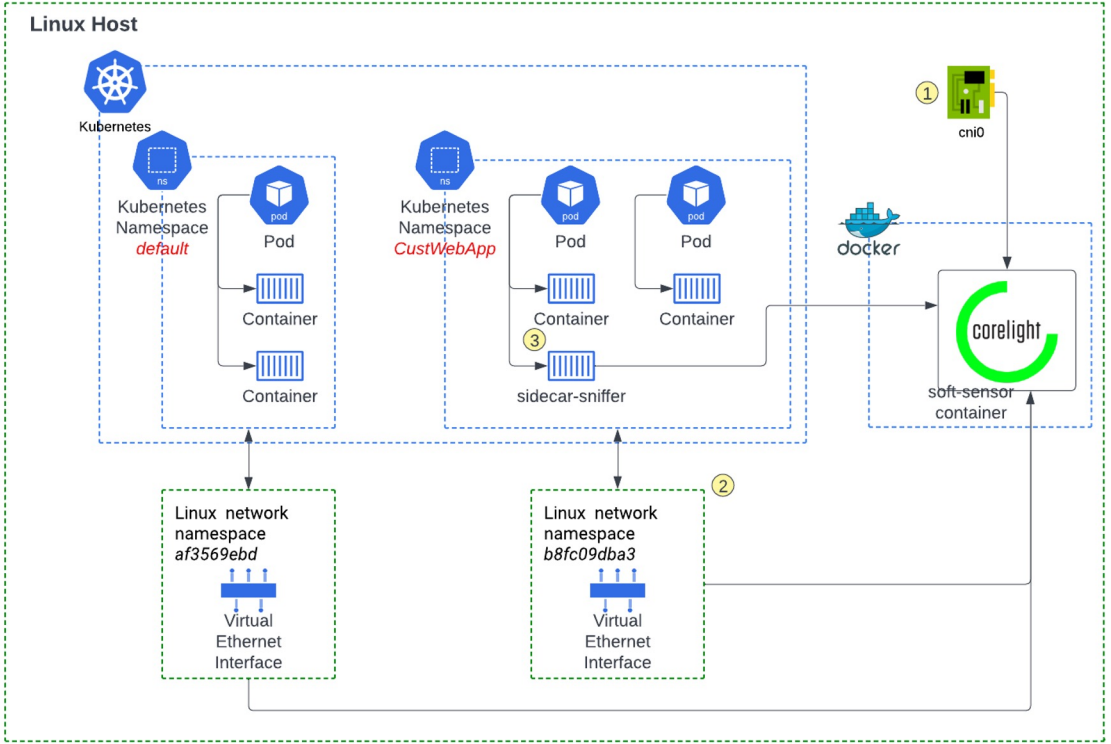
Traditional tools for K8s security visibility (host-based agents and application log monitoring) leave several blind spots

- Lack of visibility into interaction between containers
- Ineffective in detecting cross-account escape scenarios
- Black-box detections are noisy without actionable insights

Network Monitoring in Kubernetes environments

Acquiring traffic

1. CNI Native traffic mirroring
2. Sidecar per container
3. Sensor-agent per host
4. Kubernetes Plugin



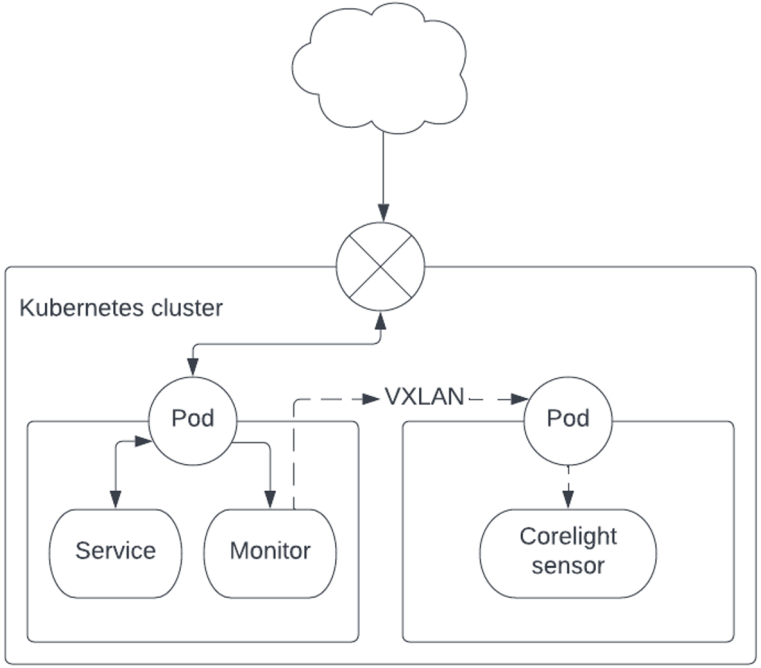
corelight

Network Monitoring in Kubernetes environments

	CNI native traffic mirroring	Sidecar per container	Sensor-agent per host	Kubernetes plugin
How it works	Native support for mirroring traffic	Selectively inject sidecar on containers to sniff / tunnel traffic	Agent sniffs and analyzes traffic per pod/namespace	Custom K8s plugin to sniff / tunnel traffic
Pros	• Native support is scalable, requires limited support / maintenance	• Lightweight sidecar / no tax on host • Can monitor container traffic	• Per-host policy controlled by security team (not part of DevOps process)	• High flexibility
Cons	• Limited platforms support traffic mirroring	• Need to be built into the DevOps process • Need to encrypt traffic if it leaves the host	• Heavyweight - traffic determines host resources needed • Can only monitor pod/pod traffic	• Plugins are DIY / OS tooling • Requires custom K8s implementation, admin privileges

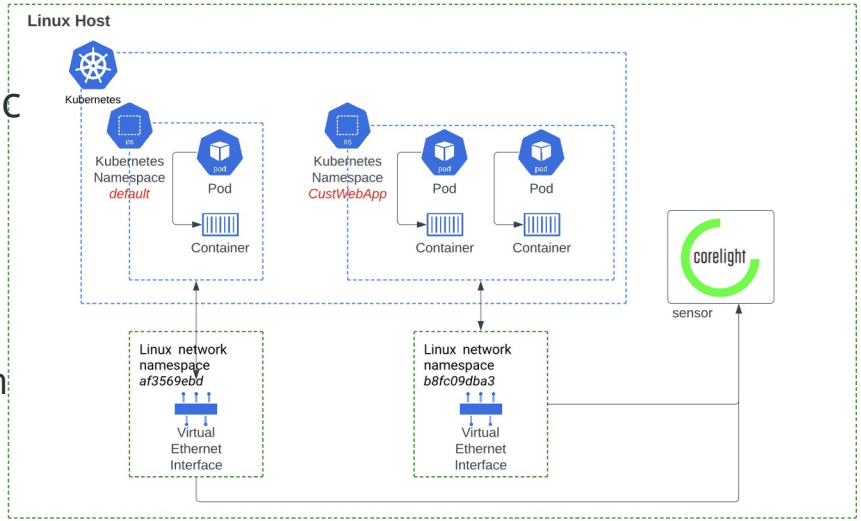
Sidecar per container

- Inject sidecar per container to sniff traffic
- VxLAN encap + tunnel to a destination (on the host or a central collection)
- Pros:
  - Lightweight sidecar / no tax on host
  - Can monitor container traffic
- Cons:
  - Need to be built into the DevOps process
  - Need to encrypt traffic if it leaves the host



Sensor-agent per host

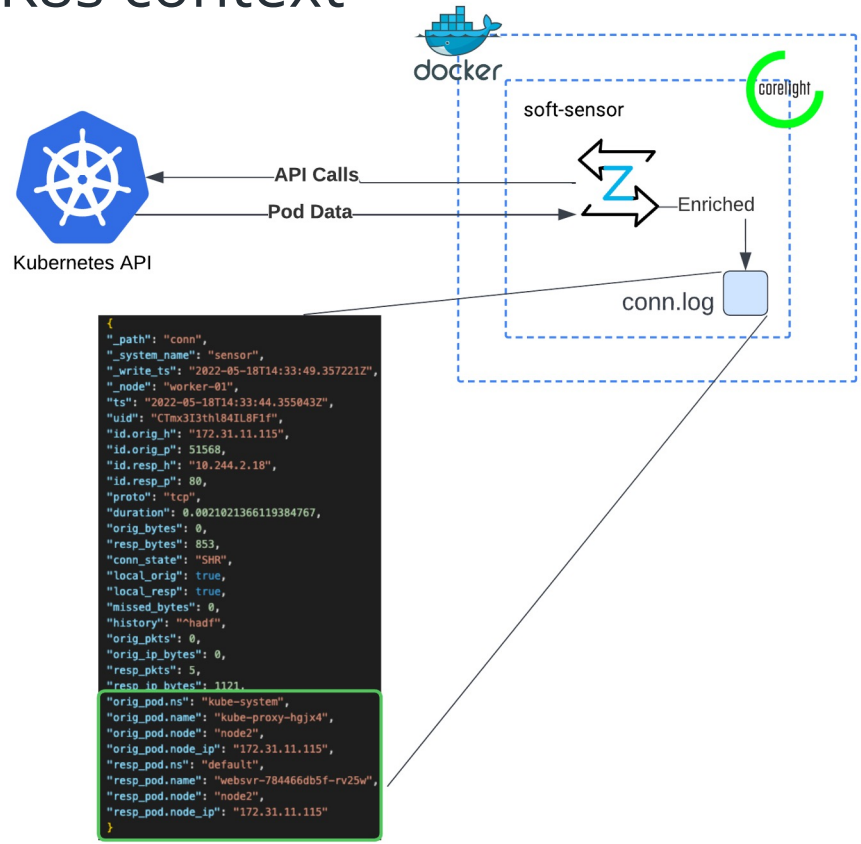
- Host agent to tap veth per namespace
- Agents can mirror or mirror+analyze traffic
  - Sensor-agent: Software sensor to mirror + analyze
  - Mirror-agent: e.g. gigamon, cpacket to mirror only
- Pros:
  - Per-host policy controlled by security team (not part of DevOps process)
- Cons:
  - Heavy - volume of traffic determines resources needed for the sensor-agent
  - Can only monitor pod/pod traffic



corelight

Enrichment of Zeek Logs with K8s context

- Queries K8s API to get all pods, then populates zeek record/table with IP to pod detail mapping
- Any Pod details available from the API could be added to conn.log
- Zeek Script uses Input Framework or ZeekJS plugin



Detecting Log4j in Kubernetes environment


```
{
"_path": "notice",
"_system_name": "sensor",
"_write_ts": "2022-03-08T18:54:18.846539Z",
"_node": "worker-01",
"ts": "2022-03-08T18:54:18.846539Z",
"uid": "CURtov2gIkEQCel897",
"id.orig_h": "10.244.1.23",
"id.orig_p": 54700,
"id.resp_h": "10.244.1.24",
"id.resp_p": 1389,
"proto": "tcp",
"note": "CVE_2021_44228::LOG4J_LDAP_JAVA",
"msg": "Possible Log4j exploit CVE-2021-44228 exploit, JAVA over LDAP. Refer to sub field for sample of payload.",
"sub": "0\\x81\\x8e\\x02\\x01\\x02d\\x81\\x88\\x04\\x01a0\\x81\\x820\\x16\\x04\rjavaClassName1\\x05\\x04\\x03foo0(\
"src": "10.244.1.23",
"dst": "10.244.1.24",
"p": 1389,
"peer_descr": "worker-01",
"actions": [
    "Notice::ACTION_LOG"
],
"suppress_for": 3600.0
```


notice.log indicating LOG4J detection


```
{
"_path": "conn",
"_system_name": "sensor",
"_write_ts": "2022-03-08T18:54:18.846012Z",
"_node": "worker-01",
"ts": "2022-03-08T18:54:18.846012Z",
"uid": "CURtov2gIkEQCel897",
"id.orig_h": "10.244.1.23",
"id.orig_p": 54700,
"id.resp_h": "10.244.1.24",
"id.resp_p": 1389,
"proto": "tcp",
"conn_state": "OTH",
"local_orig": true,
"local_resp": true,
"missed_bytes": 0,
"history": "C",
"orig_pkts": 12,
"orig_ip_bytes": 510,
"resp_pkts": 5,
"resp_ip_bytes": 312,
"orig_pod.ns": "default",
"orig_pod.name": "websvr2-58d56cf5-2whrf",
"orig_pod.node": "node2",
"orig_pod.node_ip": "172.31.11.115",
"resp_pod.ns": "default",
"resp_pod.name": "ldap-784466db5f-rv25w",
"resp_pod.node": "node2",
"resp_pod.node_ip": "172.31.11.115"
}
```


Conn.log enriched with K8s context

Other Considerations

- Pod-pod & container-container traffic will have high volume; how to balance optimal visibility against heavy log volume?
- Engineered visibility - some container traffic flows are much more interesting than others; how to surgically select flows to mirror?
- Processing of traffic - Centralized on a separate host vs distributed on each host

QUESTIONS?

corelight

NETWORK DETECTION & RESPONSE | ON PREM AND CLOUD