Simulate a Live Attack for Kubernetes

The Wiz Runtime Sensor collects runtime events on Kubernetes-cluster workloads and translates them to detections. Use this benign attack scenario to generate detection events simulating events that will be triggered in case of a similar attack.

> This scenario uses a benign script for the simulation and should be safe to execute in your environment. However, we highly recommend you run it only in test or demo environments, and that you do not run it on your production environment.

Step 1: Deploy and run the attack scenario

Step 2: Evaluate Runtime Sensor detections generated during the attack

Prerequisites

- Kubernetes version 1.20 or higher, where the Runtime Sensor was successfully deployed on a cluster.

> Only x86_64 architecture is supported. If you can't provide x86_64 nodes, use the Sensor Evaluation (Lite Version) instead.

- Access credentials with permissions to deploy and delete pods.
- A local installation of `kubectl`.
- Access to the Kubernetes cluster API from your machine with the ability to run `kubectl` commands.
- Wiz Sensor installed on the Kubernetes cluster

Deploy and run the attack scenario

The following scenario simulates an attack that was observed in the wild, where a threat actor known as TeamTNT could gain access to cloud resources to depl

crypto miner. For the purpose of this simulation, we are using a prebuilt bash script, similar to one delivered by TeamTNT.

1. Download the attack-scenario.yaml file.
2. Deploy the container image to simulate initial attacker access by running:

```
kubectl apply -f attack-scenario.yaml
```

3. Initiate the attack scenario and monitor its progress by running:

```
kubectl exec -it wiz-sensor-evaluation-scenario-pod -- bash /root/scenario.sh
```

4. After 5 minutes, kill the running process using `ctrl+c` and remove the evaluation pod and related secret by running:

```
kubectl delete -f attack-scenario.yaml
```

Evaluate Runtime Sensor detections generated during the attack

The Runtime Sensor detections generated by the attack scenario are now represented in the Explorer > Cloud Events page in your Wiz portal. To understand each detection and the associated MITRE techniques, let's review the attack scenario high-level workflow step by step.

Upload payload

Attack stage

The first step in the attack is to connect to the attacker's command & control server in order to download the crypto miner to the `bin` directory of the target resource.

Related detection(s) in Wiz

- Suspected drop and execute - ingress tool with payload decode or file/folder permission change commands were executed
- File created or modified in bin folder
- File in a code library directory was added/modified
- Egress/Ingress tool with a miner related keyword was executed

Prepare stealthy payload execution

Attack stage

The second step is to hide the crypto miner and evade detection. The attacker overwrites `/etc/ld.so.preload` to load a user mode rootkit (using dynamic library injection), ensuring that the soon to be run crypto miner would stay hidden from process listing.

Related detection(s) in Wiz

- Library preload configuration file was added/modified (T1574.006 Hijack Execution Flow: Dynamic Linker Hijacking)

Establish persistence on the resource

Attack stage

To make sure the crypto miner activity is persistent on the resource and continues to run also after the resource restarts, the attacker creates a `cron` job.

Related detection(s) in Wiz

- File/directory permissions on cron job directory was detected (T1053.003 Scheduled Task/Job: Cron)
- Cron job was added/modified (T1053.003 Scheduled Task/Job: Cron)

Secret and credential discovery

Attack stage

To obtain credentials (which will be used later on to move laterally in your environment), the attacker scans the disk image for AWS, SSH, and/or Kubernetes credentials, saves them in a dedicated location on the disk, and uploads them to the command & control server.

Related detection(s) in Wiz

- Kubernetes service account token was accessed (T1552.001 Unsecured Credentials: Credentials In Files)
- Kubernetes service account token was accessed by text reader/search binary (T1552.001 Unsecured Credentials: Credentials In Files)
- Hidden directory was created (T1564.001 Hide Artifacts: Hidden Files and Directories)
- Containerized process created a connection to the IMDS (T1552.005 Uns… Credentials: Cloud Instance Metadata API)

Establish resource foothold for interactive commands

Attack stage

Following the discovery of the secrets, the attacker now initiates a reverse shell on the resource in order to execute different commands for lateral movement attempts (based on the restrictions of the AWS role of the resource explained in the next step).

Related detection(s) in Wiz

- Process executed with non-standard socket input (T1059.004 Command and Scripting Interpreter: Unix Shell)
- Network manipulation/scanning tool was executed

Lateral movement attempt

Attack stage

The attacker starts moving laterally in your environment by creating other instances and expanding the crypto-mining activity using AWS cli commands. Executing `aws iam list-roles` commands are the first step of attached-roles discovery of the stolen AWS credentials.

Related detection(s) in Wiz

- Containerized process created a connection to the IMDS (T1552.005 Unsecured Credentials: Cloud Instance Metadata API)

Execute crypto miner

Attack stage

The last step is executing the crypto miner on the resource.

Related detection(s) in Wiz

- Connection to a known cryptomining domain (T1496 Execution)
- Malware Execution
- Container drift - executable not present in container image was executed (T1496 Resource Hijacking)

Simulate a Live Attack for Kubernetes (lite)

Simulate a Live Attack for Linux