Threat Actors Now Target Docker via Container Escape Features

trendmicro.com/en_us/research/21/b/threat-actors-now-target-docker-via-container-escape-features.html

February 9, 2021

An increasing number of enterprises and organizations have adopted the microservice architecture for its simplicity and flexibility. In fact, a 2019 survey states that 89% of technology leaders believe that microservices are vital for enterprises to remain competitive in an ever-evolving digital world. As more developers deploy containers on-premises and in cloud services, critical data could be inadvertently exposed due to security control failures, making them an interesting target for threat actors. We are seeing ongoing attacks against misconfigured services, such as cryptocurrency-mining malware being deployed in exposed Redis instances and as rogue containers using a community-contributed container image on Docker Hub.

We saw an attack where cryptocurrency-mining malware searched and killed off other existing cryptocurrency miners in infected Linux systems to maximize their own computing power. This attack showcased the malicious actors' familiarity with Docker and Redis, as the malware featured in this attack looked for exposed application programming interfaces (APIs) in these platforms. However, we're currently seeing something completely different — a payload specifically crafted to be able to escape privileged containers with all of the root capabilities of a host machine. It's important to note that being on Docker doesn't automatically mean that a user's containers are all privileged. In fact, the vast majority of Docker users do not use privileged containers. However, this is further proof that using privileged containers without knowing how to properly secure them is a bad idea.

Technical analysis

Case details by Alfredo Oliveira

In July 2019, Felix Wilhelm from the Google Security Team tweeted a proof of concept (PoC) showcasing how "trivial" it would be to break out a privileged Docker container or a Kubernetes pod abusing the *cgroups release_agent* feature.

Our team recently spotted a container abuse attack using the same approach to try to break out of our honeypot environment.

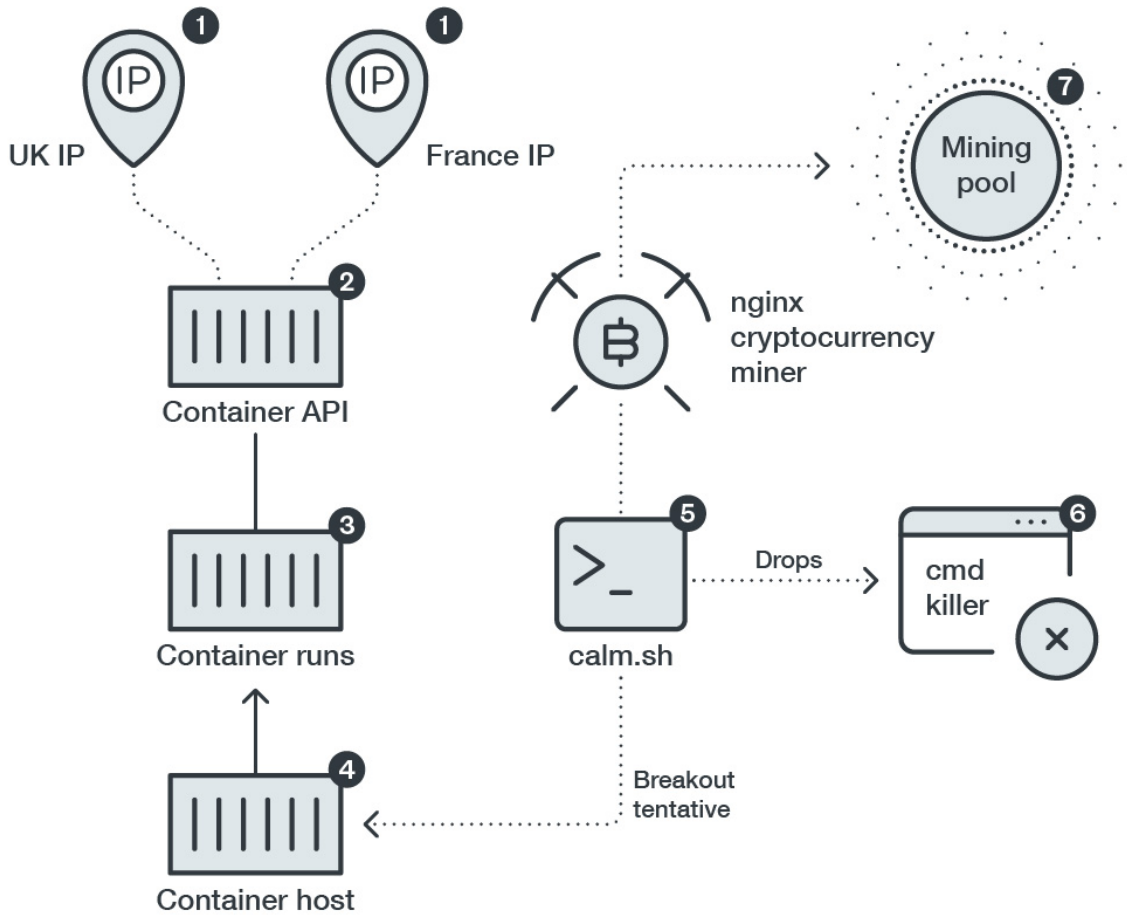

Figure 1. Attack chain that features the abuse of a privileged Docker container

Based on our investigation, the attacker or bot source IP addresses feature addresses from the UK and France. A malicious container running on a Docker-run host is deployed and ran. The attacker's entry point is a shell script called *calm.sh*. *Calm.sh* will then drop another shell script called *cmd*. Calm.sh will call *nginx*, a fake app that is actually an executable and linkable format (ELF) cryptocurrency miner. The attackers possibly used the name *nginx*, which is usually associated with an HTTP server, to fly under the radar and fool victims into thinking that it's legitimate.

Our honeypot was caught by a network scanner that has become very popular among container attacks. The scanner called *zgrab* mapped the container with the exposed API, and we found that the following connection deployed the malicious container in the system.

```
GET /v1.16/version HTTP/1.1
Host:
User-Agent: Mozilla/5.0 zgrab/0.x
Accept: */*
Accept-Encoding: gzip
```

Figure 2. The connection that deployed the malicious container on the exposed API

The API using the command *create* pulled the container image called "gin" from its registry. One of the options used as a deployment parameter was *privileged*, since it's a requirement for this specific escape technique.

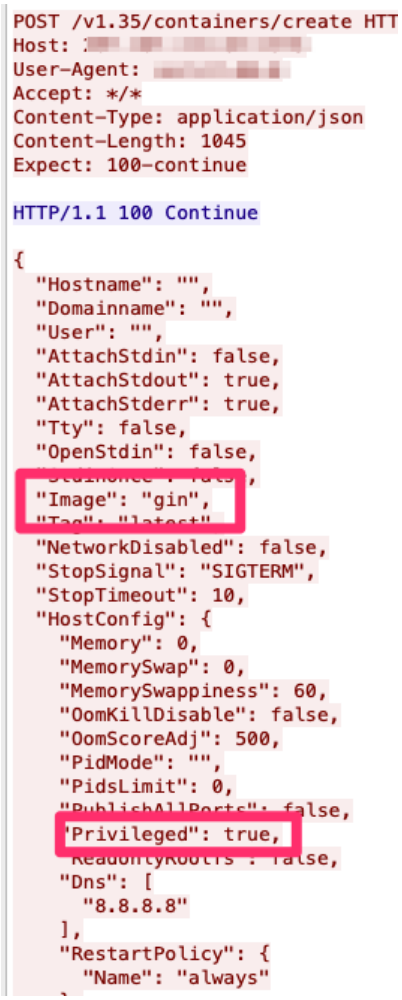

Figure 3. Code that shows the API pulling the container image “gin” and the privileged deployment parameter

In case the exploitation is not successful, we observed that the attackers didn’t miss the opportunity to launch a cryptocurrency miner on a vulnerable server that didn’t need a privileged container to run on.

Figure 4. A cryptocurrency miner running on a vulnerable server

The cryptocurrency miner binary is an ELF file called *nginx*, which is also embedded inside a malicious container image. It attempts to operate stealthily by pretending to be a valid service instead of being a malicious file with the purpose of using all of the infected container’s available resources to mine cryptocurrency — a trend that we have been talking about for a while now.

Figure 5. The nginx ELF file embedded in a malicious container image

Is vulnerability checking enough to secure a container image?

An important process that will help guarantee that a container image is kept secure is to scan it for vulnerabilities — but that shouldn’t be all that should be done to secure containers. Security teams should also regularly scan container images for malware and exploit files.

In this specific attack, the *nginx* binary is a known sample to Trend Micro's threat intelligence and solutions, which is why our anti-malware engine promptly detected it. It is a myth that Linux-based operating systems are immune from threats and risks such as this. As we've recently discussed in another article, malicious actors are zeroing in on Linux as a lucrative target due to the influx of enterprises and organizations moving to the cloud and using Linux in their critical business operations.

Conclusion

Malicious actors are targeting popular DevOps technologies and finding new ways to attack containers and cloud environments.

Most of the earlier samples we analyzed did not implement any security checks. What attackers seemed to have been doing is to try everything and to see what would work; it was a matter of throwing everything at the wall and seeing what would stick.

We're now seeing that malicious actors are becoming more sophisticated with their techniques. They are currently implementing security checks to try to exploit a bad implementation and escape from the container to the host or deploy a cryptocurrency miner and profit from their victims' resources.

Trend Micro solutions

Trend Micro™ Cloud One™is a security services platform for cloud builders. It provides automated protection for cloud migration, cloud-native application development, and cloud operational excellence. It also helps identify and resolve security issues sooner and improves delivery time for DevOps teams. It includes the following:

- Workload Security: runtime protection for workloads
- Container Security: automated container image and registry scanning
- File Storage Security: security for cloud file and object storage services
- Network Security: cloud network layer IPS security
- Application Security: security for serverless functions, APIs, and applications
- Conformity: real-time security for cloud infrastructure — secure, optimize, and comply

Indicators of compromise (IoCs)

SHA-256	File name	Detection name
6f2825856a5ae87face1c68ccb7f56f726073b8639a0897de77da25c8ecbeb19	nginx	Coinminer.Linux.MALXMR.SMDSL64
548236b18ae6c6b588f1180ac70561f20c1bce33b98ef15e385b5c060921b871	calm.sh	Trojan.SH.DOKI.A

Cloud

We provide a technical analysis of a container abuse attack that features a payload that's specifically crafted to be able to escape privileged Docker containers.

By: Alfredo Oliveira, David Fiser February 09, 2021 Read time: (words)

Content added to Folio