



ProxyShell exploitation leads to BlackByte ransomware

The BlackByte ransomware operators leverage ProxyShell Microsoft Exchange vulnerabilities for initial access along with Cobalt Strike for lateral movement. Here's what to look out for.

HARRISON VAN RIPER

Originally published November 30, 2021. Last modified December 2, 2021.

As law enforcement **arrests continue to put a dent in the plague of ransomware**, new variants continue to pop up week after week. BlackByte ransomware was first publicly identified in a July 2021 **BleepingComputer forum post** from a user seeking help decrypting their encrypted files. Since then, there's been a slow trickle of information regarding this new variant, with interest in it peaking in early October when operators **attacked an Iowa grain cooperative**.

Red Canary came across BlackByte working a **short-term incident response engagement** in conjunction with our **partners at Kroll**. Trustwave SpiderLabs **released two excellent blogs** analyzing a BlackByte sample they observed, and we wanted

to take a look at the operation more broadly by analyzing the initial access, post-exploitation, and exfiltration phases prior to BlackByte encryption. While we don't have all of the answers, we wanted to provide the community with our analysis with hopes we can continue the BlackByte discussion.

Initial access

In the campaign we observed, BlackByte operators gained initial access by exploiting the ProxyShell vulnerabilities (CVE-2021-34473, CVE-2021-34523, CVE-2021-31207) present on the customer's Microsoft Exchange server ([T1190 Exploit Public-Facing Application](#)). Widely reported and [acknowledged](#) by Microsoft in August 2021, ProxyShell exploitation [allows an adversary to gain pre-authentication remote code execution](#). Here's a quick primer on the ProxyShell exploitation process that we observed:

1. An adversary remotely created a draft email with an attachment saved in the user's Drafts folder. The attachment contained the encoded web shell.
2. The adversary exported the entire mailbox (malicious draft email included) to an ASPX file, which is a type of personal storage table (PST) file format. The process of writing the ASPX file decodes the encoded web shell.

From an endpoint perspective, we observed the service `MSExchangeMailboxReplication.exe` writing an ASPX file to the folder `\\127.0.0.1\c$\program files\microsoft\exchange server\v15\frontend\httpproxy\owa\auth\current\themes` ([T1505.003 Server Software Component: Web Shell](#)):

Threat occurred

```
Process spawned by services.exe  
c:\program files\microsoft\exchange server\v15\bin\msexchangemailboxreplication.exe  
600b8beccf18b31619faa09646758d19  
3aaa91a4ceb07ba8e2ffb49ee8910409b039193651e4c2e426dac98f00333a4a
```

The following `aspx` file was written to disk:

```
2021-10-07 15:25:00.141 GMT  
filemod  
First wrote to \\127.0.0.1\c$\program files\microsoft\exchange server\v15  
\frontend\httpproxy\owa\auth\current\themes\ukbse.aspx
```

Since this service should not normally write ASPX files, this presents a detection opportunity.

Detection opportunity: **Microsoft Exchange Mailbox Replication service writing Active Server Pages**

```
process == msexchangemailboxreplication.exe
&&
create_filename_extension == .aspx

||

modify_filename_extension == .aspx
```

Though we don't know the adversary's source IP address from log sources, we observed 185.93.6[.]31 making a network connection in a later stage of execution. We assess that it likely conducted the initial ProxyShell exploitation in this incident. Security researcher Kevin Beaumont [observed the same IP address](#) exploiting the ProxyShell vulnerabilities to drop web shells on vulnerable servers in early September 2021, one month prior to this intrusion. We also directly observed several inbound connections to customer Exchange servers, which we assess was likely vulnerability scanning ([T1595.002 Active Scanning: Vulnerability Scanning](#)).

Post-exploitation

Operators used their web shell to drop a **Cobalt Strike** beacon (c84d4ead6c5a2afa9e844806de549dcf) on the compromised Exchange server to allow more functionality directly on the compromised system (**T1105 Ingress Tool Transfer**). Based on the beacon's **configuration file**, Cobalt Strike injected into the `wuauclt.exe` (Windows Update Agent) process. We detected this process launching with no command-line parameters. This is somewhat unusual (and presents a detection opportunity), as `wuauclt.exe` (Windows Update Agent) typically launches with a parameter (**T1055 Process Injection**).

Detection opportunity: **Wuauclt.exe** executing with no command line argument

```
process == wuauclt.exe
```

```
&&
```

```
process_command_line_contains != ""
```

Note: Double quotes ("") within the command line means null.

Operators then used Cobalt Strike to dump credentials for a service account on the compromised system ([T1003 OS Credential Dumping](#)). After gaining access to a service account, the adversaries installed the remote desktop application [AnyDesk](#) to access multiple systems ([T1105 Ingress Tool Transfer](#)). In addition to lateral movement through AnyDesk, the operators created additional Cobalt Strike beacons within the `Admin$` share folders on compromised domain controllers ([T1021.002 Remote Services: SMB/Windows Admin Shares](#)).

BlackByte execution

Cobalt Strike introduced and executed BlackByte (`9344afc63753cd5e2ee0ff9aed43dc56`) in the environment. BlackByte launched with a command-line parameter of `-single` with a SHA-256 hash. We assess this parameter may be a unique identifier for the infected system, but it's still unclear exactly what this parameter does. As a precaution, we have redacted this value from our screenshots. If anyone in the community has insight into this, please email intel@redcanary.com.

Threat occurred

```
Process spawned by svchost.exe  
c:\users\public\complex.exe 9344afc63753cd5e2ee0ff9aed43dc56
```

Command Line: `C:\Users\Public\complex.exe -single [REDACTED]`

BlackByte (which had a file name of `complex.exe`) executing with a SHA-256 command line argument (which we have redacted in this screenshot)

Once executed, BlackByte deletes Task Manager (`taskmgr`) and Resource Monitor (`resmon`), and issues an obfuscated **PowerShell** command to stop the Windows Defender service (`WinDefend`) (**T1562.001 Impair Defenses: Disable or Modify Tools**). This is likely done to avoid detection and keep Windows Defender at bay while BlackByte continues execution. BlackByte also creates a TMP copy of itself, which we assess may have been used during its worming phase.

Process spawned by complex.exe

c:\windows\system32\cmd.exe f5ae03de0ad60f5b17b82f2cd68402fe

Command Line: `cmd /c del C:\Windows\System32\Taskmgr.exe /f /q & del C:\Windows\System32\resmon.exe /f /q & powershell -command "$x = [System.Text.Encoding]::Unicode.GetString([System.Convert]::FromBase64String('Vw'+'BpA'+'G4ARAB'+'LAGYA'+'ZQB'+'uAG'+'QA'));Stop-Service -Name $x;Set-Service -StartupType Disabled $x"`

Decoded Command Base64 Command line: WinDefend

This command will stop Windows Defender from executing on Startup.

Base-64-encoded PowerShell command to disable the WinDefend service

Next, BlackByte injected into the `regedit.exe` process, which executed with the same `-single` command line parameter as the original BlackByte binary. Regedit initiated a connection to the same IP address mentioned previously, `185.93.6[.]31`.

BlackByte was possibly communicating back to the adversary's command and control (C2) server via this Regedit connection, though this remains an intelligence gap.

Preparing to worm

Typically, we would expect Cobalt Strike to be the main driver behind privilege escalation and lateral movement within a compromised environment. However, BlackByte handles both of those on its own. In the sample we observed, BlackByte set three different registry values to escalate privileges and begin setting the stage for lateral movement and encryption (special thanks to [Trustwave's Spiderlabs](#) for filling in this intelligence gap in their breakdown of BlackByte) ([T1112 Modify Registry](#)):

- Elevate local privileges: `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f`
- Enable OS to share network connections between different privilege levels: `HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System /v EnableLinkedConnections /t REG_DWORD /d 1 /f`
- **Enable long path values for file paths, names, and namespaces** to ensure encryption of all file names and paths: `HKLM\SYSTEM\CurrentControlSet\Control\FileSystem /v LongPathsEnabled /t REG_DWORD /d 1 /f`

Detection opportunity: **Privilege escalation and encryption preparation via registry modifications**

```
Process == regedit.exe
&&
Reg_key ==
("HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" || "HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System" || "HKLM\SYSTEM\CurrentControlSet\Control\FileSystem /v LongPathsEnabled")
&&
command_line_contains == (REG_DWORD&& 1)
```

BlackByte then conducted network reconnaissance and system preparation prior to lateral movement within the environment.

1. BlackByte executed two `netsh advfirewall firewall` commands to enable the “Network Discovery” and “File and Printer Sharing” rule groups ([T1562.004 Impair Defenses: Disable or Modify System Firewall](#)).
2. BlackByte executed a PowerShell command to query Active Directory for all of the computer hostnames and installed the Remote Server Admin Tools package ([T1059.001 Command and Scripting Interpreter: PowerShell](#)).
3. BlackByte executed hundreds of reconnaissance and system discovery commands including `net view` and `arp -a` ([T1018 Remote System Discovery, T1016 System Network Configuration Discovery](#)).
4. BlackByte created a temporary copy of itself before deleting the original `complex.exe` binary ([T1070.004 Indicator Removal on Host: File Deletion](#)). We assess this temporary file, rather than the original `complex.exe` file, was used to propagate to the next system.
5. BlackByte then made a connection to the `185.93.6[.]31` address mentioned above.

Preparing for ransom

BlackByte issued several precursor commands prior to conducting the encryption cycle. The `vssadmin resize shadowstorage` command was used to resize shadow copy storage in two different ways, setting the `/MaxSize` parameter to either “unbounded” or “401MB.” Additionally, BlackByte issued an obfuscated PowerShell command to [delete shadow copies](#) directly through [WMI](#) objects ([T1490 Inhibit System Recovery](#)).

Detection opportunity: **Vssadmin Resizing shadowstorage**

```
process == vssadmin.exe  
&&  
process_command_line_contains == resize shadowstorage&& (unbounded || 401MB)
```

Next, BlackByte issued commands to delete the **scheduled task** “Raccine Rules Updater ” disable the SQLTELEMETRY service (**T1562.001 Impair Defenses: Disable or Modify Tools**). **Raccine** is a so-called “ransomware vaccine” created by security researcher **Florian Roth**, designed to intercept and prevent precursors and active ransomware behavior.

Detection opportunity: **Raccine scheduled task deletion**

Process_name == schtasks.exe

&&

command_line_contains == (delete&& Raccine)

Process spawned by regedit.exe

c:\windows\system32\schtasks.exe 2e9e198247bf0e9bd94b42286798a5ac

Command Line: schtasks.exe /DELETE /TN "\"Raccine Rules Updater\""" /F

This command deletes the scheduled task **Raccine Rules Updater**, which may be associated with the **Raccine** Ransomware protection binary.

Deletion of the Raccine Rules Updater scheduled task

Exfiltration

Using WinRAR, the adversary compressed local data from compromised endpoints and uploaded the archives to the anonymous file-sharing sites anonymfiles[.]com and file[.]io ([T1560.001 Archive Collected Data: Archive via](#)

Utility, T1567.002 Exfiltration Over Web Service: Exfiltration to Cloud Storage). The operators then attempt to further extort the customer by threatening to release this data publicly through the BlackByte Tor leak site, an all too common tactic among ransomware operators.

Encryption

We observed the use of print bombing to deploy physical ransom notes. BlackByte has the typical text-based note titled `BlackByte_restoremyfiles.hta`. However, it also sets a scheduled task to perform a print bombing technique, which causes all connected printers to physically print ransom notes upon execution of the task (**T1053.005 Scheduled Task/Job: Scheduled Task**). At the top of every hour following an initial scheduled task setting, a note would be printed that says “Your[sic] HACKED by BlackByte team. Connect us to restore your system.”

```
Process spawned by svchost.exe  
c:\windows\system32\cmd.exe f4f684066175b77e0c3a000549d2922c
```

```
Command Line: C:\Windows\System32\cmd.exe /c for /l %x in (1,1,75) do start  
wordpad.exe /p C:\Users\tree.dll
```

This command is consistent with Ransomware execution sending Ransomware notes to printers in a technique dubbed `print bombing`.

Execution of the print bombing scheduled task

Detection opportunity: **Print bombing technique**

Process == cmd.exe

&&

Command_line_contains ==wordpad|| (notepad && /p)

Several other ransomware variants use the print bombing technique, such as Egregor and LockBit, but it's another reminder that the intent of ransomware is to cause fear and disruption among users.

For more detailed information about the encryption process, read [Trustwave's blog](#), which breaks down this routine in great detail ([T1486 Data Encrypted for Impact](#)). Interestingly, it relies on a downloaded file (forest.png, in Trustwave's case) that contains the encryption key. Without this, the encryption will fail. Additionally, Trustwave developed a decryptor based on this key, which can also be found in their blog.

Malware analysis notes

BlackByte has extensive obfuscation and some anti-debugging features that made analyzing the sample difficult. The sample was UPX-packed, and initially, we observed several Golang strings making us think this could be a Go version of BlackByte ([T1027.002 Obfuscated Files or Information: Software Packing](#)). However, after further analysis, the sample appears to be written in a combination of C and Go. Additionally, we noted the sample dynamically loaded API components, increasing the difficulty of analysis. Our sample also differed slightly from Trustwave's analysis, potentially indicating multiple variants of BlackByte in the wild, though we could not confirm this. We continue to analyze our sample and will provide updates if we find anything noteworthy.

Conclusion

Not that anyone needs to be reminded of the prevalence of both Cobalt Strike and ransomware, but hopefully this intrusion gives some insight into an ever-evolving ransomware threat landscape. As we stated in our [October Intelligence Insights blog](#), the best way to prevent a widespread ransomware infection after an adversary has already entered your environment is to identify precursor activity, such as shadow copy deletion, suspicious registry modification, or unusual process behavior.

Note: *If you've been impacted by BlackByte, [Trustwave has released a decryption tool](#). The tool relies on an encryption key found within a downloaded PNG (or other) file during the intrusion. Following Trustwave's publication, [BlackByte operators](#)*

***called out the decryptor** and questioned its capability. As such, the group's tactics may change and the tool may become unreliable.*

IOCS

Cobalt Strike beacon	c84d4ead6c5a2afa9e844806de549dcf	829751cfdc2376e916244f94baf839ce4491ccb75f0a89778c092bde79bd8643
BlackByte ransomware	9344afc63753cd5e2ee0ff9aed43dc56	1df11bc19aa52b623bdf15380e3fded56d8eb6fb7b53a2240779864b1a6474ad
Print bombing ransom note DLL	e2eb5b57a8765856be897b4f6dadca18	91f8592c7e8a3091273f0ccbfe34b2586c5998f7de63130050cb8ed36b4eec3e

Cobalt Strike server	185.93.6.31	
----------------------	-------------	--

OBSERVABLES

Technique	Description	Observable
	BlackByte execution	C:\Users\Public\complex.exe -single
	Print bombing command line	C:\Windows\System32\cmd.exe /c for /l %x in (1,1,75) do start wordpad.exe /p C:\Users\tree.dll

<p><u>T1059.001 Command and Scripting Interpreter: PowerShell</u></p> <p><u>T1027 Obfuscated Files of Information</u></p> <p><u>T1562.001 Impair Defenses: Disable or Modify Tools</u></p>	<p>EXE deletion and encoded PowerShell command to disable Windows Defender</p>	<pre>cmd /c del C:\Windows\System32\Taskmgr.exe /f /q & del C:\Windows\System32\resmon.exe /f /q & powershell -command "\$x =[System.Text.Encoding]::Unicode.GetString([System. Convert]::FromBase64String('Vw'+'BpA'+'G4ARAB'+'IA GYA'+'ZQB'+'uAG'+'QA'));Stop-Service -Name \$x;Set- Service -StartupType Disabled \$x"</pre>
---	--	---

Technique	Description	Observable
<p><u>T1059.001 Command and Scripting Interpreter: PowerShell</u></p> <p><u>T1562.001 Impair Defenses: Disable or Modify Tools</u></p>	<p>Decoded PowerShell for disabling Windows Defender</p>	<pre>cmd /c del C:\Windows\System32\Taskmgr.exe /f /q & del C:\Windows\System32\resmon.exe /f /q & powershell -command "\$x = [System.Text.Encoding]::Unicode.GetString([System.C onvert]::FromBase64String(WinDefend)); Stop-Service -Name \$x;Set-Service -StartupType Disabled \$x"</pre>
<p><u>T1490 Inhibit System Recovery</u></p>	<p>ShadowStorage resize to unbounded</p>	<pre>cmd.exe /c vssadmin resize shadowstorage /for=e: /on=e: /maxsize=unbounded</pre>

Technique	Description	Observable
<u>T1490 Inhibit System Recovery</u>	ShadowStorage resize to 401MB	cmd.exe /c vssadmin resize shadowstorage /for=h: /on=h: /maxsize=401MB
<u>T1059.001 Command and Scripting Interpreter: PowerShell</u>	PowerShell command to install Remote Server Admin Tools package	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe Install-WindowsFeature -Name "RSAT-AD-PowerShell" -IncludeAllSubFeature

Technique	Description	Observable
<u>T1059.001 Command and Scripting Interpreter: PowerShell</u>	PowerShell command to query Active Directory for computer names	powershell -command "Import-Module ActiveDirectory;Get-ADComputer -Filter * -Properties * FT Name"
<u>T1562.001 Impair Defenses: Disable or Modify Tools</u>	Scheduled task deletion	schtasks.exe /DELETE /TN "\"Raccine Rules Updater\""/F

Technique	Description	Observable
	Web shell location	\\127.0.0.1\c\$\program files\microsoft\exchange server\v15\frontend\httpproxy\owa\auth\current\themes\ukbse.aspx
<p><u>T1059.001 Command and Scripting Interpreter: PowerShell</u></p> <p><u>T1027 Obfuscated Files of Information</u></p>	Encoded PowerShell for deleting shadow copies via Get-WmiObject	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -command "\$x = [System.Text.Encoding]::Unicode.GetString([System.Convert]::FromBase64String('RwBlAHQALQBxAG0AaQBPAgIAagBlAGMAdAAg'+ 'AFcAaQBuADMAMgBfAFMAaABhAGQAbwB3AGMABwBwAHkAIAB8AC'+ 'AARgBvAHlARQBhAGMAaAAtAE8AYgBqAGUAYwB0ACA AewAkA'+ 'F8ALgBEAGUAbABlAHQAZQAoACkAOwB9AA=='));Invoke-Expression \$x"

Technique	Description	Observable
<p><u>T1059.001 Command and Scripting Interpreter: PowerShell</u></p> <p><u>T1490 Inhibit System Recovery</u></p>	<p>Decoded PowerShell for deleting shadow copies via Get-WmiObject</p>	<p>C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -command "\$x = [System.Text.Encoding]::Unicode.GetString([System.Convert]::FromBase64String(Get-WmiObject Win32_Shadowcopy ForEach-Object {\$_.Delete();}));Invoke-Expression \$x"</p>

Technique	Description	Observable
<p><u>T1112 Modify Registry</u></p>	<p>Commands to add registry values</p>	<pre>reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\ Policies\System /v EnableLinkedConnections /t REG_DWORD /d 1 /f reg add HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\ Policies\System /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f reg add HKLM\SYSTEM\CurrentControlSet\Control\FileSystem /v LongPathsEnabled /t REG_DWORD /d 1 /f</pre>

Technique	Description	Observable
<u>T1562.004 Impair Defenses: Disable or Modify System Firewall</u>	Netsh firewall rule additions	netsh advfirewall firewall set rule group="Network Discovery" new enable=Yes netsh advfirewall firewall set rule group="File and Printer Sharing" new enable=Yes

References

<https://www.bleepingcomputer.com/forums/t/755181/blackbyte-ransomware-blackbyte-support-topic/>

<https://www.linkedin.com/pulse/english-blackbyte-ransomware-mysterious-dropper-encoder-fasolo>

<https://www.desmoinesregister.com/story/money/agriculture/2021/10/06/iowa-grain-cooperative-recovering-cyberattack-remains-mum-ransom/6007123001/>

<https://www.sentinelone.com/blog/the-good-the-bad-and-the-ugly-in-cybersecurity-week-40-3/>

<https://mcit.gov.ws/2021/08/06/cs-advisory-ca003-blackbyte-ransomware/>

https://www.reddit.com/r/sysadmin/comments/q0b8ra/blackbyte_ransomware/

<https://community.spiceworks.com/topic/2330823-suspicious-draft-email>

<https://www.mandiant.com/resources/pst-want-shell-proxysql-exploiting-microsoft-exchange-servers>

<https://docs.microsoft.com/en-us/answers/questions/545608/unexpected-spam-email-in-outlook-draft-folder.html>

https://www.reddit.com/r/sysadmin/comments/ppk6di/mysterious_email_in_exchange_users_drafts/

<https://twitter.com/cyb3rops/status/1448209631331954693?s=20>

<https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/blackbyte-ransomware-pt-1-in-depth-analysis/>

<https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/blackbyte-ransomware-pt-2-code-obfuscation-analysis/>

<https://docs.microsoft.com/en-us/windows/win32/fileio/maximum-file-path-limitation?tabs=cmd>