Magnitude exploit kit – evolution

securelist.com/magnitude-exploit-kit-evolution/97436/

Authors

Expert Boris Larin

Exploit kits are not as widespread as they used to be. In the past, they relied on the use of already patched vulnerabilities. Newer and more secure web browsers with automatic updates simply do not allow known vulnerabilities to be exploited. It was very different back in the heyday of Adobe Flash because it's just a plugin for a web browser, meaning that even if the user has an up-to-date browser, there's a non-zero chance that Adobe Flash may still be vulnerable to 1-day exploits. Now that Adobe Flash is about to reach its end-of-life date at the end of this year, it is disabled by default in all web browser and has pretty much been replaced with open standards such as HTML5, WebGL, WebAssembly. The decline of exploit kits can be linked to the decline of Adobe Flash, but exploit kits have not disappeared completely. They have adapted and switched to target users of Internet Explorer without the latest security updates installed.

Microsoft Edge replaced Internet Explorer as a default web browser with the release of Windows 10 in 2015, but Internet Explorer is still installed for backward compatibility on machines running Windows 10 and it has remained a default web browser for Windows

7/8/8.1. The switch to Microsoft Edge development also meant that Internet Explorer would no longer be actively developed and would only receive vulnerability patches without general security improvements. Still, somehow, Internet Explorer remains a relatively popular web browser. According to NetMarketShare, as of April 2020 Internet Explorer is used on 5.45% of desktop computers (for comparison, Firefox accounts for 7.25%, Safari 3.94%, Edge 7.76%). Despite the security of Internet Explorer being five years behind that of its modern counterparts, it supports a number of legacy script engines. CVE-2018-8174 is a vulnerability in a legacy VBScript engine that was originally discovered in the wild as an exploited zero-day. The majority of exploit kits quickly adopted it as their primary exploit.

Since the discovery of CVE-2018-8174 a few more vulnerabilities for Internet Explorer have been discovered as in-the-wild zero-days: CVE-2018-8653, CVE-2019-1367, CVE-2019-1429, and CVE-2020-0674. All of them exploited another legacy component of Internet Explorer – a JScript engine. It felt like it was just a matter of time until exploit kits adopted these new exploits.

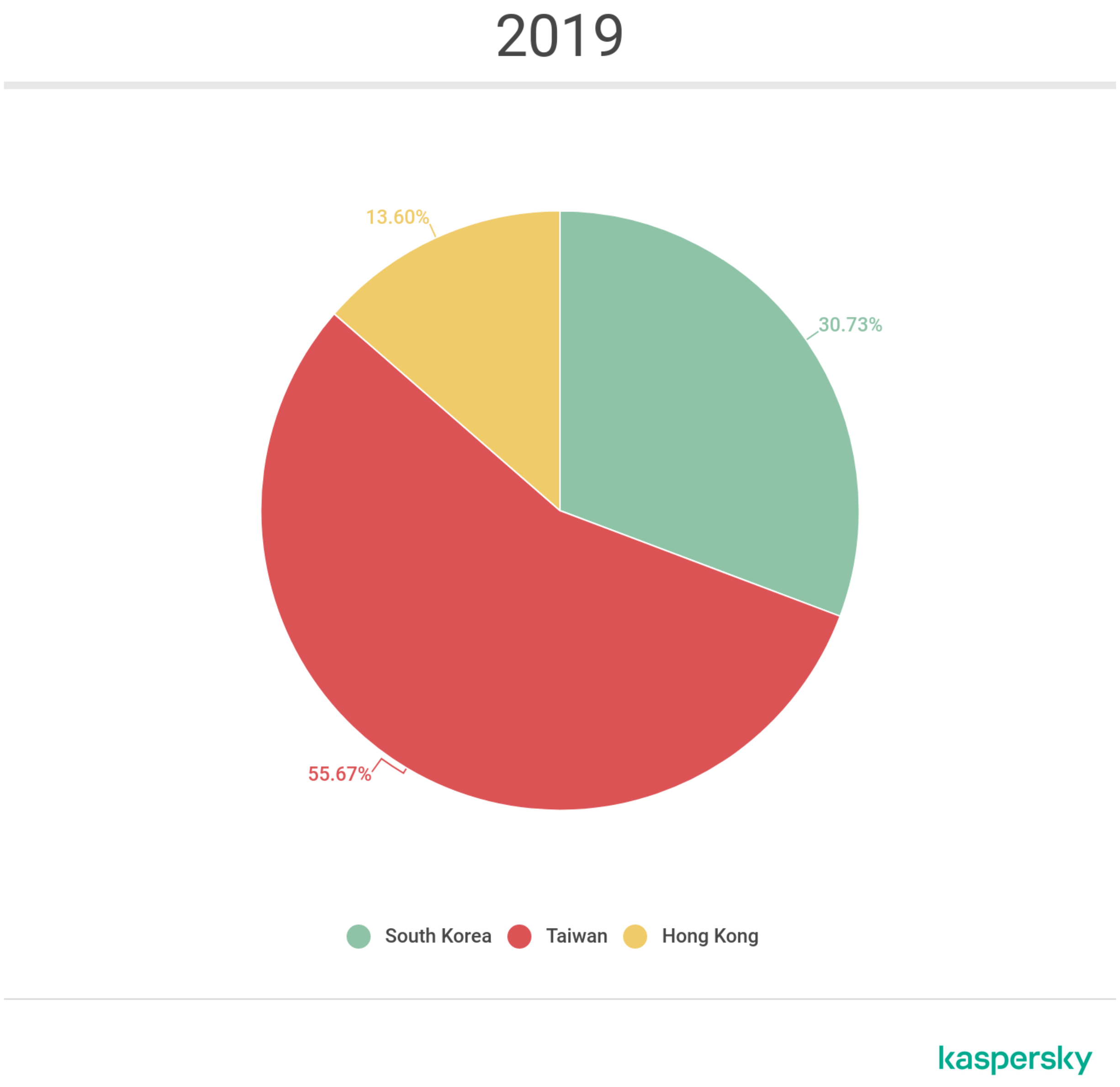
Exploit kits still play a role in today's threat landscape and continue to evolve. For this blogpost I studied and analyzed the evolution of one of the most sophisticated exploit kits out there – Magnitude EK – for a whole year.

This blogpost in a nutshell:

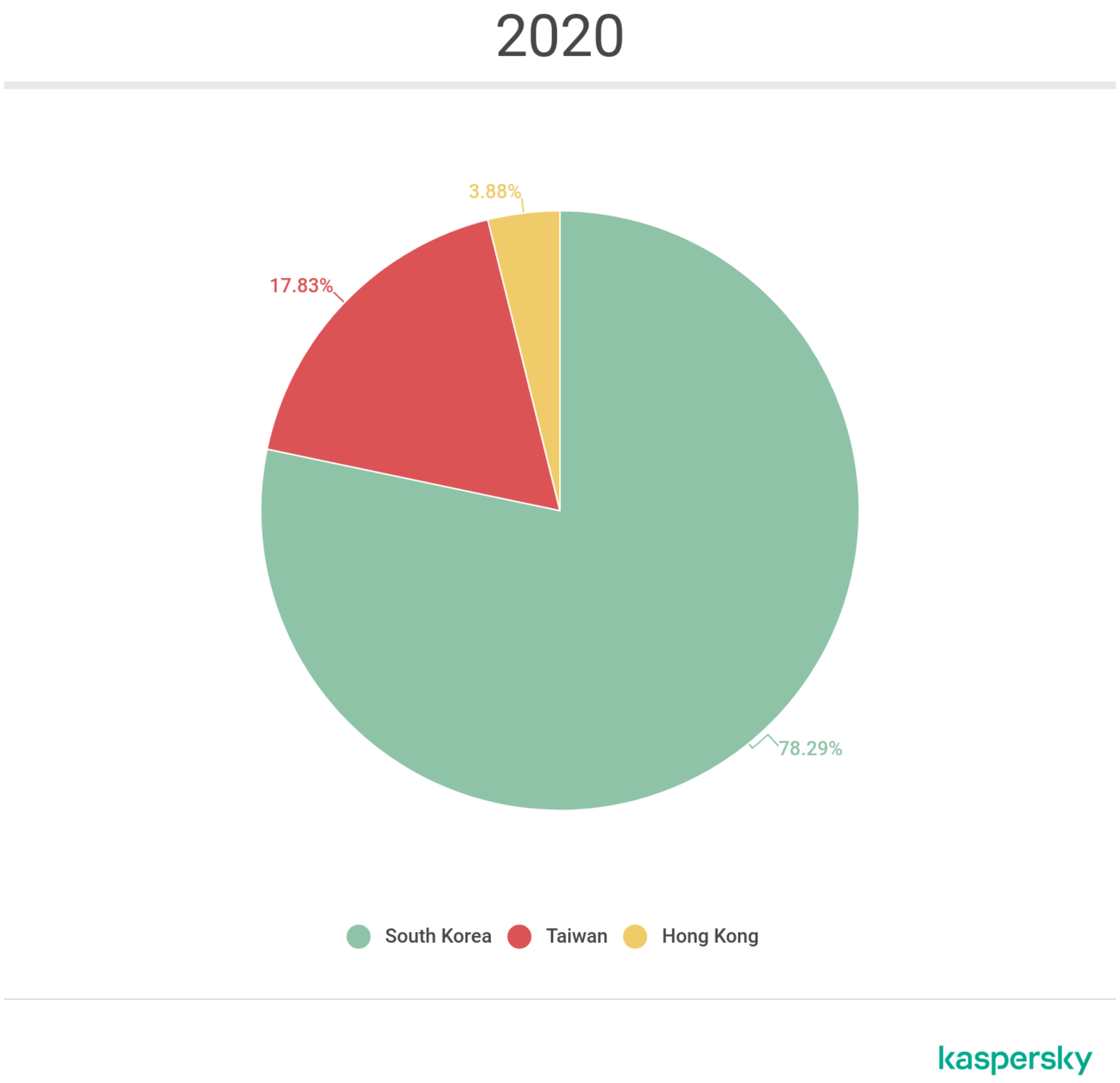
- Magnitude EK continues to deliver ransomware to Asia Pacific (APAC) countries via malvertising
- Study of the exploit kit's activity over a period of 12 months shows that Magnitude EK is actively maintained and undergoes continuous development
- In February this year Magnitude EK switched to an exploit for the more recent vulnerability CVE-2019-1367 in Internet Explorer (originally discovered as an exploited zero-day in the wild)
- Magnitude EK uses a previously unknown elevation of privilege exploit for CVE-2018-8641 developed by a prolific exploit writer

Introduction

Magnitude EK is one of the longest-standing exploit kits. It was on offer in underground forums from 2013 and later became a private exploit kit. As well as a change of actors, the exploit kit has switched its focus to deliver ransomware to users from specific Asia Pacific (APAC) countries via malvertising.



Active attacks by Magnitude EK in 2019 according to Kaspersky Security Network (KSN) (download)



Active attacks by Magnitude EK in 2020 according to Kaspersky Security Network (KSN) (download)

Our statistic shows that this campaign continues to target APAC countries to this day and during the year in question Magnitude EK always used its own ransomware as a final payload.

Infection vector

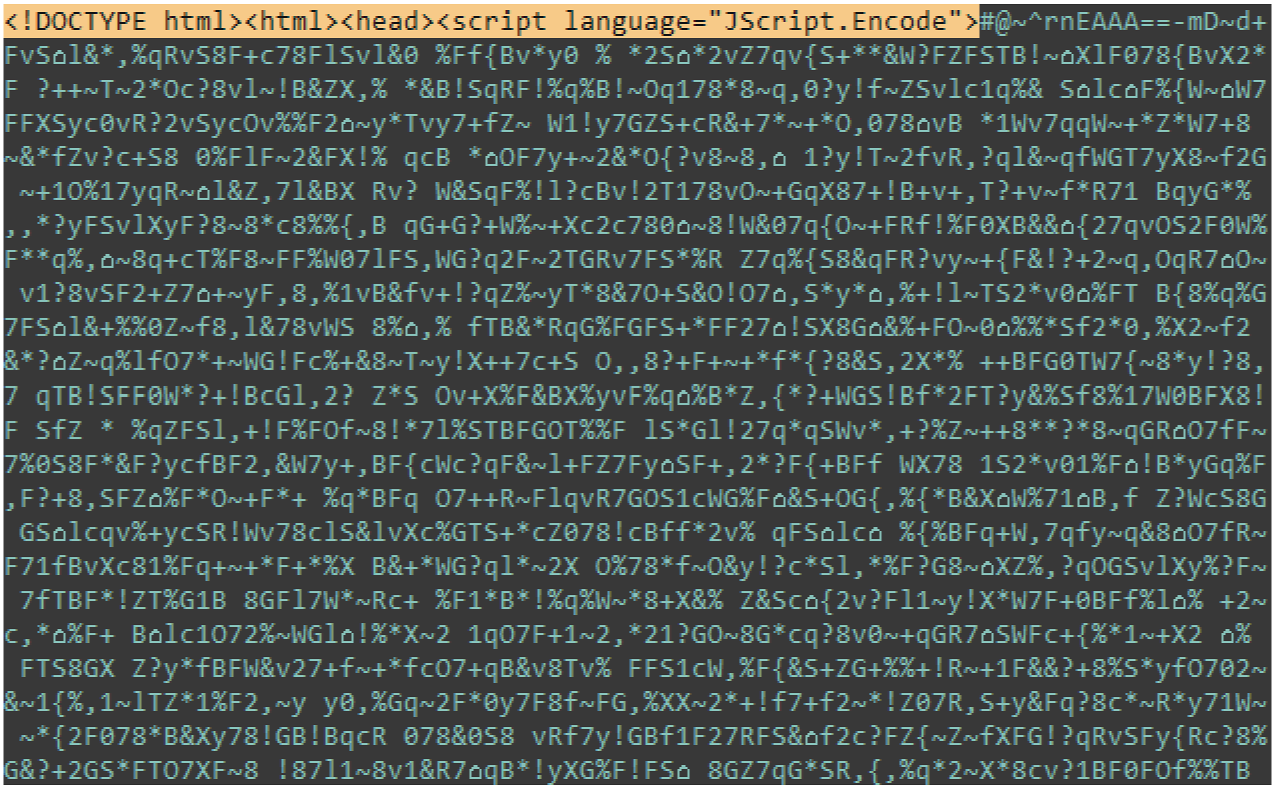
Like the majority of exploit kits out there, in 2019 Magnitude EK used CVE-2018-8174. However, the attackers behind Magnitude EK were one of the first to adopt the much newer vulnerability CVE-2019-1367 and they have been using it as their primary exploit since

February 11, 2020. As was the case with CVE-2018-8174, they didn’t develop their own exploit for CVE-2019-1367, instead reusing the original zero-day and modifying it with their own shellcode and obfuscation.

CVE-2019-1367 is a Use-After-Free vulnerability due to a garbage collector not tracking a value that was not rooted in the legacy JavaScript engine jscript.dll. By default, Internet Explorer 11 uses Jscript9.dll, but it’s still possible to execute the script using the legacy engine by enabling compatibility mode with Internet Explorer 7/8. This can be done with the following script attributes:

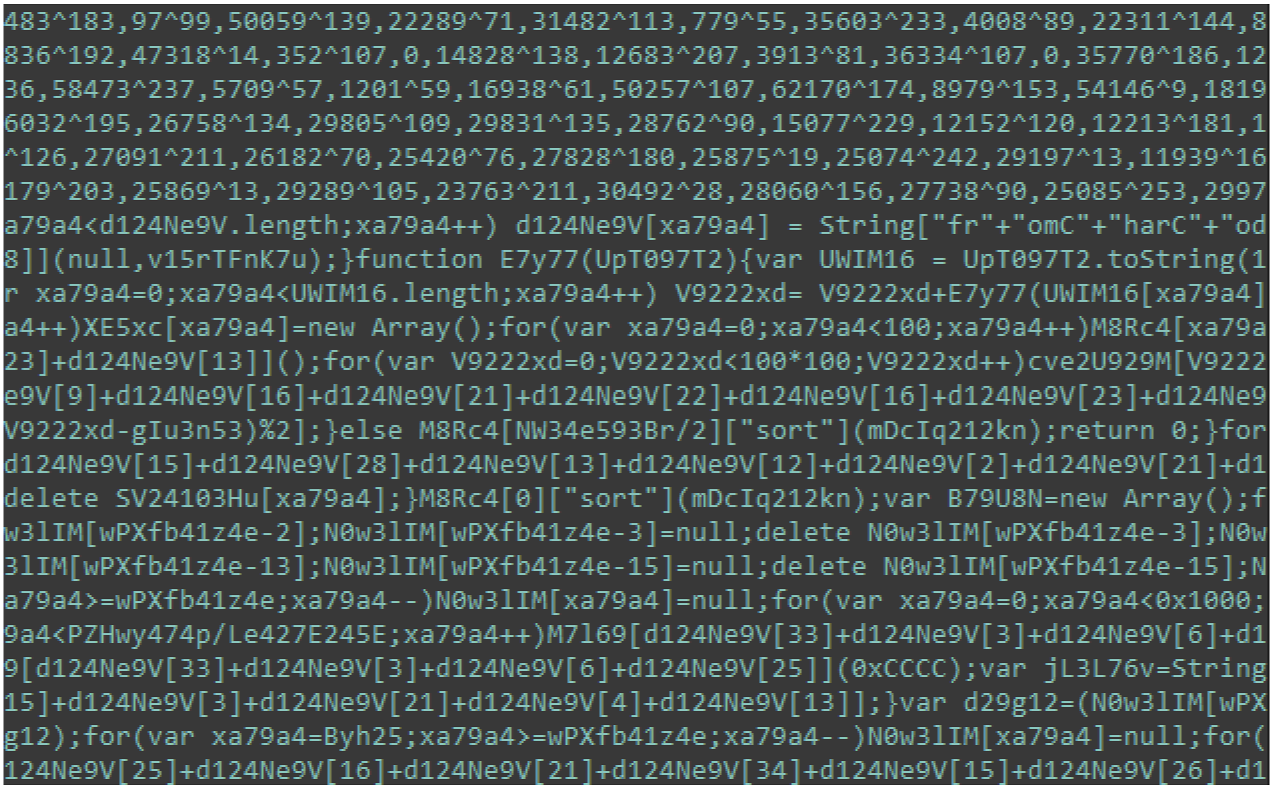
```
<meta http-equiv="x-ua-compatible" content="IE=EmulateIE8" />
<script language="JScript.Compact">…</script>

<meta http-equiv="x-ua-compatible" content="IE=EmulateIE8" />
<script language="JScript.Encode">…</script>
```

The original exploit uses JScript.Compact, a special profile defined for embedded devices. But JScript.Encode is much more interesting because it was developed by Microsoft to protect scripts and prevent source code from being copied. This script attribute can execute scripts encoded with Microsoft Script Encoder (screnc.exe) and it also disables script debugging. Basically, it’s a DRM for JavaScript. Magnitude EK changed from its original exploit to take advantage of this feature.



```
<!DOCTYPE html><html><head><script language="JScript.Encode">#@~^rnEAAA==-mD~d+
FvS⌂l&*,%qRvS8F+c78FlSvl&0 %Ff{Bv*y0 % *2S⌂*2vZ7qv{S+**&W?FZFSTB!~⌂XlF078{BvX2*
F ?++~T~2*Oc?8vl~!B&ZX,% *&B!SqRF!%q%B!~Oq178*8~q,0?y!f~ZSvlc1q%& S⌂lc⌂F%{W~⌂W7
FFXSyc0vR?2vSycOv%%F2⌂~y*Tvy7+fZ~ W1!y7GZS+cR&+7*~+*O,078⌂vB *1Wv7qqW~+*Z*W7+8
~&*fZv?c+S8 0%FlF~2&FX!% qcB *⌂OF7y+~2&*O{?v8~8,⌂ 1?y!T~2fvR,?ql&~qfWGT7yX8~f2G
 ~+1O%17yqR~⌂l&Z,7l&BX Rv? W&SqF%!l?cBv!2T178vO~+GqX87+!B+v+,T?+v~f*R71 BqyG*%
,,*?yFSvlXyF?8~8*c8%%{,B qG+G?+W%~+Xc2c780⌂~8!W&07q{O~+FRf!%F0XB&&⌂{27qvOS2F0W%
F**q%,⌂~8q+cT%F8~FF%W07lFS,WG?q2F~2TGRv7FS*%R Z7q%{S8&qFR?vy~+{F&!?+2~q,OqR7⌂O~
 v1?8vSF2+Z7⌂+~yF,8,%1vB&fv+!?qZ%~yT*8&7O+S&O!O7⌂,S*y*⌂,%+!l~TS2*v0⌂%FT B{8%q%G
7FS⌂l&+%%0Z~f8,l&78vWS 8%⌂,% fTB&*RqG%FGFS+*FF27⌂!SX8G⌂&%+FO~0⌂%%*Sf2*0,%X2~f2
&*?⌂Z~q%lfO7*+~WG!Fc%+&8~T~y!X++7c+S O,,8?+F+~+*f*{?8&S,2X*% ++BFG0TW7{~8*y!?8,
7 qTB!SFF0W*?+!BcGl,2? Z*S Ov+X%F&BX%yvF%q⌂%B*Z,{*?+WGS!Bf*2FT?y&%Sf8%17W0BFX8!
F SfZ * %qZFSl,+!F%FOf~8!*7l%STBFGOT%%F lS*Gl!27q*qSWv*,+?%Z~++8**?*8~qGR⌂O7fF~
7%0S8F*&F?ycfBF2,&W7y+,BF{cWc?qF&~l+FZ7Fy⌂SF+,2*?F{+BFf WX78 1S2*v01%F⌂!B*yGq%F
,F?+8,SFZ⌂%F*O~+F*+ %q*BFq O7++R~FlqvR7GOS1cWG%F⌂&S+OG{,%{*B&X⌂W%71⌂B,f Z?WcS8G
 GS⌂lcqv%+ycSR!Wv78clS&lvXc%GTS+*cZ078!cBff*2v% qFS⌂lc⌂ %{%BFq+W,7qfy~q&8⌂O7fR~
F71fBvXc81%Fq+~+*F+*%X B&+*WG?ql*~2X O%78*f~O&y!?c*Sl,*%F?G8~⌂XZ%,?qOGSvlXy%?F~
 7fTBF*!ZT%G1B 8GFl7W*~Rc+ %F1*B*!%q%W~*8+X&% Z&Sc⌂{2v?Fl1~y!X*W7F+0BFf%l⌂% +2~
c,*⌂%F+ B⌂lc1O72%~WGl⌂!%*X~2 1qO7F+1~2,*21?GO~8G*cq?8v0~+qGR7⌂SWFc+{%*1~+X2 ⌂%
 FTS8GX Z?y*fBFW&v27+f~+*fcO7+qB&v8Tv% FFS1cW,%F{&S+ZG+%%+!R~+1F&&?+8%S*yfO702~
&~1{%,1~lTZ*1%F2,~y y0,%Gq~2F*0y7F8f~FG,%XX~2*+!f7+f2~*!Z07R,S+y&Fq?8c*~R*y71W~
 ~*{2F078*B&Xy78!GB!BqcR 078&0S8 vRf7y!GBf1F27RFS&⌂f2c?FZ{~Z~fXFG!?qRvSFy{Rc?8%
G&?+2GS*FTO7XF~8 !87l1~8v1&R7⌂qB*!yXG%F!FS⌂ 8GZ7qG*SR,{,%q*2~X*8cv?1BF0FOf%%TB
```

Exploit packed with JScript.Encode technique



```
483^183,97^99,50059^139,22289^71,31482^113,779^55,35603^233,4008^89,22311^144,8
836^192,47318^14,352^107,0,14828^138,12683^207,3913^81,36334^107,0,35770^186,12
36,58473^237,5709^57,1201^59,16938^61,50257^107,62170^174,8979^153,54146^9,1819
6032^195,26758^134,29805^109,29831^135,28762^90,15077^229,12152^120,12213^181,1
^126,27091^211,26182^70,25420^76,27828^180,25875^19,25074^242,29197^13,11939^16
179^203,25869^13,29289^105,23763^211,30492^28,28060^156,27738^90,25085^253,2997
a79a4<d124Ne9V.length;xa79a4++) d124Ne9V[xa79a4] = String["fr"+"omC"+"harC"+"od
8]](null,v15rTFnK7u);}function E7y77(UpT097T2){var UWIM16 = UpT097T2.toString(1
r xa79a4=0;xa79a4<UWIM16.length;xa79a4++) V9222xd= V9222xd+E7y77(UWIM16[xa79a4]
a4++)XE5xc[xa79a4]=new Array();for(var xa79a4=0;xa79a4<100;xa79a4++)M8Rc4[xa79a
23]+d124Ne9V[13]]();for(var V9222xd=0;V9222xd<100*100;V9222xd++)cve2U929M[V9222
e9V[9]+d124Ne9V[16]+d124Ne9V[21]+d124Ne9V[22]+d124Ne9V[16]+d124Ne9V[23]+d124Ne9
V9222xd-gIu3n53)%2];}else M8Rc4[NW34e593Br/2]["sort"](mDcIq212kn);return 0;}for
d124Ne9V[15]+d124Ne9V[28]+d124Ne9V[13]+d124Ne9V[12]+d124Ne9V[2]+d124Ne9V[21]+d1
delete SV24103Hu[xa79a4];}M8Rc4[0]["sort"](mDcIq212kn);var B79U8N=new Array();f
w3lIM[wPXfb41z4e-2];N0w3lIM[wPXfb41z4e-3]=null;delete N0w3lIM[wPXfb41z4e-3];N0w
3lIM[wPXfb41z4e-13];N0w3lIM[wPXfb41z4e-15]=null;delete N0w3lIM[wPXfb41z4e-15];N
a79a4>=wPXfb41z4e;xa79a4--)N0w3lIM[xa79a4]=null;for(var xa79a4=0;xa79a4<0x1000;
9a4<PZHwy474p/Le427E245E;xa79a4++)M7l69[d124Ne9V[33]+d124Ne9V[3]+d124Ne9V[6]+d1
9[d124Ne9V[33]+d124Ne9V[3]+d124Ne9V[6]+d124Ne9V[25]](0xCCCC);var jL3L76v=String
15]+d124Ne9V[3]+d124Ne9V[21]+d124Ne9V[4]+d124Ne9V[13]];}var d29g12=(N0w3lIM[wPX
g12);for(var xa79a4=Byh25;xa79a4>=wPXfb41z4e;xa79a4--)N0w3lIM[xa79a4]=null;for(
124Ne9V[25]+d124Ne9V[16]+d124Ne9V[21]+d124Ne9V[34]+d124Ne9V[15]+d124Ne9V[26]+d1
```

Unpacked exploit. Shellcode, names and some strings are obfuscated

Shellcode

Their shellcodes piqued my interest. They use a huge number of different shellcode encoders, from the classical Metasploit shikata_ga_nai encoder and DotNetToJScript to a variety of custom encoders and stagers.

It was also impossible not to notice the changes happening to their main shellcode responsible for launching the ransomware payload. The attackers are fine-tuning their arsenal on a regular basis.

Magnitude EK has existed since at least 2013, but below you can see just the changes to payload/shellcode that occurred over the period of one year (June 2019 to June 2020). During this period we observed attacks happening almost every day.

Timeline of shellcode/payload changes

Date	Description
June 2019	Shellcode downloads a payload that's decrypted with a custom xor-based algorithm. All strings are assembled on stack and to change payload the URL shellcode needs to be recompiled. The payload is a PE module. The module export function name is hardcoded to "GrfeFVGRe". The payload is executed in an Internet Explorer process. It contains an elevation of privilege exploit with support for x86 and x64 versions of Windows and an encrypted ransomware payload. After elevation of privilege it injects the ransomware payload to other processes, spawns the wuapp.exe process and injects there as well. If process creation fails, it also runs the ransomware from the current process.
July 20, 2019	Payload module export function name is auto-generated.
November 11, 2019	Shellcode tries to inject the payload to other processes. If API function Process32First fails, it spawns the process wuapp.exe from Windows directory and injects the payload there. The injection method is WriteProcessMemory + CreateRemoteThread. The payload is ransomware without elevation of privilege. The payload module export function name is hardcoded again, but now to "lssrcdxhg".
November 20, 2019	Looks like they messed up the folder with shellcodes; in some attacks they use a shellcode from June, and later the same day they start to use their November shellcode with the new hardcoded export name "by5eftgdbfgsq323".
November 23, 2019	They start to use the elevation of privilege exploit again, but now they also check the integrity level of the process. If it's a low integrity process, then they execute the payload with the exploit in the current process; if that's not the case, then it's injected into other processes. The process is no longer created from shellcode, but it's still created from the payload. The payload module export name is hardcoded to "gv65eytervsawer2".

January 17, 2020	It looks like the attackers had a short holiday at the beginning of the year. The shellcode remains the same, but the payload module export function name is hardcoded to “i4eg65tgq3f4”. The payload changed a bit. The name of the created process is now assembled on stack. The name of the process also changed – it no longer spawns a wuapp.exe, but instead launches the calculator calc.exe and injects the ransomware payload there.
January 27, 2020	The payload is no longer a PE module but plain shellcode. The payload consists of ransomware without elevation of privilege.
February 4, 2020	The payload is a PE module again, but once again the export name is auto-generated.
February 10, 2020	The shellcode comes with two URLs for different payloads. The shellcode checks the integrity level and depending on process integrity level, it executes the elevation of privilege payload or uses the ransomware payload straightaway. All strings and function imports in the exploit are now obfuscated. The payload does not spawn a new process, and only injects to others.
February 11, 2020	Magnitude EK starts using CVE-2019-1367 as its primary exploit. The attackers use the shellcode from January 27, 2020, but they have modified it to check for the name of a particular process. If the process exists, they don’t execute the payload from Internet Explorer. The process name is “ASDSvc” (AhnLab, Inc.).
February 17, 2020	The attackers switch to the shellcode from February 10, 2020, but the payload module export function name is hardcoded to “xs324qsawezzse”.
February 28, 2020	Shellcode encryption is removed. The payload module export function name is hardcoded to “sawd6vf3y5”.
March 1, 2020	Strings are no longer stored on stack.
March 6, 2020	Back to the shellcode from February 17, 2020.
March 10, 2020	The attackers add some functionality implemented after February 17, 2020: payload encryption is removed and strings are no longer stored on stack. The payload module export function name is still hardcoded to “xs324qsawezzse”.
March 16, 2020	Functionality added so as not to inject into a particular process (explorer.exe). The injection method is also changed to NtCreateSection + NtMapViewOfSection + RtlCreateUserThread.

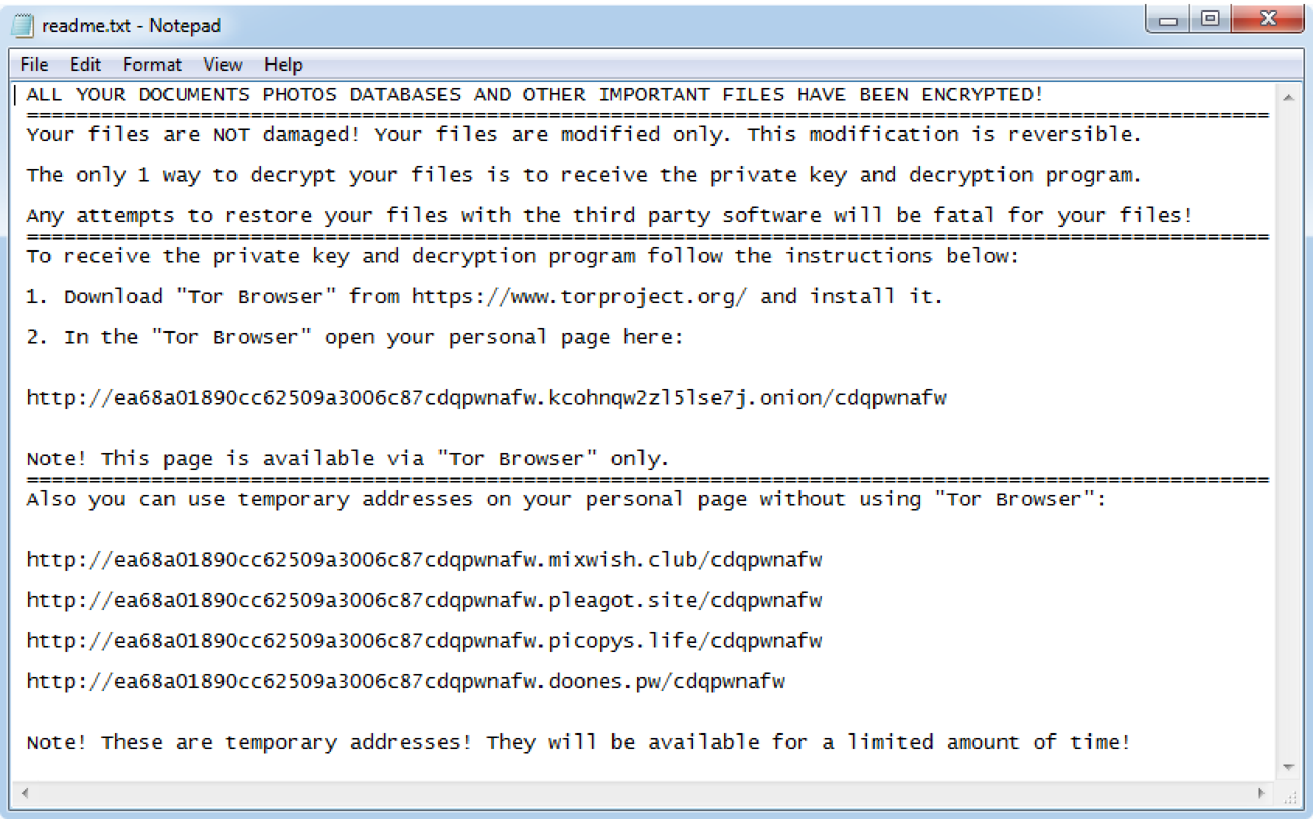
April 2, 2020	The attackers add some functionality similar to that used in November 2019. They check the integrity level of a process and if it's a low integrity process, they execute the payload from the current process. If that's not the case, they inject it to other processes (other than explorer.exe) and at the end create a new process and inject it there as well. The created processes are C:\Program Files\Windows Media Player\wmlaunch.exe or C:\Program Files (x86)\Windows Media Player\wmlaunch.exe depending on whether it's a WOW64 process or not.
April 4, 2020	Shellcode updated to use a new injection technique: NtQueueApcThread. The shellcode also comes with a URL for a ransomware payload without elevation of privilege. The shellcode checks the integrity level and if it's a low integrity process, the shellcode calls ExitProcess(). Use of the hardcoded export name "xs324qsawezzse" is also stopped.
April 7, 2020	Back to the shellcode from April 2, 2020.
May 5, 2020	Previously the attackers adjusted their injection method, but now they revert back to injection via the WriteProcessMemory + CreateRemoteThread technique.
May 6, 2020	They continue to make changes to the code injection method. From now on they use NtCreateThreadEx.

Elevation of privilege exploit

The elevation of privilege exploit used by Magnitude EK is quite interesting. When I saw it for the first time, I wasn't able to recognize this particular exploit. It exploited a vulnerability in the win32k kernel driver and closer analysis revealed that this particular vulnerability was fixed in December 2018. According to Microsoft, only two win32k-related elevation of privilege vulnerabilities were fixed that month – CVE-2018-8639 and CVE-2018-8641. Microsoft previously shared more information with us about CVE-2018-8639, so we can say with some certainty that the encountered exploit uses vulnerability CVE-2018-8641. The exploit has huge code similarities with another zero-day that we had found previously – CVE-2019-0859. Based on these similarities, we attribute this exploit to the prolific exploit writer known as "Volodya", "Volodimir" or "BuggiCorp". Volodya is famous for selling zero-day exploits to both APT groups and criminals. In the past, Volodya advertised his services at exploit(dot)in, the same underground forum where Magnitude EK was once advertised. We don't currently know if the exploit for CVE-2018-8641 was initially used as a zero-day exploit or it was developed as a 1-day exploit through patch diffing. It's also important to note that a public exploit for CVE-2018-8641 also exists, but it's incorrectly designated to CVE-2018-8639 and it exploits the vulnerability in another fashion, meaning there are two completely different exploits for the same vulnerability.

Ransomware

Magnitude EK uses its own ransomware as its final payload. The ransomware comes with a temporary encryption key and list of domain names and the attackers change them frequently. Files are encrypted with the use of Microsoft CryptoAPI and the attackers use Microsoft Enhanced RSA and AES Cryptographic Provider (PROV_RSA_AES). The initialization vector (IV) is generated pseudo randomly for each file and a 0x100 byte long blob with encrypted IV is appended to the end of the file. The ransomware doesn't encrypt the files located in common folders such as documents and settings, appdata, local settings, sample music, tor browser, etc. Before encryption, the extensions of files are checked against a hash table of allowed file extensions that contains 715 entries. A ransom note is left in each folder with encrypted files and at the end a notepad.exe process is created to display the ransom note. To hide the origin of the executed process, the ransomware uses one of two techniques: "wmic process call create" or "pcalua.exe –a … -c …". After encryption the ransomware also attempts to delete backups of the files with the "wmic shadowcopy delete" command that is executed with a UAC-bypass.



```
ALL YOUR DOCUMENTS PHOTOS DATABASES AND OTHER IMPORTANT FILES HAVE BEEN ENCRYPTED!
=================================================================================================
Your files are NOT damaged! Your files are modified only. This modification is reversible.

The only 1 way to decrypt your files is to receive the private key and decryption program.

Any attempts to restore your files with the third party software will be fatal for your files!
=================================================================================================
To receive the private key and decryption program follow the instructions below:

1. Download "Tor Browser" from https://www.torproject.org/ and install it.

2. In the "Tor Browser" open your personal page here:


http://ea68a01890cc62509a3006c87cdqpwnafw.kcohnqw2zl5lse7j.onion/cdqpwnafw


Note! This page is available via "Tor Browser" only.
=================================================================================================
Also you can use temporary addresses on your personal page without using "Tor Browser":


http://ea68a01890cc62509a3006c87cdqpwnafw.mixwish.club/cdqpwnafw

http://ea68a01890cc62509a3006c87cdqpwnafw.pleagot.site/cdqpwnafw

http://ea68a01890cc62509a3006c87cdqpwnafw.picopys.life/cdqpwnafw

http://ea68a01890cc62509a3006c87cdqpwnafw.doones.pw/cdqpwnafw


Note! These are temporary addresses! They will be available for a limited amount of time!
```

Example of Magnitude EK ransom note

The core of the ransomware did not undergo many changes throughout the year. If we compare old samples with more recent versions, there are only a few notable changes:

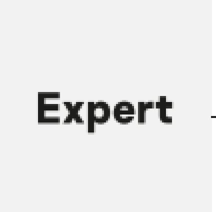
- In older versions, immediately at launch the payload gets the default UI language of the operating system using the GetSystemDefaultUILanguage API function and compares the returned value against a couple of hardcoded language IDs (e.g. zh-HK – Hong Kong S.A.R., zh-MO – Macao S.A.R., zh-CN – People's Republic of China, zh-SG – Singapore, zh-TW – Taiwan, ko-KR – Korea, ms-BN – Brunei Darussalam, ms-MY – Malaysia). If the language ID doesn't match, then ExitProcess() will be executed. In newer versions, the check for the language ID was removed.
- In older versions, the ransomware deletes file backups with the command "cmd.exe /c "%SystemRoot%\system32\wbem\wmic shadowcopy delete" via UAC-bypass in eventvwr.exe. In the newer version, the command is obfuscated with caret character insertion "cmd.exe /c "%SystemRoot%\system32\wbem\wmic ^s^h^a^d^o^w^c^o^p^y^ ^d^e^l^e^t^e" and executed via UAC-bypass in CompMgmtLauncher.exe.

Conclusions

The total volume of attacks performed by exploit kits has decreased, but they still exist, are still active, and still pose a threat, and therefore need to be treated seriously. Magnitude is not the only active exploit kit and we see other exploit kits that are also switching to newer exploits for Internet Explorer. We recommend installing security updates, migrating to a newer operating system (make sure you stay up to date with Windows 10 builds) and also not using Internet Explorer as your web browser. Throughout the entire Magnitude EK campaign we have detected the use of Internet Explorer exploits with the verdict PDM:Exploit.Win32.Generic.

- Browser
- Exploit Kits
- Malware Descriptions
- Malware Technologies
- Ransomware
- Vulnerabilities and exploits

Authors



Boris Larin

Magnitude exploit kit – evolution

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