# 9. Deputydog Campaign

Reference: <https://www.fireeye.com/blog/threat-research/2014/02/operation-snowman-deputydog-actor-compromises-us-veterans-of-foreign-wars-website.html>

On February 11, FireEye identified a zero-day exploit (*CVE-2014-0322*) being served up from the U.S. Veterans of Foreign Wars’ website (*vfw[.]org*). After compromising the VFW website, the attackers added an iframe into the beginning of the website’s HTML code that loads the attacker’s page in the background. The attacker’s HTML/JavaScript page runs a Flash object, which orchestrates the remainder of the exploit. The exploit includes calling back to the IE 10 vulnerability trigger, which is embedded in the JavaScript. Specifically, visitors to the VFW website were silently redirected through an iframe to the exploit at [*www.[REDACTED].com/Data/img/img.html*](http://www.[REDACTED].com/Data/img/img.html). The attacker uses the Microsoft.XMLDOM ActiveX control to load a one-line XML string containing a file path to the EMET DLL. Then the exploit code parses the error resulting from the XML load order to determine whether the load failed because the EMET DLL is not present. The exploit proceeds only if this check determines that the EMET DLL is not present. Once the attacker’s code has full memory access through the corrupted Flash Vector object, the code searches through loaded libraries gadgets by machine code. The attacker then overwrites the vftable pointer of a flash.Media.Sound() object in memory to point to the pivot and begin ROP. After successful exploitation, the code repairs the corrupted Flash Vector and flash.Media.Sound to continue execution. Subsequently, the malicious Flash code downloads a file containing the dropped malware payload. The beginning of the file is a JPG image; the end of the file (offset 36321) is the payload, encoded with an XOR key of 0x95. The attacker appends the payload to the shellcode before pivoting to code control. Then, when the shellcode is executed, the malware creates files “*sqlrenew.txt*” and “*stream.exe*”. The tail of the image file is decoded, and written to these files. “*sqlrenew.txt*” is then executed with the LoadLibraryA Windows API call. As documented above, this exploit dropped an XOR (0x95) payload that executed a ZxShell backdoor (MD5: *8455bbb9a210ce603a1b646b0d951bce*). The ZxShell backdoor is a widely used and publicly available tool used by multiple threat actors linked to cyber espionage operations. This particular variant called back to a command and control server located at *newss[.]effers[.]com*. This domain currently resolves to *118.99.60.142*. The domain *info[.]flnet[.]org* also resolved to this IP address on 2014-02-12.

# HawkEye Campaign

Reference: <https://www.fortinet.com/blog/threat-research/hawkeye-malware-analysis>

Here is the email content, masquerading as an airline ticket confirmation, which asks the targeted victim to click on a link. It was designed so that a victim downloads a 7z file from the link shown in figure 1 that contains this new variant of HawkEye and runs it on the victim’s system. Unfortunately, on initial analysis the URL was not available and I received a “404 Not Found” message in the browser. Browsing to its main page. It turned out to be an FTP service, containing several related network folders about this campaign, with most containing the same malware sample (Figure 2). After the downloaded 7z file was decompressed, we retrieved the EXE file “TICKET%2083992883992AIR8389494VERVED37783PDF.exe”, which is the new variant of HawkEye. Once HawkEye started, it spawned a suspended child process, “RegAsm.exe”, from the Microsoft .Net framework installation directory – which is a tool for Assembly Registration. Meanwhile, HawkEye extracted a PE file into its memory and then moved the PE file into “RegAsm.exe”. The dynamically extracted PE file is the main program of HawkEye. It’s called “HawkEye\_RegAsm,” to differentiate these files in the analysis. HawkEye\_RegAsm began running after resuming running “RegAsm.exe” after being suspended.HawkEye\_RegAsm is a .Net written program, which is packed by ConfuserEx v1.0.0 to protect itself. HawkEye\_RegAsm starts a thread to perform the above tasks, and then every 10 minutes it sends its collected information to its Yandex email address. HawkEye\_RegAsm sets up a clipboard and keyboard logger using Windows-native APIs (such as SetWindowsHookEx, SetClipboardViewer, etc.) Its local functions can record victim’s behaviors when the victim types on the keyboard as well as when copying data into the system clipboard. Figure 4 shows an example of the information that HawkEye\_RegAsm collected from its keyboard and clipboard logger, as well as the software title from when the event occurred. HawkEye\_RegAsm performs a similar task as to the RegAsm.exe. It spawns two suspended child processes, “vbc.exe”, which are from the same directory as RegAsm.exe. HawkEye dynamically extracts two PE files into its memory, which are then copied into the two newly created child processes of “vbc.exe”. It also modifies its ThreadContext data (It calls the API, SetThreadContext) and makes its entry point to the transfered PE file. When “vbc.exe” resumes running it can be executed. It’s a trick that malware often performs to camouflage itself behind of a normal process. The two “vbc.exe” processes collect credentials from the victim’s system. One is used to collect the credentials of browsers. The other one focuses on email clients and IM clients to steal credentials and profiles. Both PE files injected into “vbc.exe” have the same code framework. They first call a function to collect credentials and save them in memory, and second, it reads the collected data, formats it, and saves it to a tmp file from its command line parameter. Figure 5 shows HawkEye calling the CreateProcess API to start one of the two “vbc.exe” processes, with the parameter shown below in the “Locals” sub-tab. You can see the full path of “vbc.exe”. “/stext ""C:\Users\\*\*\*\*\*\*\*\*\*\AppData\Local\Temp\tmpBE3D.tmp""" is the parameter passed to it. The tmp file name is random and different from the two “vbc.exe” processes, which temporarily saves collected credentials. The two PE files are not packer protected and not .Net written program. The first “vbs.exe” collects credentials from victim’s browsers and the system credential manager for IE. In my analysis, this variant of HawkEye focuses on the following browsers: Microsoft Internet Explorer, Google Chrome, Apple Safari, Opera, Mozilla Sunbird, Mozilla Firefox, Mozilla Portable Thunderbird, Mozilla SeaMonkey, YandexBrowser, Vivaldi browser, and more. Figure 6 shows some strings defined in the ASM code of the browsers that the HawkEye malware wants to collect credentials from. The collected credentials are then saved into the tmp file from its command line parameter. HawkEye\_RegAsm keeps checking this tmp file, and once the credentials are collected, it is done. HawkEye\_RegAsm then reads the entire data of this tmp file into its memory and the deletes it immediately. The second PE file in “vbc.exe”collects profile and credential information of the email and IM software client installed on a victim’s machine. The clients it targets are: Qualcomm Eudora, Mozilla Thunderbird, MS Office Outlook, IncrediMail, Groupmail, MSNMessenger, Yahoo!Pager/Yahoo!Messenger and Windows Mail. Below is an example list that HawkEye stole from the Chrome browser on my test machine. As you can see, it includes login URL, Browser name, User name, Password, Created time, and the full path of the file where the collected information came from. The second PE file in “vbc.exe” not only collects the client’s login username and password, but also profile information, such as the recipent Server address, recipient Server Port, protocol Type (POP3), SMTP Server, SMTP Port, etc. Figure 7 shows a screenshot of Ollydbg when “vbc.exe” was about to write the collected recipient Server addresses into its tmp file. It writes one line once. The same tmp file is finally read by HawkEye\_RegAsm and then deleted. Ok. Now let’s go back to the main process of HawkEye\_RegAsm, which controls all tasks of HawkEye and sends the victim’s credentials. In its main program, it calls Thread.Sleep(600000), and pauses while collecting credentials every 10 minutes. That is, it reports the collected data to attacker the once every 10 minutes. It first sends an HTTP request, *http://bot.whatismyipaddress.com*, to ask for my machine’s public IP. This is a way to ensure that the victim’s machine is able to access the internet. If it did not reply with a public IP, it stops sending collected data to the email box. In addition, the IP appears in the email subject so it can identify victims. The attacker’s email is in *Yandex.mail*, whose email account and password are used when sending collected data through the Yandex SMTP server.

# DustySky Campaign

The attackers would usually send a malicious email message that either links to an archive file (RAR or ZIP compressed) or has one attached to it. Below are malicious email messages that have been sent to multiple targets on September and December 2015. If the victim extracts the archive and clicks the .exe file, the lure document or video are presented while the computer is being infected with DustySky. In recent samples the group used Microsoft Word files embed with a malicious macro, which would infect the victim if enabled. Note, that these infection methods rely on social engineering - convincing the victim to open the file (and enabling content if it is disabled) - and not on software vulnerabilities. The subject line of the malicious email message, as well as the name and content of the lure document, are usually related to recent events in diplomacy, defense, and politics. Sometimes lure topics are gossip or sex related and might even include a pornographic video. In recent samples, fake invoices and a copy of the public Google privacy policy were used. The content of the lure document is always copied from a public news item or other web content, and is never an original composition of the attackers. When linked from the malicious message, the malware would be hosted either on a cloud service (many times in *copy.com*, a legitimate file hosting service), or on a server controlled by the attackers. IP address *45.32.13.169* and all the domains that are pointing to it8 host a webpage which is a copy of a legitimate and unrelated software website - iMazing, an iOS management software. The fake website, similarly to the legitimate one, offers visitors to download the iMazing software. However, the version on the fake website is bundled with DustySky malware. Upon execution of the malicious version (2f452e90c2f9b914543847ba2b431b9a) the legitimate iMazing is installed, while in the background DustySky is dropped as a file named *Plugin.exe* (*1d9612a869ad929bd4dd16131ddb133a*), and executed. *Plugin.exe* immediately starts communicating with its command and control sever using the hardcoded address *ns.suppoit[.]xyz* and *supo.mefound[.]com*, both also pointing to above mentioned *45.32.13.169*.

# TrickLoad Spyware Campaign

This spyware arrives on a system as a file dropped by other malware or as a file downloaded unknowingly by users when visiting malicious sites. It may be dropped by the following malware: TROJ\_UPATRE.YYSTV. This spyware drops the following copies of itself into the affected system and executes them: *%Application Data%\{malware file name}.exe* It drops the following files: *%Application Data%\client\_id*, *%Application Data%\group\_tag*. It uses the Windows Task Scheduler to add a scheduled task that executes the copies it drops. It adds the following processes: *svchost.exe*. It creates the following folders: *%Application Data%\Modules\*, *%Application Data%\Modules\injectDll32\_configs*. It adds the following mutexes to ensure that only one of its copies runs at any one time: Global\TrickBotIt injects codes into the following process(es): This spyware saves the files it downloads using the following names: *%Application Data%\Modules\injectDll32*; *%Application Data%\Modules\systeminfo32*; *%Application Data%\Modules\config.conf* (updated config file); *%Application Data%\Modules\injectDll32\_configs\dinj*; *%Application Data%\Modules\injectDll32\_configs\dpost*; *%Application Data%\Modules\injectDll32\_configs\sinj*. This spyware connects to the following URL(s) to get the affected system's IP address: *rnalip.com*

# Emotet campaign

Reference: <https://www.welivesecurity.com/2018/12/28/analysis-latest-emotet-propagation-campaign/>

As we might expect, if the user decides to download the email attachment and open the document, it asks them to enable the macros. Again, as is usual, some justification for this requirement is provided. Figure 2 shows that in this case it is implied this is necessary because the document was created using Office 365, but really it is so it can execute a function embedded in the file. Clearly, this behavior is already known to be malicious. Effectively, this text box contains a “cmd” command, which launches a PowerShell script that tries to connect to five sites and then download the payload, which in this case is an obfuscated variant of Emotet. As we have discussed in previous posts (for example, in this post from November 9), once the payload is executed, it establishes persistence on the computer and reports its success to its C&C server. Having completed this initial infection, further downloads can occur, installing attack modules and secondary payloads which carry out other kinds of actions on the compromised computer. The various additional modules extend the range of malicious activities that can compromise the user’s device, in order to steal credentials, propagate itself on the network, harvest sensitive information, carry out port forwarding, and many other possibilities.

# Uroburos Campaign

Reference: <https://www.gdatasoftware.com/blog/2014/11/23937-the-uroburos-case-new-sophisticated-rat-identified>

The first task of the malware is to install the file *credprov.tlb* in *%APPDATA%\Microsoft\*. This file is the main payload of the malware. The dropper executes the following command in order to install a second file: *rundll32.exe* %APPDATA%\Microsoft\credprov.tlb,Install *%APPDATA%\Microsoft\shdocvw.tlp*. The second file is *shdocw.tlp*. The two files are Microsoft Windows dynamic libraries. To be started during the boot process of the infected machine, the malware creates the following registry key: *HKCU\Software\Classes\CLSID\{42aedc87-2188-41fd-b9a3-0c966feabec1}\InprocServer32* = *%APPDATA%\shdocvw.tlp*. This registry key is used to associate the library *shdocvw.tlp* to the object 42aedc87-2188-41fd-b9a3-0c966feabec1 as previously explained in the article about COMpfun. The purpose is to load the library into each and every process executed on the infected system. If the version of the malware is older than 3.26, the dropper creates an additional file called *winview.ocx*. We noticed that the file name is still the same as the file name used by *Agent.BTZ* in the past. During the startup of the infected machine, the *shdocvw.tlp* library is loaded into all processes. If the process is *explorer.exe*, this library will load the other library called *credprov.tlb*. This library is the real payload. Its features are common for a Remote Administration Tool (RAT): ComRAT’s communication to the command and control server is performed by the browser process and not by *explorer.exe* in order to avoid being blocked by a firewall on the system or any additional security products. The communication between the processes is performed by named pipe.

1. APT41 Campaign

Reference: <https://www.fireeye.com/blog/threat-research/2020/03/apt41-initiates-global-intrusion-campaign-using-multiple-exploits.html>

On March 5, 2020, researcher Steven Seeley, published an advisory and released proof-of-concept code for a zero-day remote code execution vulnerability in Zoho ManageEngine Desktop Central versions prior to 10.0.474 (*CVE-2020-10189*). Beginning on March 8, FireEye observed APT41 use *91.208.184[.]78* to attempt to exploit the Zoho ManageEngine vulnerability at more than a dozen FireEye customers, which resulted in the compromise of at least five separate customers. FireEye observed two separate variations of how the payloads (*install.bat* and *storesyncsvc.dll*) were deployed. In the first variation the *CVE-2020-10189* exploit was used to directly upload “*logger.zip*”, a simple Java based program, which contained a set of commands to use PowerShell to download and execute *install.bat* and *storesyncsvc.dll*. FireEye observed APT41 leverage the Microsoft BITS Admin command-line tool to download *install.bat* (MD5: *7966c2c546b71e800397a67f942858d0*) from known APT41 infrastructure *66.42.98[.]220* on port 12345.In both variations, the install.bat batch file was used to install persistence for a trial-version of Cobalt Strike BEACON loader named *storesyncsvc.dll* (MD5: *5909983db4d9023e4098e56361c96a6f*). *Storesyncsvc.dll* was a Cobalt Strike BEACON implant (trial-version) which connected to exchange.dumb1[.]com (with a DNS resolution of *74.82.201[.]8*) using a jquery malleable command and control (C2) profile. Within a few hours of initial exploitation, APT41 used the *storescyncsvc.dll* BEACON backdoor to download a secondary backdoor with a different C2 address that uses Microsoft CertUtil, a common TTP that we’ve observed APT41 use in past intrusions, which they then used to download *2.exe* (MD5: *3e856162c36b532925c8226b4ed3481c*). The file *2.exe* was a VMProtected Meterpreter downloader used to download Cobalt Strike BEACON shellcode. The downloaded BEACON shellcode connected to the same C2 server: *91.208.184[.]78*. We believe this is an example of the actor attempting to diversify post-exploitation access to the compromised systems.

# Espionage Campaign

Reference: <https://www.riskiq.com/blog/labs/spear-phishing-turkish-defense-contractors/>

The group used tactics that have become extremely useful for cyber spies—spear phishing emails that social engineer the victim to download an attached or embedded file and then enable macros. These macros contain executable files that download a Remote Access Trojan (RAT), which can log keystrokes, take screenshots, record audio and video from a webcam or microphone, and install and uninstall programs and manage files. While it could, of course, be a fake website, it’s more likely a compromised host as it also contained phishing pages for the dating website Match.com: Normal email for the Gelirler domain would come from the IP specified in the MX record of *gelirler.gov.tr*, which is *212.133.164.130*. Their SPF records, which enforce this process, have been set to "v=spf1 mx -all.” The attachment is an XLS document with the title “*2017-94197 SAYILI GENELGE [DUYURU].xls*.” Opening the document shows a prevalent attack flow: Macros. The macro contains a slightly obfuscated malicious executable file inside. The executable data is stored inside the macro in the form of arrays with integer values spread throughout the macro script. The data from the arrays is combined and written to disk in the Application Data folder. The filename chosen seems to be random for every macro—most likely generated automatically. In the XLS shown above, the PE is written to %appdata%rqco.exe and executed. This file is a small (3kb) loader, which downloads the second stage of the attack. The loader has no imports, but at runtime, resolves the UrlDownloadToFile function from the URLMON library to download stage two, and then ShellExecute from kernel32 to run the downloaded stage two. The stage-two payload downloads from hxxp://unifscon[.]com/R9\_Sys.exe. The C2 server configured for the attack on the defense contractor is civita2.no-ip.biz. Then, a little while after sending out the spear phishing emails, we can see the IP resolution change with, most likely, IP addresses of compromised machines used for SOCKS5 proxying to hide the C2.