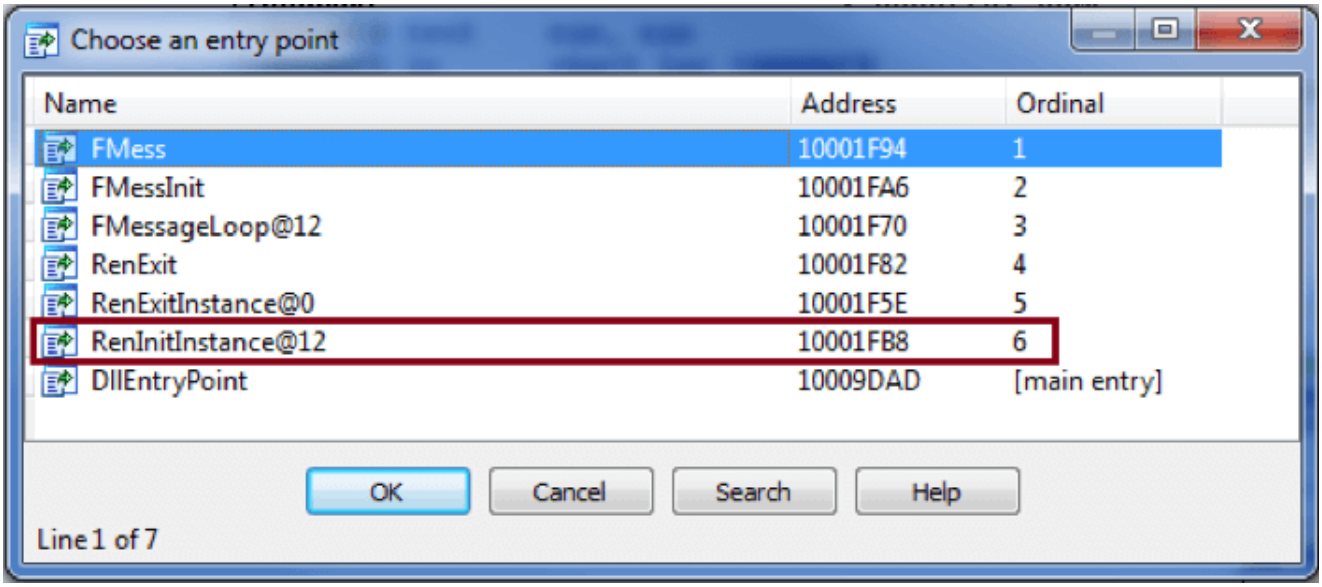
Hussarini – Targeted Cyber Attack in the Philippines

fortinet.com/blog/threat-research/hussarini---targeted-cyber-attack-in-the-philippines.html

July 8, 2018



Two weeks ago, FortiGuard Labs spotted a malicious document with the politically themed file name “Draft PH-US Dialogue on Cyber Security.doc”. This document takes advantage of the vulnerability CVE-2017-11882. Upon successful exploitation, it drops a malware in the victim’s %temp% directory.

Our analysis of this malware shows that it belongs to *Hussarini*, also known as *Sarhust*, a backdoor family that has been used actively in APT attacks targeting countries in the ASEAN region since 2014.

According to reports, the Philippines is the most exposed country in ASEAN to the cyberattacks known as advanced persistent threats, or APTs. After several massive data breaches in 2016, the Philippines started to invest in beefing up their defences against cyberattacks. In spite of these investments, however, the Philippines is still among the most prone countries to be targeted by these sorts of APT attacks.

Exploit Document

Our analysis begins with the exploit document named “Draft PH-US Dialogue on Cyber Security.doc” that takes advantage of the CVE-2017-11882 vulnerability. Upon successful exploitation, the malicious document file drops two files in the %Temp% directory.

- Outllib.dll
- OutExtra.exe

OutExtra.exe is a signed legitimate application from Microsoft named finder.exe. This file is part of the Microsoft Office suite and can be used to find keywords within Outlook data files. However, in this attack, this file is used to load the *Hussarini* backdoor via DLL hijacking.

DLL hijacking is a technique used by some APT malware in which instead of the legitimate application (.exe) loading the benign DLL, the application is tricked into loading a DLL containing malicious code. Using this technique, a malware can evade the Host Intrusion Prevention System (HIPS) of security programs that monitor the behaviors of executed files. Most HIPS tools whitelist signed or trusted files, thereby excluding any malware loaded using DLL hijacking by those signed files from any behavior monitoring. In the context of this attack, OutExtra.exe is a signed legitimate application; however, it is tricked into loading a malware file that is disguised as the legitimate Outllib.dll file.

Decoy Document

To avoid suspicion by the victim, the exploit downloads a decoy document from the legitimate-seeming hxxp://157.52.167.71:29317/office/word/2003/ph2/philip.varilla. During our analysis, though, the download link for the decoy document was already down.

However, taking a look at the URI, we spotted several clues as to where/from whom the hacker possibly wants the user to believe the document came from. First is the “ph2,” which could mean “ph” for Philippines and “2” as its second directory of decoy documents. And second is “philip.varilla”. A quick Google search on this name led us to the Service Director of the Philippines Department of Information and Communications Technology (DICT), who has the same name. DICT is the agency responsible for the planning, development, and promotion of the Philippines’ information and communications technology including cybersecurity.

Hussarini

In fact, the file Outllib.dll is actually the *Hussarini* backdoor, a DLL which exports functions containing the malicious code. When the file OutExtra.exe is executed, some of these functions are called, effectively executing its malicious code.

Fig 1. Hussarini export functions

While the original Outllib.dll also has some of the functions above, notice that this backdoor has a lot more exported functions than just the backdoor DLL.

Fig 2. Export functions of the original Outllib.dll file

There is also a big difference in the file size. The original Outllib.dll has a file size range of 4-8 MB, depending on the version, but the fake file only ranges from 40-50 KB. One of the reasons for this big difference in file size is that only one of the exported functions of the

backdoor DLL contains the malicious code. All of the others are just RETN functions that do nothing.

Thus, we will focus our analysis on the function that contains the backdoor code named *RenInitInstance@12.* This function begins by instantiating a class initializing the setting of the bot. It then creates two concurrent threads taking the created object as its parameter.

Fig 3. Hussarini main function

The first thread acts as a client thread that is responsible for communicating with the command and control (C&C) server and listening for commands. The response is then parsed and passed to the second thread that acts as the worker. The worker thread executes the commands and reports the result to the client thread.

Before communicating with its command and control (C&C), the malware saves a *ServerID* in the registry with a randomly generated value. This ID identifies this bot in the botnet.

Fig 4. Bot ServerID

C&C Communication

Interestingly, there's a private IP address 10.1.0.105 in the code that may have been used as a test C&C server. This IP address is replaced at runtime with the real C&C publicdfaph.publicvm.com.

When communicating with its C&C, Hussarini uses its own custom protocol encoded with base64, which is sent over HTTP. Due to the high-level nature of the code and limited analysis time, we were not able to identify all the fields of the protocol, but knowing some important parts is enough to understand how the communication works.

Fig 5. Hussarini protocol

The initial data it sends contains the generated *ServerID*, the size, and the checksum of the message. This data is encoded with base64 and sent as an argument to the HTTP GET request.

Fig 6. Initial message to the command and control server

The response from the C&C is enclosed with the tag <CHECK></CHECK>. The data in between is also base64 encoded. When decoded, we can see that it follows the same data structure as the initial message.

Fig 7. Initial response from the command and control server

After the check, the malware gets the following system information, which it sends to the C&C via an HTTP POST request:

· User name, Computer Name, OS, and CPU information

Fig 8. Sending of machine information to the C&C

It took a while before we were able to receive commands from the C&C, and when we did we suddenly lost connection to the C&C, which could mean that the hacker controls the C&C manually. The connection disruption could be because the attacker detected that the bot was being analysed and blocked our analysis environment from communicating with the C&C, or just that the machine is not interesting to performing further attacks. The received data was enclosed with the tag <COMMAND></COMMAND>, encoded with base64, and still follows the same data structure of the protocol when decoded. The commands are pretty obvious since its strings are not encrypted after decoding the data.

Fig 9. Receiving commands from the C&C

The first command includes the string “cache.txt”. The worker thread is called with this command to create the following files in the same directory where the malware is running.

· cache.txt

· cache.txt.cfg

The second command contains commands for the Command Prompt (cmd.exe).

Fig 10. Receiving next commands from the C&C

The following *cmd* commands sent from the C&C are then passed to the worker thread and written to cache.txt.

Fig 11. Commands written in cache.txt

The description of the commands written on cache.txt are as follows:

· *systeminfo* – get the systeminfo of the computer

· *arp –a* – view the mapping of IP addresses to MAC addresses

· *ipconfig /all* – display all current TCP/IP network configuration values

· *netstat -ano* - displays protocols statistics and current TCP/IP network connections

· *tasklist -v* – list of apps and services with their Process ID (PID) for all task running

· *net start* – starts any of the various services that are running

· *net view* – displays the other computers that are visible on the network

· *dir "c:\users" /o-d* – displays all users

After sending these *cmd* commands, the C&C stopped responding. It could be that the hacker decided to block the connection after seeing the result of these commands.

Based on its code, this malware is capable of executing the following commands from the hacker:

- Create, read, write files
- Download and execute files/components
- Launch remote shell using cmd.exe

While other APT backdoors have more capabilities, such as keylogging, taking screen shots, etc., Hussarini only has few, including the capability to download and execute files/components. However, its functionality can be extended by the hacker. Also, since this backdoor can launch a remote cmd shell, all cmd commands are available for the attacker to use, such as those seen in Figure 11.

Hussarini uses a dynamic domain to maintain anonymity and to also possibly be able to change the C&C server IP address. At the time of this analysis, however, we only saw it resolved to the IP address 157.52.167.71, which is the same IP used to host the decoy document.

Fig 12. C&C domain name resolution

Final thoughts

We are unsure how the document is being distributed; however, taking a look at the C&C domain name *publicdfaph.publicvm.com* and considering the file name of the exploit document “Draft PH-US Dialogue on Cyber Security.doc” and the clues from the decoy document download link hxxp://157.52.167.71:29317/office/word/2003/ph2/philip.varilla, there is one scenario that seems to make the most sense.

It seems likely that the names used were crafted to make it look like the document came from the Department of Information and Communications Technology (DICT), or possibly from the Service Director, which would imply that the employees of the Department of Foreign Affairs (DFA) are the target. Clearly, the use of *publicdfaph* in the C&C domain name is used to camouflage the traffic and to trick a network administrator into thinking that this domain is under DFA so as not to raise any suspicion.

Conclusion

Hussarini was first mentioned in APT campaigns targeting the Philippines and Thailand in 2014. Today, this malware is still actively being used against the Philippines. The Department of Information and Communications Technology was only formed in 2016 and has acknowledged that the Philippines’ state of cybersecurity is still in infancy. In general, this

contributes to the Philippines continuing to be a target for cybercrime/cyberespionage or even state-sponsored attacks. Because of this, we expect that attacks targeting that region will continue to evolve in both quantity and quality. As always, we here at FortiGuard Labs will continue to monitor events in order to hunt for, catch, and block these attacks.

Solution:

Fortinet detects the exploit document as MSOffice/CVE_2017_11882.A!tr and the Hussarini samples as W32/Sarhust.D!tr. Malicious URLs related to this malware are also blocked by FortiGuard Web Filtering Service.

-= FortiGuard Lion Team =-

IOCs:

SHA256:

154261a4aab73f1ceef28695d8837902cc1e8b5cca0b8fc81ddeda350564adc0 - MSOffice/CVE_2017_11882.A!tr

05dcc7856661244d082daa88a074d2f266c70623789a7bb5a919282b178d8f98 - W32/Sarhust.D!tr

CC:

hxxp://157.52.167.71:29317/office/word/2003/ph2/philip.varilla

hxxp://publicdfaph.publicvm.com:8080/

Check out our latest Quarterly Threat Landscape Report for more details about recent threats.

Sign up for our weekly FortiGuard Threat Brief or for our FortiGuard Threat Intelligence Service.