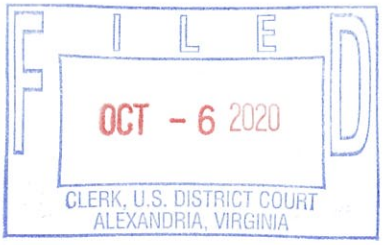
IN THE UNITED STATES DISTRICT COURT
FOR THE EASTERN DISTRICT OF VIRGINIA
Alexandria Division



MICROSOFT CORPORATION, a Washington corporation, and FS-ISAC, INC., a Delaware corporation,

Plaintiffs,

v.

JOHN DOES 1-2, CONTROLLING A COMPUTER BOTNET AND THEREBY INJURING PLAINTIFFS, AND THEIR CUSTOMERS AND MEMBERS,

Defendants.

Civil Action No: 1:20 CV 1171

FILED UNDER SEAL

DECLARATION OF VIKRAM THAKUR IN SUPPORT OF PLAINTIFFS' APPLICATION FOR AN EMERGENCY *EX PARTE* TEMPORARY RESTRAINING ORDER AND ORDER TO SHOW CAUSE RE PRELIMINARY INJUNCTION

I, Vikram Thakur, declare as follows:

1. I am a Technical Director with the Symantec Enterprise Division of Broadcom Inc. ("Symantec"). I make this declaration in support of Plaintiffs' Application For An Emergency Temporary Restraining Order And Order To Show Cause Re Preliminary Injunction. I make this declaration of my own personal knowledge, and, if called as a witness, I could and would testify competently to the truth of the matters set forth herein.

2. In my role at Symantec, I am a part of a global team investigating incidents related to online attacks, security threats, botnets and fraud. In particular, over the past 14 years I have been involved in identifying and mitigating online threats for millions of Symantec product end users. My role at Symantec has provided me an in-depth insight into how malware authors deploy and utilize online threats for their monetary gain. Prior to joining Symantec, I was a System and Network Administrator at Florida State University ("FSU"). At FSU my primary responsibilities were to implement and monitor the availability and security of the

hybrid network of computing devices, operating systems and security appliances. I studied Computer Science at FSU, and got a Masters degree in the same. I am a regular contributor to industry security organizations as well as a contributor to United States Federal Government working groups in this area. A true and correct copy of the current version of my curricula vitae is attached to this declaration as **Exhibit 1.**

I. OVERVIEW OF INTERNET BOTNETS

3. A botnet is a group of compromised end-user computers, all controlled by malicious actors or organizations, without the knowledge or consent of the computer's owner or user. Botnets can be comprised of hundreds or thousands of these compromised computers owned by individuals and businesses. Some botnets have pulled in hundreds of thousands of infected end-user computers, and some comprise over a million computers.

4. Cybercriminals secretly control these botnets in order to conduct illegal acts. Cybercriminals can tell their botnet armies to, for example, install keystroke-logging programs, which will then report the end-users' sensitive information, such as banking passwords or credit card numbers. Research indicates that much of the stolen data bought and sold in the underground economy is directly attributable to botnet operators. Often, botnet operators will rent access to the botnet for specific tasks and may even sell off sections of the botnet, thus enabling these botnets to operate as a "rental-as-a-service." For these reasons, botnets are dangerous instrumentalities that pose a threat to computer users worldwide.

5. Cybercriminals also use botnets to engage in coordinated attacks against governments, private corporations, and the public generally. For example, in 2007, a large-scale botnet operation conducted a denial-of-service attack which infiltrated and shut down major portions of the Internet in Estonia. Similarly, the government of Georgia was attacked by botnets in 2008, resulting in several official government websites being taken down. A botnet called the "Rustock" was responsible for sending many billions of spam emails. Financial-fraud botnets such as "Zeus," "Citadel," "Shylock," "Ramnit" and the botnet in question in this case,

"Trickbot," have been collectively responsible for the theft of millions—and potentially even hundreds of millions—of dollars from online banking accounts.

6. Botnets are grown by infecting multiple computers with malicious software and connecting them to a common command and control (C&C) infrastructure over the Internet, through which they may all be directed. Infection may occur through a variety of means, the most common being misleading victims into installing the software, vulnerability exploitation, and leveraging pre-existing infections on a victim's computer. To facilitate control of a botnet, botnets have means for the controller to issue commands, retrieve stolen information, conduct reconnaissance into the networks of individual botnet nodes, and more. These means are commonly referred to as the "command and control", or "C&C", structure of the botnet, and those structures vary in implementation.

7.

8. This declaration describes the functionality, operation and injury caused by a botnet known as the "Trickbot" botnet.

II. OVERVIEW AND HISTORY OF TRICKBOT

9. I and a team of investigators have investigated the Trickbot botnet malware payload. During the investigation, we reverse engineered the Trickbot software to understand its internal operation and structure. While Trickbot is a multi-threat botnet, it primarily serves to distribute additional malware and steal account credentials and then money from the users of infected computers. These stolen credentials have also been long suspected of being used for deployment of ransomware. Components of this threat are primarily detected by Symantec products as Trojan.Trickybot and SONAR.Trickbot.

10. Since its discovery, Trickbot has spread prolifically across the Internet. In fact, my review of representative statistics gathered by Symantec over the past year confirms that Trickbot was one of the most commonly blocked types of malware on the Internet. While these statistics report instances in which Trickbot was successfully blocked, the data reflects a high

rate of attempted infections, which generally will correspond to a high rate of successful infection of inadequately protected computers.

11. Symantec has studied Trickbot since its inception and determined that Trickbot is modular in design, and able to carry out a wide range of illegal activity, including financial account theft, ransomware attacks, attacks on other computers and distribution of other forms of malware.

III. TRICKBOT'S TECHNICAL DETAILS

How Trickbot Is Propagated

12. Trickbot is typically distributed through spam or phishing email campaigns containing a malicious attachment appearing as a Microsoft Word document or by tricking the victim to click on a malicious URL to download a document. Once the malicious Word document is opened, a macro (automated task using features of the computer) will execute either directly to download the next stage of Trickbot's main component or it will execute other commands which eventually download this main component. In some cases, I have observed Trickbot downloading completely different malicious software, particularly ransomware.

13. Once on a computer, Trickbot has been observed as spreading across the home user or organization's private network by leveraging publicly documented weaknesses in software.

Modularized Design And How Trickbot Infects End-User Computers

14. Trickbot is a highly module-based malware, extending its functionalities by downloading and executing several modules after acquiring them from the command and control infrastructure depending on the intended malicious attack. During our investigation we have witnessed an infected device that we purposely infected in a controlled environment that downloads different modules at different points in time.

15. Trickbot's initial component, referred to as its 'loader', contains within itself a list of Internet addresses from where to download the 'main' component. This information is

typically encrypted. However, the initial payload will unpack and decrypt this embedded configuration and acquire the main component from these pre-configured and attacker-controlled servers.

16. Trickbot's main component then prepares its framework for feature specific modules and initiates a live connection to its pre-configured C&Cs. This main component is responsible for downloading the various modules. Its task is also assisting the feature modules to communicate with their specific C&Cs.

17. Another initial task of Trickbot is to check the architecture of the victim computer, and save it along with the bot's own information within files named '*settings.ini*' or '*compatibility.ini*' based on the system information.

18. Once these initial tasks are completed, the initial payload module enumerates the decrypted C&C list to select a live Internet address from which to download and run the corresponding modules. The downloaded modules are either in 32-bit or 64-bit architecture based on the end user's computer.

19. Trickbot's functionality can be split into two major parts – the 'main module', and the modules downloaded from C&Cs that are launched by the main module, including:

a. injectDll - For injecting into banking websites to steal credentials
b. networkDll – A reconnaissance module for gathering system information and network/domain topology to determine whether the victim device can be infected with ransomware
c. Systeminfo - For gathering basic information on the host
d. importDll - For stealing data from a browser
e. Pwgrab - For grabbing passwords from various spots
f. cookiesDll - For stealing/grabbing cookies
g. mailsearcher - For traversal over all files in all drives in the system looking for something specific
h. NewBCtestDll - Backconnect SOCK5 module
i. psfin - Point-of-Sale 'recon' module
j. vncDll - Remote control/VNC module
k. tabDll - For spreading over SMB using EternalRomance and MS17-010
l. outlookDll - For stealing data saved by Microsoft Outlook
m. shareDll - For lateral movement/enumeration via LDAP and SMB exploitation;
n. wormDll - For lateral movement/enumeration module via LDAP and SMB

exploitation. The share and worm modules work in cooperation.

o. rdpScanDll - New module that uses brute-force remote desktop protocol (RDP) for a specific list of victims

20. It is observed that there are four export functions commonly used in Trickbot modules:

a. Start - initialize the prerequisite information like C2 or URL
b. Control - execute the main function
c. FreeBuffer - release the heap
d. Release - delete the object

The "Initial Module"

21. Trickbot's initial module first copies itself to specific folders on the victim's computer. These folders can be represented as "%appdata%" or "%ProgramData%" folders. Trickbot is able to maintain its presence on victim computers across reboots by copying itself into these folders and creating a scheduled task, named "CleanMemoryWinTask" as an example, that executes these files every 9 minutes. Also, by registering itself as a scheduled task, Trickbot is able to execute even when the computer is powered off and on. After finishing these tasks, it will exit itself and wait for the task scheduler to launch it from the copied destination folder.

22. If Trickbot is launched in the correct location, which means in "%appdata%" or "%ProgramData%", then it will start its main functionality, such as connecting to C&Cs and downloading and launching other modules.

23. The initial module is designed to evade execution within controlled environments, a common sandboxing tool used within the computer security industry to identify functionality of unknown files. To detect the presence of being executed within such a sandbox environment, Trickbot looks for the presence of the following DLL[1] files, and would exit if any of them were found:

a. pstorec.dll
b. vmcheck.dll
c. dbghelp.dll

[1] DLL stands for dynamic-link library that contains code and data that is used by many programs at once. DLLs promote modular architecture and ease of deployment and installation.

d. wpespy.dll
e. api_log.dll
f. SbieDll.dll
g. SxIn.dll
h. dir_watch.dll
i. Sf2.dll
j. cmdvrt32.dll
k. snxhk.dll

24. After checking for the sandbox environment, the initial payload will check the current user's privileges. If it is running under low privilege, then it auto elevates the process using User Access Control (UAC) bypass, a technique to execute programs with elevated privileges without the end user being prompted. Once the Trickbot initial payload process gains high privilege, then it will start to check if any security products are installed and running on the victim system. It attempts to stop these security products using various commands such as:

a. /c net stop SAVService
b. /c net stop SAVAdminService
c. /c net stop Sophos AutoUpdate Service
d. /c net stop SophosDataRecorderService
e. /c net stop Sophos MCS Agent
f. /c net stop Sophos MCS Client
g. /c net stop sophossps
h. /c net stop Sntp Service
i. /c net stop Sophos Web Control Service
j. /c net stop swi_service
k. /c net stop swi_update_64
l. /c sc stop SAVService
m. /c sc delete SAVService
n. It stops Windows defender service:
o. cmd.exe /c sc stop WinDefend
p. cmd.exe /c sc delete WinDefend

25. It also attempts to kill anti-virus (AV) related processes named:

a. MBAMService
b. SAVService
c. SavService.exe
d. ALMon.exe
e. SophosFS.exe
f. ALsvc.exe
g. Clean.exe
h. SAVAdminService.exe

i. SavService.exe
j. ALMon.exe

26. After running these environment checks, the initial payload will either decrypt C&C configuration data from itself or directly load the updated C&C configuration data from saved local file settings.ini. The configuration data is in XML format. The root element is always *<mccconf>* and have the following nodes:

a. <ver> element denotes the current version of the config file
b. <gtag> refers the group tag which indicates the group begin it
c. <servs> element contains a list of IP addresses and tagged with <serv>
d. <autorun> element gives the name of default modules to be downloaded from C&C
e. <module> gives the module names
f. ctl refers to the command to be passed to the module

27. Most of the Trickbot communications with C&C servers use HTTPS GET type connections. HTTP/HTTPS POST requests are used to send back some gathering information. The C&C addresses are always numeric IP addresses and the port numbers used are usually 80, 8082, 443, 8443, 446, 449, or 499. The Trickbot initial payload supports various commands that it uses to communicate with the C&C servers. The various commands have the following pattern:

```
{server-ip:port}/{gtag}/{client_id}/{Command}/{other payload info}/
```

Table 1: Some Commands Used In Trickbot C2 Communication

COMMAND	DESCRIPTION	URL
0	Register comprised client	/0/
1	Keep alive command	/1/{random_str}
5	Download module or config	/5/{module}|{config}
10	Log information	/10/{command_id}/{random_str}/{0|1|/
14	Module execution result	/14/{random_str}/{result}/0/
23	Update base config	/23/{random_str}
25	Update bot	/25/{random_str}
60	web request/response for injectDll	/60/{random_str}
63	POST data for module systeminfo or injectDll	/63/{module_name}/{module_command}/ {result base64 string}/{result_code}

28. Once a module is downloaded, Trickbot's initial payload will check if any

configuration data for the module needs to be download. If so, the initial payload tries to download the specified configuration data and parse it. Then the initial payload launches a new '*svchost.exe*' process in suspended mode and injects the entire module into the process after allocating virtual memory in the remote process. It resolves the import table and overwrites each entry using *WriteProcessMemory*. It also fixes the relocation table. It finally writes an instruction at the entry point of the *svchost.exe* process to point it to the module entry point. After the module has been injected, it writes shell code into *'svchost.exe'* memory. This shell code serves as a communication mechanism between the initial payload and the module. The address of the shell code is passed as an argument to the "*Start*" export function. Finally, the suspended *svchost* process resumes and the module is considered launched. All downloaded modules are encrypted and saved as files on the victim computer's disk.

Module "injectDll": Trickbot's Main Banking and Browser Hijacking Module

29. *injectDll* is the main banking module that is designed to infiltrate a victim's device and exfiltrate banking credentials. Trickbot's initial payload downloads this module and its configuration data as '*dinj*', '*sinj*' and '*dpost*', then launches it using *svchost.exe*. When *injectDll* is running, it targets various browsers by intercepting the network traffic between the victim computer and the legitimate remote server. The targeted browser processes include:

a. chrome.exe
b. iexplorer.exe
c. firefox.exe
d. microsoftedgecp.exe

30. When any of the targeted browser processes are encountered by Trickbot, *injectDll* extracts its embedded DLL and injects it into the running browser process. Once the DLL is injected into the browser process, some of its functions are considered 'patched' or 'intercepted' by the injected DLL. From Symantec's analysis, the hooked functions are listed as the following:

a. iexplorer.exe

i. WININET.DLL!HttpSendRequestA
ii. WININET.DLL!HttpSendRequestW
iii. WININET.DLL!HttpSendRequestExA
iv. WININET.DLL!HttpSendRequestExW
v. WININET.DLL!InternetCloseHandle
vi. WININET.DLL!InternetReadFile
vii. WININET.DLL!InternetReadFileExA
viii. WININET.DLL!InternetQueryDataAvailable
ix. WININET.DLL!HttpQueryInfoA
x. WININET.DLL!InternetWriteFile
xi. WININET.DLL!HttpEndRequestA
xii. WININET.DLL!HttpEndRequestW
xiii. WININET.DLL!InternetQueryOptionA
xiv. WININET.DLL!InternetQueryOptionW
xv. WININET.DLL!InternetSetOptionA
xvi. WININET.DLL!InternetSetOptionW
xvii. WININET.DLL!HttpOpenRequestA
xviii. WININET.DLL!HttpOpenRequestW
xix. WININET.DLL!InternetConnectA
xx. WININET.DLL!InternetConnectW

b. firefox.exe

i. nss3.dll!PR_OpenTCPSocket
ii. nss3.dll!PR_Connect
iii. nss3.dll!PR_Close
iv. nss3.dll!PR_Write
v. nss3.dll!PR_Read

c. chrome.exe

i. It looks for various signatures inside chrome.dll in memory. If a match found, it hooks those APIs. These APIs don't have names, so this module hooks them by address.

31. *InjectDll* needs three configuration data files to execute. Each of the three configuration files- *sinj*, *dinj* and *dpost* are XML files and used for different purposes. Records in *sinj* are similar to the follow format:

```
<sinj>
<mm>https://retail.santander.co.uk*</mm>
<sm>https://retail.santander.co.uk/LOGSUK_NS_ENS/BtoChannelDriver.ssob
to*</sm>
<nh>odsakrjtsmyalzxfdpvbgqieowch.com</nh>
<url404></url404>
<srv>198.46.190.28:443</srv>
```

```
</sinj>
```

32. Each record begins with the tag '*sinj*'. The '*mm*' field will match any URL (website address) starting with the field value. The '*sm*' field is more specific for the URL that matches the record. For each matched URL, the matched URL's hostname will be replaced with '*nh*' value. The exactly matched URL request will be redirected to the C&C address in the '*srv*' field and the response from this attacker server is then transferred back to the victim browser and rendered.

33. Records inside '*dinj*' are started with the tag '*igroup*'. Each '*igroup*' tag has multiple '*dinj*' records. One of the '*igroup*' record for example is as the following:

```
<igroup>
<dinj>
<lm>*.com/pub/html/login.html*</lm>
<hl>http://37.44.215.203/response.php</hl>
<pri>100</pri>
<sq>2</sq>
</dinj>
<dinj>
<lm>*.com/pub/html/favicon.ico*</lm>
<hl>http://37.44.215.203/response.php</hl>
<pri>100</pri>
<sq>2</sq>
</dinj>
<dinj>
<lm>*favicon.ico=843729ac35951a040681c469b4a89c0b*</lm>
<hl>http://37.44.215.203/response.php</hl>
<pri>100</pri>
<sq>1</sq>
</dinj>
</igroup>
```

34. The '*lm*' field will do a wildcard match with any HTTP request URL with the field value, then the response will be sent to the '*hl*' field specified in the record then transferred back to the browser.

35. The '*dpost*' configuration file contains a list of URLs for handling any post request. The post data will be sent to one of the handlers in the '*dpost*' xml data. The data in '*dpost*' is like this:

```
<dpost>
<handler>http://186.159.1.217:8082</handler>
<handler>http://186.10.243.70:8082</handler>
<handler>http://75.183.130.158:8082</handler>
<handler>http://186.183.151.194:8082</handler>
<handler>http://181.129.160.10:8082</handler>
<handler>http://181.57.97.138:80</handler>
<handler>http://200.21.51.30:80</handler>
<handler>http://191.103.252.29:80</handler>
<handler>http://200.35.47.199:80</handler>
<handler>http://190.152.125.162:80</handler>
<handler>http://79.137.119.209:443</handler>
<handler>http://216.189.145.231:443</handler>
<handler>http://194.5.250.130:443</handler>
<handler>http://192.210.152.190:443</handler>
<handler>http://195.123.240.31:443</handler>
<handler>http://89.46.223.252:443</handler>
</dpost>
```

36. When *injectDll* is hosted by *svchost.exe* and the embedded component is injected into the running browser, Trickbot interprets any HTTP request made by browser and response returned before rendering contents in the browser. Trickbot checks the requested URL and matches it with the '*sinj*' rules and '*dinj*' rules. If any rule is hit, the request and response data will be sent to the rule's C&C. Then the altered response is sent back to the browser and the browser renders the altered data. As a result, if the victim computer is used to make a bank transaction – like the login credential, bank account or payment amount – the legitimate website's content shown to the end user will be changed underneath without the user's notice. During my research, when visiting some bank pages on a Trickbot compromised computer and browser, the original response from the financial institution was modified by inserting a malicious '*<script>*' tag into the original HTTP response.

Module networkDll

37. *NetworkDll* gathers even more system information, like serial number, organization, usernames, domain name and uses tools like net, ipconfig and nltest to list all network adapters, domain controllers, domains and shares. After gathering all the information, networkDll sends the gathered information to the initial payload. The initial payload will then send back all these data to a C&C.

38. It collects the following data from WMI interface using the query "SELECT * FROM Win32_OperatingSystem":

a. CSName
b. Caption
c. CSDVersion
d. OSArchitecture
e. ProductType
f. BuildType
g. WindowsDirectory
h. SystemDirectory
i. BootDevice
j. SerialNumber
k. InstallDate
l. LastBootUpTime
m. RegisteredUser
n. Organization
o. TotalVisibleMemorySize
p. FreePhysicalMemory

39. It also finds out what product type the machine is – Workstation, Domain controller or Server.

40. It executes the following commands and sends the result, read from the command line, to the initial payload:

a. cmd /c ipconfig /all
b. cmd /c net config workstation
c. cmd /c net view /all
d. cmd /c net view /all /domain
e. cmd /c nltest /domain_trusts /all_trusts

41. It queries Active Directory (AD) system information object and gets the following data, which it then sends to the initial payload:

a. User Name
b. Computer Name
c. Site Name
d. Domain Shortname
e. Forest Name
f. Domain Controller
g. Forest Trees

42. It enumerates LDAP information and updates the initial payload.

43. Finally, it sends a status message to the C&C server with IP address pair in the dpost config file.

Module Systeminfo

44. *Systeminfo64* is a system and network information gathering tool. It gathers basic information about hosts, like operating system (OS), users, installed programs and services information:

a. OS information queried using WQL: SELECT * FROM Win32_OperatingSystem, then Caption, OSArchitecture, CSDVersion
b. Users information queried using API NetUserEnum
c. Programs information queried using registry: HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall

45. Services information queried in registry:

HKEY_LOCAL_MACHINE\CurrentControlSet\Services

Module importDll

46. This module steals data like web history, local storage, cookies from browsers such as Chrome, Firefox, Internet Explorer and Edge. After collecting data from each browser, it stores the information inside the '*config.conf*' file and notifies the parent bot process. The initial payload then sends the data to the C&C. It repeats this process for all four browsers. All browser names are hardcoded. The *config.conf* file is deleted after sending the data to the attacker's C&C

Module pwgrab

47. *pwgrab* is a credential stealing module. It steals credentials from Chrome and Internet Explorer's password storage, various RDP and SSH related services, and password managers. Once the module executes, it collects relevant credentials by copying the contents of program files, reading registry keys, and making specific API calls (depending on what is installed on the host). Once collected, these credentials are encoded and exfiltrated to a command and control server via a series of POST requests.

Module cookiesDll

48. The cookies module reads cookie files from the following locations:

a. %USERPROFILE%\AppData\Roaming\Microsoft\Windows\Cookies
b. %USERPROFILE%\AppData\Local\Microsoft\Windows\INetCookies
c. %USERPROFILE%\AppData\Local\Microsoft\Windows\INetCookies\Low

49. It queries Firefox cookies from sqlite database:

a. %USERPROFILE%\AppData\Roaming\Mozilla\Firefox\Profiles*\cookies.sqlite
b. select name, value, host, path, creationTime, expiry from moz_cookies where expiry > strftime('%s', 'now')

50. It queries Chrome cookies from sqlite db:

a. %USERPROFILE%\AppData\Local\Google\Chrome\User Data\Default\Cookies
b. select name, encrypted_value, host_key, path, length(encrypted_value), creation_utc, expires_utc from cookies where datetime (expires_utc/1000000 + strftime('%s', '1601-01-01'), 'unixepoch') > datetime('now', 'utc')

Module mailsearcher

51. This module collects emails from all files in the system. It also uses a separate config file downloaded from the C&C: Modules\mainsearcher32_configs\mailconf. The decrypted '*mailconf*' file is like the following:

```
<mail>
<handler>188.255.249.27:443</handler>
</mail>
```

52. The mailseacher module is an encrypted DLL and injected into a newly created svchost process. Its main function is to search for emails in files in all drives and accumulate them. Then it sends the harvested emails to the server.

Module NewBCtestDll

53. This module acts as a reverse shell. It randomly picks one of the C&C addresses from the configuration list to connect to. If the desired data is received, it will start cmd.exe and create pipes to handle the input/output to the cmd.exe process. the received data then will be passed to the input pipe to cmd.exe. the output from cmd.exe will be read through the pipe and sent back to the server.

Module psfin

54. *Psfin* is a Point-Of-Sale reconnaissance module. Psfin64 will identify lucrative point-of-sale targets like stores, kiosks and payment terminals and steal financial details from

these high-end targets.

Module vncDll

55. *vncDll* module is a remote access tool. When launched by Trickbot's initial payload via *svchost.exe*, it allows Trickbot to capture and interact with the user's desktop. The "Start" export function initiates the VNC activities. In one example, when performing analysis of this component, I found that it tried to connect to the following IP address: 195[.]211[.]162[.]186. If the connection was successful, then it would start the VNC activity and keeps running until it received a "Release" message from the server. When VNC activity is being performed, the *vncDll* module creates a desktop called "*AlterDesk01*" and draws the desktop items. It can also enumerate child windows and send the data to the connected C&C server periodically. It adds many menu items and its corresponding tasks will be performed when invoked by remote.

56. The added menu items are:

a. Explorer - opens "c:\"
b. Internet Explorer - opens iexplore /home
c. Edge - opens Edge
d. Chrome - opens chrome
e. Firefox - open Firefox
f. Outlook Office
g. Windows Scheduler (MMC) - opens "c:\windows\system32\mmc.exe taskschd.msc"
h. Windows Scheduler (Console) - opens "c:\windows\system32\cmd.exe /K schtasks.exe |more"

Module tabDll

57. *tabDll* exploits a vulnerability known as CVE-2017-0144, also used by the infamous WannaCry ransomware, to spread itself to unpatched devices. This module has the following sub-modules: *screenlocker*, *mimikatz*, *SseExecuter*. When launched by Trickbot's initial payload, it will do the following tasks:

a. Set the "UseLogonCredential" value to 1 in the following registry: HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\SecurityProviders\WDigest
b. Then it checks for OS version. If OS version is less than that of windows 7, it will inject *screenlocker* module into explorer.exe.

c. It decrypts *mimikatz dll* in memory and loads into the current process. It then calls the "*getUserPasswordPairs*" export function.
d. Once *mimikatz* has been deployed, it downloads files from two URLs hxxp://185[.]228[.]233[.]107/toler.png and hxxp://185[.]228[.]233[.]107/table.png. Both files are Trickbot executable files packed differently
e. It then launches the *eternalromance* exploit against the network. Once it is successful, it uses the SseExecuter.exe in memory as the payload. The argument to the exe is the downloaded Trickbot binary from above URL. "*EternalRomance*" is built from scratch and found similar to publicly available python implementation.

58. *Screenlocker* module - The purpose of this module is to simply lock the screen.

59. *Mimikatz* module – The publicly available *mimikatz* module has been compiled into a DLL and encrypted inside the *tabDll32*. When *tabDll32* is loaded by Trickbot's main module, it decrypts Mimikatz and loads into the *tabDll32* loaded process and calls the export function "getUserPasswordPairs".

60. *SseExecuter* module - This module launches the executable passed as the arguments and creates a startup entry for it. The executable is stored in the %APPDATA% directory, and a startup entry is created in the registry as a run entry. It also deletes the service created by the exploit.

Module outlookDll

61. This module collects Microsoft Outlook credentials from the system. This module collects Outlook credentials from the system:

a. HKCU\Software\Microsoft\Windows NT\CurrentVersion\Windows Messaging Subsystem\Profiles\Outlook
b. HKCU\Software\Microsoft\Office\15.0\Outlook\Profiles\Outlook\
c. HKCU\Software\Microsoft\Office\16.0\Outlook\Profiles\Outlook\

62. The information it collects including:

a. Email
b. User
c. Password
d. Server
e. Port

Module shareDll

63. This module will download Trickbot's initial payload, then transfer over the downloaded content to the $ADMIN share of a remote computer and create persistence.

64. It downloads the latest Trickbot binary from the IP address, like *89[.]223[.]28[.]172/toler[.]png*, and it saves the file as '*loged.tmp*'. Once it finds the network shares by enumerating local networks, it copies the downloaded file into network share as follows:

a. \\computer_name\c$\stsvc.exe (or)
b. \\computer_name\ADMIN$\stsvc.exe

65. It then creates a service named '*SystemTypeSvc*' and display name as '*TechnicalSvc*' and starts the service. The SUCCESS or FAIL message will be sent to the initial payload depending on the outcome of network compromise as '*WormShare*'.

Module wormDll

66. *wormDll* has worm-like capabilities, just like *tabDll64*. It abuses protocols such as SMB and LDAP to spread itself across the network. For every machine found, it translates the name into an IP address and launches '*eternalblue*' attack at the victim IP addresses with a URL passing to the exploit. The following example URL was found passing to the exploit *http://duhasti8[.]beget[.]tech/worming[.]png*.

Module rdpScanDll

67. On or about February 1, 2020, Symantec analyzed the new Remote Desktop Protocol (RDP) scanning module *rdpScanDll* delivered by Trickbot. It commanded the infected computers to scan the network to gain access to vulnerable systems. The module implements a full RDP connection sequence by using ws2_32 socket functions. All SSL related work when doing the RDP scanning will be done by the API *InitSecurityInterfaceA*. *InitSecurityInterfaceA* provides SSL context, message encrypt and decrypt.

68. The C&C commands found to control the RDP scan module are:

a. rdp/mode - Determines RDP scanning mode. The C&C will return string "brute",

"trybrute" or "check". The "brute" and "trybrute" modes use the same scanning logic, so there is no difference between them, whereas the "check" mode later will query the C&C again to get a pre-configured list for RDP scanning.

b. rdp/freq - When querying the C&C, it will return an integer string to indicate the scan frequency. This parameter will control the resources used by this RDP scan module.
c. rdp/over - After scanning is completed, the module will report the scan result to the C&C.
d. rdp/dict - Get RDP brute force dictionary. If data is in gzip format, then it decompresses the data. This dictionary will be served as the RDP password list.
e. rdp/names – C&C command to get the names dictionary. The names dictionary will be served as the RDP username list.

69. The C&C "rdp/mode" request will issue three scan commands to the client which are "Brute", "TryBrute" and "Check". During the scanning process, the module will periodically notify the initial payload with the status message like "0 addresses tried, 0 RDP hosts detected, 0 passwords found".

a. Brute - When we analyzed the "brute" scanning mode, we found that there was an implementation flaw when performing the "brute" scanning. A "rdp/dict" will be issued to fetch a word list containing the passwords used for the brute scanning. However, there was no username/password pair used when sending the RDP packets.
b. TryBrute - The "trybrute" scanning mode will first issue "rdp/dict" command to download a word list containing the password. Then a "rdp/names" command will be issued subsequently to get a name list containing the username list.
c. Check - When doing a "check" scan task, the client will request "rdp/domains" to get a list which contains rdp host, domain, login username and password pairs. After the list is parsed, the paused scanning thread will be resumed. Each time, the scanning thread will fetch one record for the list and perform an RDP connection with the server. If any host and login credential combination succeed, then one record like "rdp|10.xxx.xxx.xxx:3389|username|password|online rdp not in domain" will be saved and later posted back to the C&C.

IV. INJURY CAUSED BY TRICKBOT

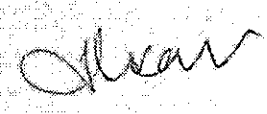
70. Trickbot causes harm to a number of parties. First, of course, serious harm is caused to users whose computers are infected with Trickbot. Through the Trickbot malware infecting the user's computer, Defendants can steal sensitive credentials and identifying information from the victim, and Defendants can then use those credentials to steal money from the user's financial accounts. Additionally, Trickbot makes a number of damaging changes to

the victim's computer. It lowers the security settings, thus leaving the user's computer vulnerable to additional infections from the Internet.

71. Additionally, users who detect that their computer is infected with Trickbot face the often daunting and time-consuming task of figuring out how to remove it from their computers and restore their computer's settings, including its security defenses, to the original and proper configuration. In my experience, users faced with this sort of aggravating situation are prone to extreme frustration, while some would burden the cost of local technical help, or even resort to purchasing new computers.

72. In addition to the users who are harmed, the financial institutions that Trickbot targets are defrauded and lose a considerable amount of money to Trickbot and other financial fraud botnets. In addition, these institutions are required to invest a considerable amount of time, money, and effort into monitoring their systems for Trickbot-perpetrated fraud.

I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct to the best of my knowledge. Executed this 4th day of October, 2020, in Los Angeles, California.



Vikram Thakur

EXHIBIT 1

Vikram Thakur

Qualifications

Over 15 years of security industry experience building and leading global threat intelligence teams. Proven command of security threat landscape including incident handling, comprehensive threat analysis, and remediation strategy development. History of maintaining productive relationships with counterparts across the anti-malware industry and governments. Leader, manager, hunter, networker, collaborator, public speaker, and go-getter. The person that gets the job done.

Competencies

Cyber Threat Intelligence
Malware Research and Analysis
Project Leadership
Personnel Management
Policies and Procedures Development
Data Collection
Process and Efficacy Improvement
Malware Detection Technologies
Incident Handling and Response
Networking
Law Enforcement and Investigations
Inter-Organization Collaboration
Public Speaking

Building and Leading Globally Distributed Teams

Professional Experience

Technical Director, Symantec, a Division of Broadcom *(March 2017 - Present)*

- Tracked cyber attackers; conducted and directed research to uncover attack campaigns; represented findings in public (print and television) and private (engineering, leadership or industry group) forums
- Regularly reported and discussed investigative research and recommendations to senior leadership within Symantec and across large enterprises; often in support of corporate, regional, national and international policy and/or strategic decision-making
- Participated as a subject matter expert in judicial proceedings against malware distributors

Sr. Researcher, Microsoft Malware Protection Center, Microsoft *(November 2015 – March 2017)*

- Researched and published on multiple targeted attack (APT) groups
- Organized and hosted Microsoft's first threat intelligence summit, at MSRA
- Formed and lead a virtual threat research team; formulating processes, resources, and KPIs

Sr. Manager, Attack Investigations Team, Symantec *(September 2011 – October 2015)*

- Founded and architected Symantec's first dedicated global threat intelligence team
- Established and formalized analytic processes for investigating various types of attack campaigns, including targeted attacks and cybercrime
- Established data collection parameters and associated tools for effective trait correlation
- Developed strategy for automated hunting of advanced threats
- Led analyst team in generating in-depth intelligence reports on high profile actors and threats
- Assisted law enforcement (NHTCU, Europol, BKA, FBI, NCA, HKPD and many more), and government agencies from across the globe (North American, Western-/Eastern-European,

and East-Asian) as a research and technical expert on cybercriminal and cyber espionage activities
- Participated as a subject matter expert in judicial proceedings against multiple high-profile fraud enabling botnets
- Collaborated with industry partners investigating multiple threats., including sharing technical information, setting up and operating sinkholes, acquiring server-side data through globally located hosting providers, and ultimately disseminating information to key stakeholders
- Presented the team's findings at various public and private forums and speaking engagements (RSA, Underground Economy, FS-ISAC, IAEA, and many more)
- Spokesperson for Symantec's global public outreach on highly sensitive security related issues and research in print, radio and live television (BBC, CNN, Fox, ABC, NBC, CBS, Al-Jazeera)

Principal Security Response Manager, Symantec *(July 2010 – September 2011)*
Senior Security Response Manager, Symantec *(July 2008 – July 2010)*
Security Response Manager, Symantec *(December 2006 – July 2008)*

- Lead and supervised the North American incident handling team
- Prioritized and collaborated with engineering team to find the most effective solutions to combat and remediate online threats in the most efficient manner
- Established and improved processes to respond to customer issues using a multitude of supported technologies (AV, IPS, Behavioral, and Reputational Engines)
- Briefed large enterprise clients on technologies as well as malware family behavior
- Designed and implemented an information sharing platform for disseminating information about existing threat landscape to Symantec personnel, including a workflow for handling of incidents warranting urgent attention
- Responded to legally driven third-party vendors disputing detection of their applications
- Improved processes to respond to false positives (FPs) by Symantec products
- Symantec spokesperson and public blogger for topical threats and research conducted in the early days of targeted attacks (APTs)
- Hired, trained and supervised junior Security Response Managers

System and Network Administrator, Florida State University *(August 2002 – October 2006)*

- Chief Analyst for network Intrusion Detection System (IDS)
- Configured IPSEC/L2TP/PPTP on the Cisco PIX to support VPN for multiple platforms: Windows/Linux/PDAs and Mac
- Unix/Linux administrator - Installation, configuration and support for both platform servers and workstations

Education

Master of Science, Computer Science, Florida State University, 2005
Bachelor of Engineering, Computer Science, University of Pune, 2000