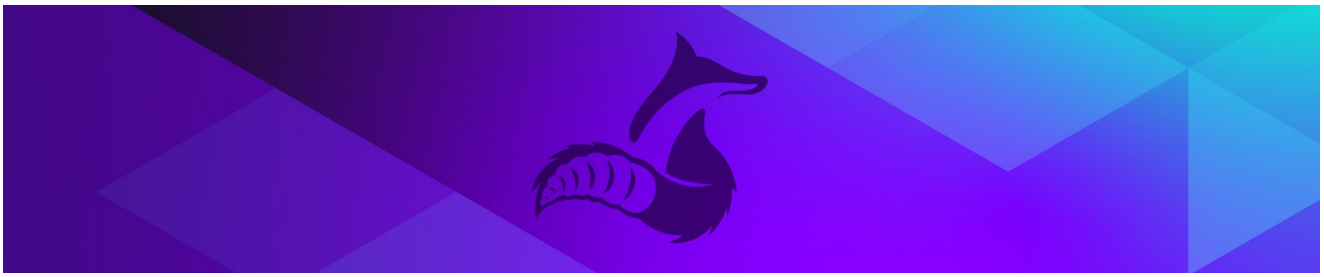
Purple Fox Rootkit Now Propagates as a Worm

guardicore.com/labs/purple-fox-rootkit-now-propagates-as-a-worm/



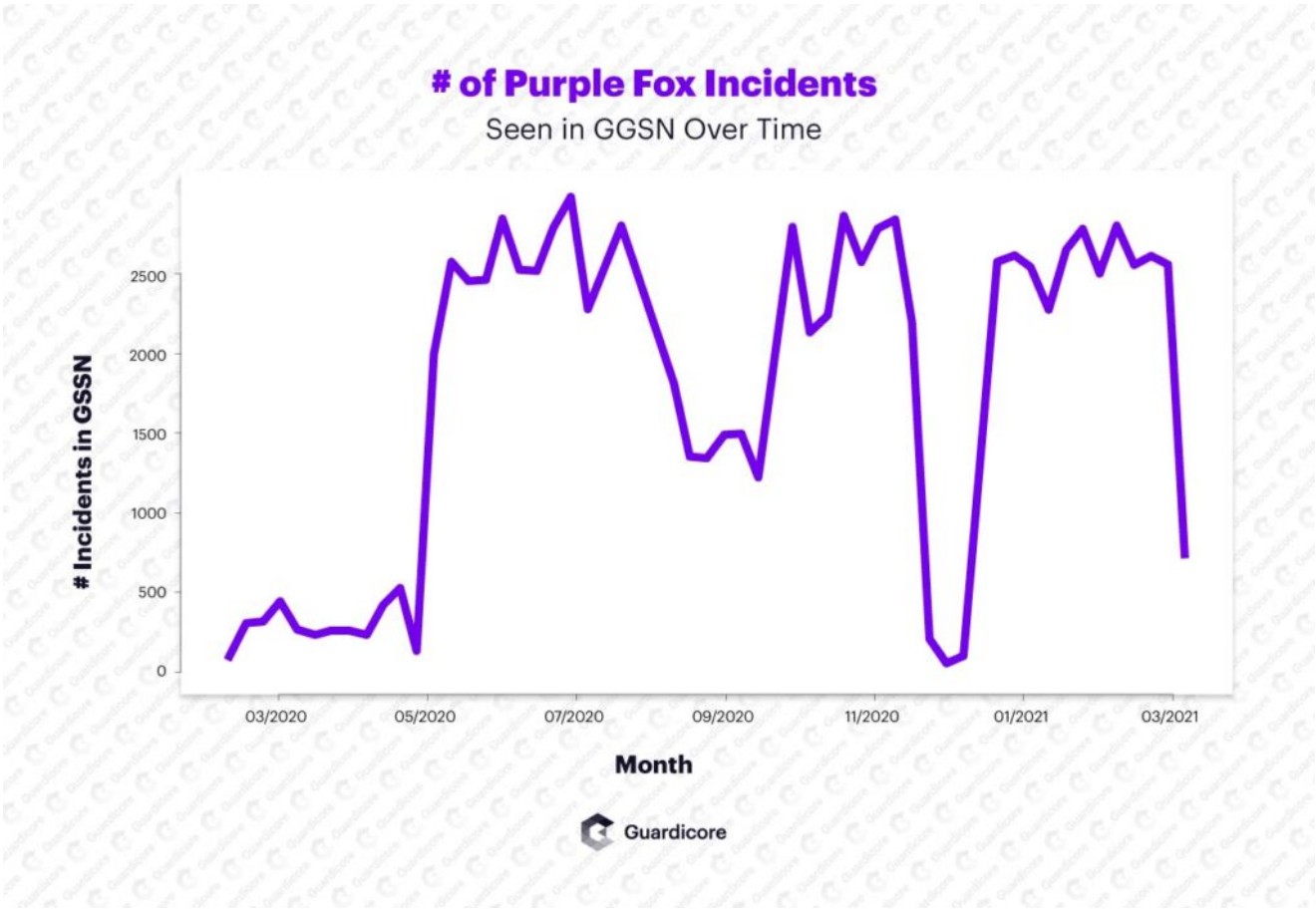
Executive Summary

- Purple Fox is an active malware campaign targeting **Windows machines**.
- Up until recently, Purple Fox's operators infected machines by using exploit kits and phishing emails.
- Guardicore Labs have identified **a new infection vector of this malware** where internet-facing Windows machines are being breached through SMB password brute force.
- Guardicore Labs have also identified Purple Fox's vast network of compromised servers hosting its dropper and payloads. These servers appear to be compromised Microsoft IIS 7.5 servers.
- The Purple Fox malware includes a **rootkit** which allows the threat actors to hide the malware on the machine and make it difficult to detect and remove.

Introduction

During the last few weeks, the Guardicore Labs team have been tracking a new campaign distributing the Purple Fox malware. Purple Fox was discovered in March of 2018 and was covered as an exploit kit targeting Internet Explorer and Windows machines with various privilege escalation exploits.

However, throughout the end of 2020 and the beginning of 2021, Guardicore Global Sensors Network (GGSN) detected Purple Fox's novel spreading technique via indiscriminate port scanning and exploitation of exposed SMB services with weak passwords and hashes.



By leveraging the capabilities of GGSN, we were able to track the spread of Purple Fox. As can be seen in the above graph, May of 2020 brought a significant amount of malicious activity and the number of infections that we have observed has risen by roughly 600% and amounted to a total of 90,000 attacks as of writing this paper.

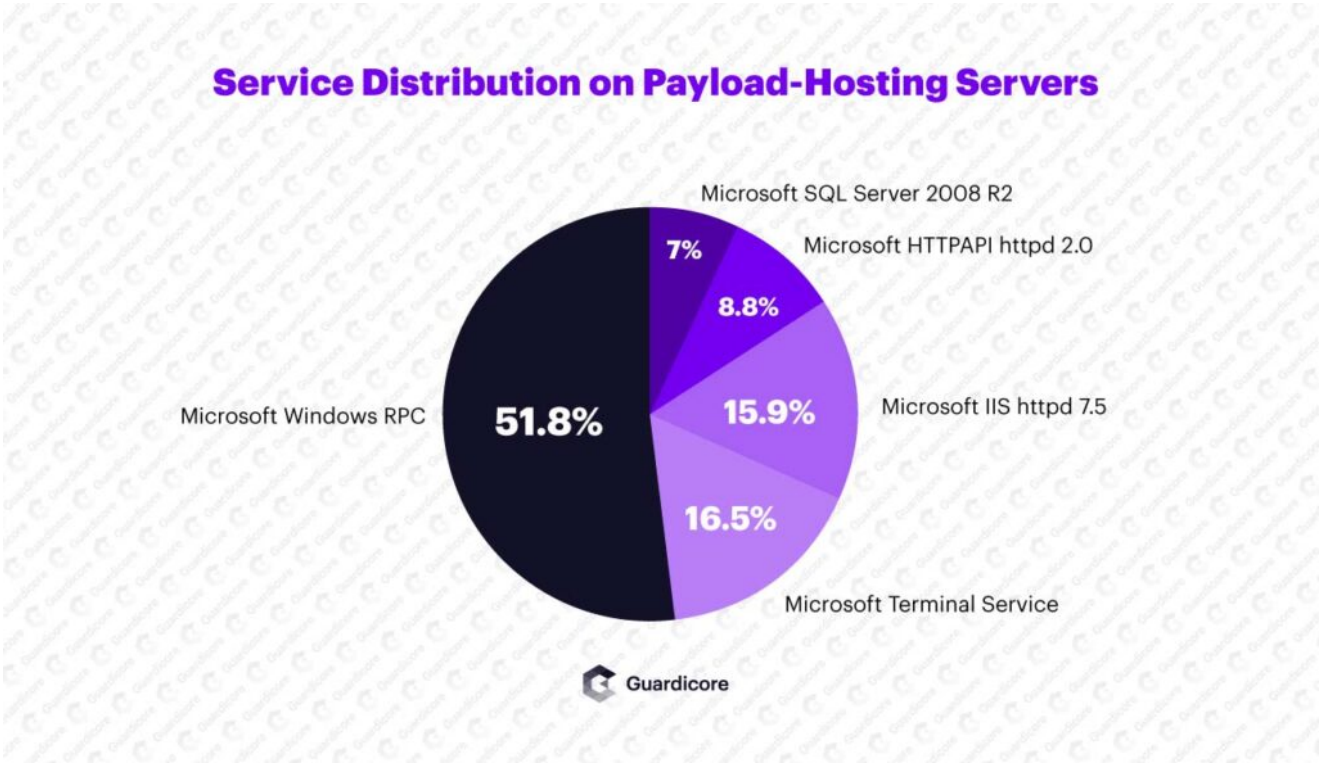
While it appears that the functionality of Purple Fox hasn't changed much **post exploitation**, its spreading and distribution methods – and its worm-like behavior – are much different than described in previously published articles. Throughout our research, we have observed an infrastructure that appears to be made out of a hodge-podge of vulnerable and exploited servers hosting the initial payload of the malware, infected machines which are serving as

nodes of those constantly worming campaigns, and server infrastructure that appears to be related to other malware campaigns.
In this blog post we will detail our findings about the new worm activity and share IOCs.

Attack Analysis

The attackers are hosting various MSI packages on nearly 2,000 servers (see IOCs section), which to our assessment are compromised machines which were repurposed to host malicious payloads. Our assumption is based on scanning multiple servers, looking at the services that are hosted on them from the perspective of operating system versions, and server versions.

We have established that the vast majority of the servers, which are serving the initial payload, are running on relatively old versions of Windows Server running IIS version 7.5 and Microsoft FTP, which are known to have multiple vulnerabilities with varying severity levels.



According to our findings, there are several ways for this campaign to start spreading:

1. The worm payload is being executed after a victim machine is compromised through a vulnerable exposed service (such as SMB).
2. The worm payload is being sent via email through a phishing campaign (which could tie the previously published findings about Purple Fox) which exploits a browser vulnerability. We have identified multiple samples that were submitted to VirusTotal through email scanners.

msi direct-cpu-clock-access runtime-modules

FEA41A78E1C94B4319CC9DEAC3BA0ABF01880482EA53C8C8B1BD3B52684B8EB0

/var/www/clean-mx/virusesevidence/output.167847160.txt

msi runtime-modules direct-cpu-clock-access

EFADAA85BAE06516D6F38C2C5AAE3813E81E28B3A0ED18F91DA07CB32B221340

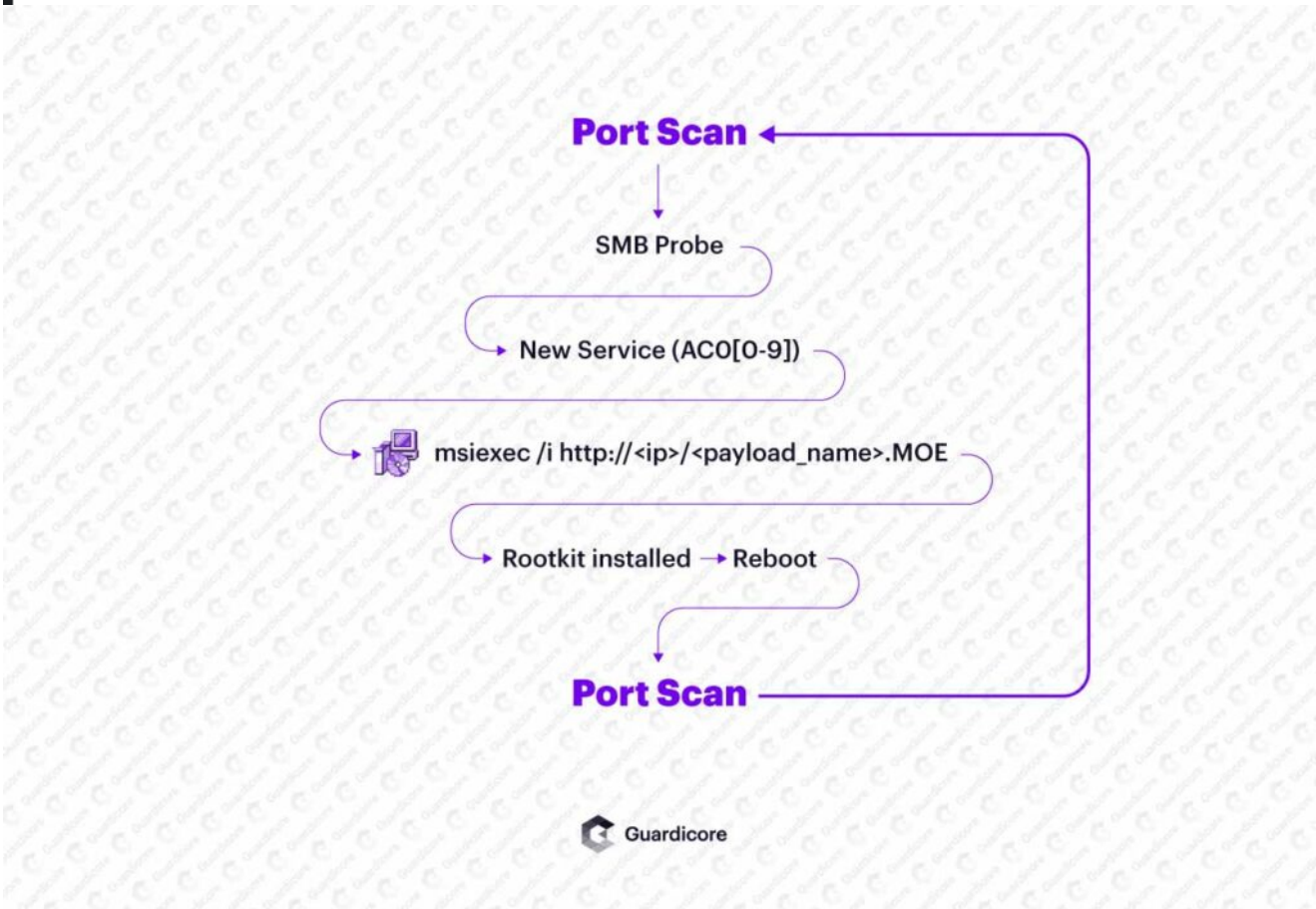
No meaningful names

msi direct-cpu-clock-access runtime-modules

3CA3C1B9E8DA6390411889E64FFAF0BC8BA24662A2E3DC8E85827A197428461D

/var/www/clean-mx/virusesevidence/output.168874457.txt

msi runtime-modules direct-cpu-clock-access

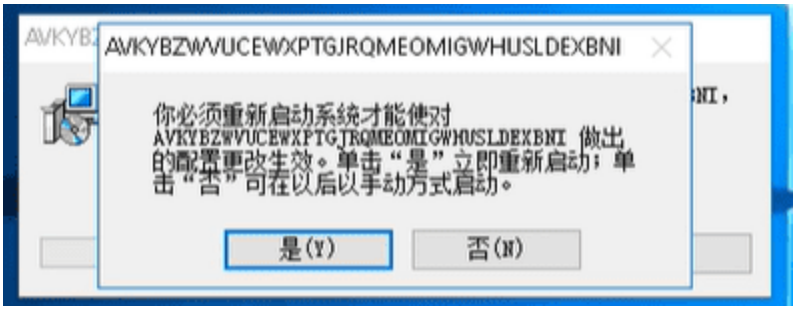


Once code execution is achieved on the victim machine, a new service whose name matches the regex AC0[0-9]{1} — e.g. AC01, AC02, AC05, etc. will be created, the purpose of this service would be to establish persistence and to execute a simple command with a 'for loop', the purpose of this command would be to iterate through a number of URLs which contain the MSI that installs Purple Fox on the machine.

As can be seen in the screenshot from the Guardicore Centra platform, *msiexec* will be executed with the /i flag, in order to download and install the malicious MSI package from one of the hosts in the statement. It will also be executed with the /Q flag for “quiet” execution, meaning, no user interaction will be required.

Once the package is executed, the MSI installer will launch.

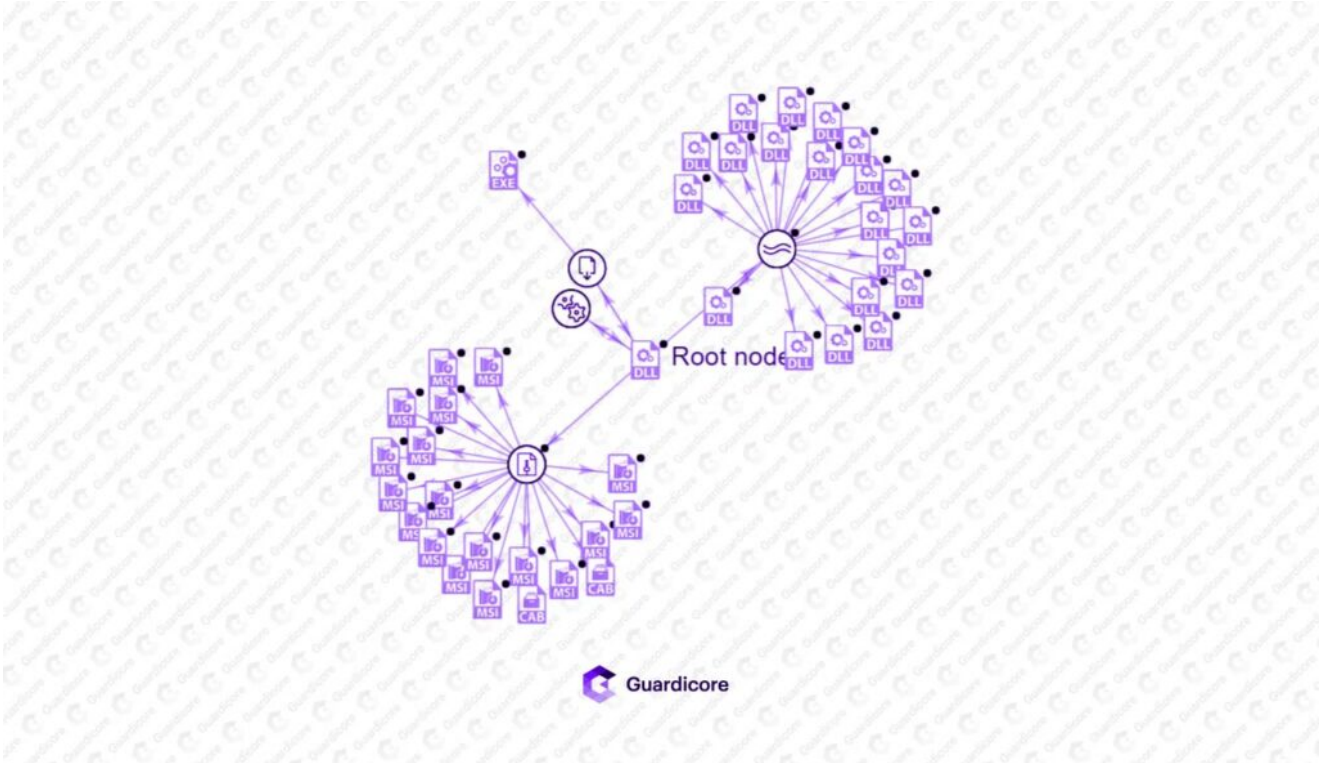
For analysis purposes,We have executed the MSI installer without the **/Q** flag (As if it’s being executed directly from an email attachment), the installer will present the following window:



The installer pretends to be a Windows Update package along with Chinese text which roughly translates to “Windows Update” and random letters. These letters are randomly generated between each different MSI installer to create a different hash and make it a bit difficult to tie between different versions of the same MSI. This is a “cheap” and simple way of evading various detection methods such as static signatures. Additionally, we have identified MSI packages with the same strings but with random null bytes appended to them in order to create different hashes of the same file.

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
0025E240	D0	81	00	00	00	00	80	A0	FD	A9	17	29	84	14	00	00	Ð.....€ ý©.)„...
0025E250	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
0025E260	00	00	00	00	00	00	00	00	00	00	00	00	00	40	0D	24	.....@.$
0025E270	D9	DF	5C	24	01	38	31	43	4B	ED	92	C1	B1	18	31	08	Ùß\$.81CKí'Á±.1.
0025E280	43	EF	99	F9	3D	A4	04	C0	60	A0	FF	C6	BE	11	9B	1E	Cï™ù=¤.À` ÿÆ¾.›.
0025E290	72	D0	BB	30	36	C2	08	76	B9	02	42	08	21	84	10	42	rÐ»06Â.v¹.B.!„.B
0025E2A0	08	21	84	10	42	08	21	84	10	42	08	21	84	10	42	08	.!„.B.!„.B.!„.B.
0025E2B0	21	84	10	42	08	21	FF	01	96	E9	FD	F7	E7	8F	65	C5	!„.B.!ÿ.–éý÷ç.eÅ
0025E2C0	79	31	0D	47	ED	8E	7C	31	B4	36	7B	90	B5	AA	F6	17	y1.GíŽ|1´6{.µªö.
0025E2D0	8F	D8	BD	A3	6E	49	A8	DD	75	D2	11	36	67	EB	CC	98	.Ø½£nI¨ÝuÒ.6gëÌ˜
0025E2E0	7B	F1	C4	FD	51	B1	39	DF	90	09	C7	3A	F6	71	5F	F5	{ñÄýQ±9ß..Ç:öq_õ
0025E2F0	29	3C	EA	8E	D7	B2	1B	6A	8F	9E	26	47	13	CD	CC	4C	)<êŽ×².j.ž&G.ÍÌL
0025E300	E6	B5	9B	2A	70	1A	B8	F5	CE	AD	CE	82	5A	3A	0C	55	æµ›*p.¸õÎ.Î‚Z:.U
0025E310	5F	CF	3E	36	5D	34	CC	11	AB	65	27	AC	83	6E	B2	5E	_Ï>6]4Ì.«e'¬ƒn²^
0025E320	2D	72	4D	5D	8C	14	29	05	0F	AD	98	C8	FD	FC	EB	06	-rM]Œ.)...˜Èýüë.
0025E330	75	C6	C6	7E	9B	18	9D	56	22	AF	F7	F3	AA	BB	D6	D7	uÆÆ~›..V"¯÷ó ª»Ö×
0025E340	04	EF	DC	5D	48	43	1D	8A	E0	95	E3	F0	F6	41	EE	F9	.ïÜ]HC.Šà•ãðöAîù
0025E350	40	54	B9	28	75	51	85	61	C5	40	AF	71	6E	67	1D	99	@T¹(uQ…aÅ@¯qng.™
0025E360	63	0A	B3	EB	98	56	4E	41	2C	96	82	8D	87	EE	87	8B	c.³ë˜VNA,–‚.‡î‡‹
0025E370	CF	C7	F7	21	5B	31	C7	41	E3	FC	36	D3	3D	39	37	A4	ÏÇ÷![1ÇAãü6Ó=97¤
0025E380	B4	65	4B	5F	25	46	74	35	A8	CB	65	6D	6A	6D	3A	CF	´eK_%Ft5¨Ëemjm:Ï
0025E390	FE	0A	D8	B3	65	AF	83	70	9B	F2	5F	00	00	00	00	00	þ.Ø³e¯ƒp›ò_.....
0025E3A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
0025E3B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
0025E3C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
0025E3D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
0025E3E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
0025E3F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....

We were, however, able to find many different versions of the same MSI and its payloads, as can be seen in the following screenshot from VT graph.



As the installation progresses, the installer will extract the payloads and decrypt them from within the MSI package. The MSI package contains three files:

1. A 64bit DLL payload (*winupdate64*).
2. A 32bit DLL payload (*winupdate32*).
3. An encrypted file containing a rootkit.

As a part of the installation process, the malware modifies the windows firewall by executing multiple *netsh* commands. The malware adds a new policy named ***Qianye*** to the windows firewall*.* Under this policy, it creates a new filter called *Filter1* and under this filter, it prohibits ports 445, 139, 135 on both TCP and UDP from any IP address on the internet (0.0.0.0) to connect to the infected machine, we believe that the attackers are doing it in order to prevent the infected machine from being reinfected, and/or to be exploited by a different threat actor.

Once the aforementioned files are being extracted, they will be executed.

This can be seen as the malware is executing the following commands:

```
netsh.exe ipsec static add policy name=qianye

netsh.exe ipsec static add filterlist name=Filter1

netsh.exe ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me
dstport=135 protocol=TCP

netsh.exe ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me
dstport=139 protocol=TCP

netsh.exe ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me
dstport=445 protocol=UDP

netsh.exe ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me
dstport=135 protocol=UDP

netsh.exe ipsec static add filter filterlist=Filter1 srcaddr=any dstaddr=Me
dstport=139 protocol=UDP

netsh.exe ipsec static set policy name=qianye assign=y

netsh.exe ipsec static add rule name=Rule1 policy=qianye filterlist=Filter1
filteraction=FilteraAtion1

netsh.exe ipsec static add rule name=Rule1 policy=qianye filterlist=Filter1 netsh.exe
ipsec static add filteraction name=FilteraAtion1 action=block
```

Additionally, the malware will install an IPv6 interface on the machine by executing the command:

```
netsh.exe interface ipv6 install
```

This action is taken in order to allow the malware to port scan ipv6 addresses as well to maximize the efficiency of the spread over (usually unmonitored) ipv6 subnets.

Important note:

*These commands can be used as **Indicators of Behavior** (IoBs) in order to check if your environment is compromised.*

Additionally, These netsh commands have also appeared in previous campaigns and are not exclusive to this iteration of Purple Fox. These commands, specifically with the Qianye policy name have been documented as a part of Rig EK and NuggetPhantom.

The last step of Purple Fox's deployment before restarting the machine is to load the rootkit that's hidden inside the encrypted payload in the MSI package.

According to our analysis, the rootkit is based on the *hidden* open source rootkit project.

00409a9c	Hid_State	u"Hid_State"	unicode
00409ab0	Hid_StealthMode	u"Hid_StealthMode"	unicode
00409ad0	Hid_HideFsDirs	u"Hid_HideFsDirs"	unicode
00409aee	Hid_HideFsFiles	u"Hid_HideFsFiles"	unicode
00409b0e	Hid_HideRegKeys	u"Hid_HideRegKeys"	unicode
00409b2e	Hid_HideRegValues	u"Hid_HideRegValues"	unicode
00409b52	Hid_IgnoredImages	u"Hid_IgnoredImages"	unicode
00409b76	Hid_ProtectedImages	u"Hid_ProtectedImages"	unicode

```
3
4   #define CONFIG_ALLOC_TAG 'gfnC'
5
6   typedef struct _HidConfigContext {
7           BOOLEAN state;
8           BOOLEAN stealth;
9           UNICODE_STRING hideFSDirs;
10          UNICODE_STRING hideFSFiles;
11          UNICODE_STRING hideRegKeys;
12          UNICODE_STRING hideRegValues;
13          UNICODE_STRING ignoreImages;
14          UNICODE_STRING protectImages;
15  } HidConfigContext, *PHidConfigContext;
16
```

The purpose of this rootkit is to hide various registry keys and values, files, etc., as detailed by its author on the git repository. Ironically enough, the *hidden* rootkit was developed by a security researcher in order to conduct various malware analysis tasks and to keep all of these research tasks **hidden** from the malware.

Hidden

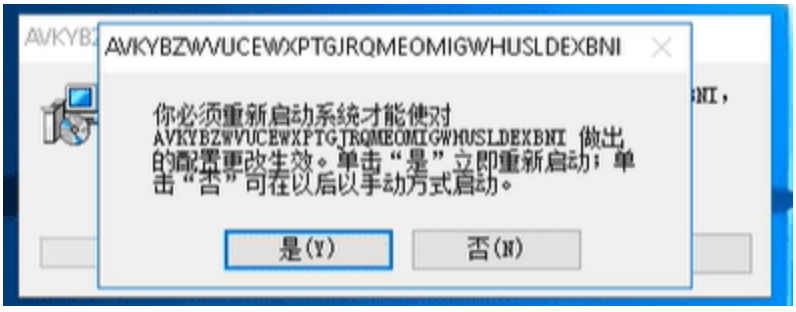
This toolset is developed like a solution for my reverse engineering and researching tasks. This is a windows driver with a usermode interface which is used for hidding specific environment on VMs, like installed rce programs (ex. procmon, wireshark), vm infrastracture (ex. vmware tools) and etc.

Features

- hide registry keys and values
- hide files and directories
- protect specific processes using ObRegisterCallbacks
- exclude specific processes from hidding and protection features
- usermode interface (lib and cli) for working with driver

This rootkit and its relationship with Purple Fox was detailed in this article by 360 Security.

Once the rootkit is loaded, the installer will reboot the machine in order to rename the malware DLL into a system DLL file that will be executed on boot. Since we executed the malware in our lab without the **/Q** flag, we were presented with the following window which asks us to restart the machine:



Once the machine is restarted, the malware will be executed as well. After it's execution, the malware will start its propagation process: the malware will generate IP ranges and start scanning them on port 445.

As the machine responds to the SMB probe that's being sent on port 445, it will try to authenticate to SMB by brute forcing usernames and passwords or by trying to establish a null session.

```
2892 270.361390    192.168.100.211    180.218.95.239     TCP    66 49920 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2893 270.361440    192.168.100.211    180.218.123.60     TCP    66 49921 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2894 270.361510    192.168.100.211    180.218.116.216    TCP    66 49922 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2895 270.361555    192.168.100.211    180.218.136.176    TCP    66 49923 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2896 270.361623    192.168.100.211    180.218.237.234    TCP    66 49924 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2897 270.361695    192.168.100.211    180.218.58.0       TCP    66 49925 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2898 270.361756    192.168.100.211    180.218.214.177    TCP    66 49915 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2899 270.362028    192.168.100.211    180.218.3.103      TCP    66 49926 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2900 270.362109    192.168.100.211    180.218.46.74      TCP    66 49927 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2901 270.362179    192.168.100.211    180.218.254.171    TCP    66 49928 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2902 270.362249    192.168.100.211    180.218.104.44     TCP    66 49929 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2903 270.362310    192.168.100.211    180.218.156.41     TCP    66 49930 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2904 270.362373    192.168.100.211    180.218.139.153    TCP    66 49931 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2905 270.362426    192.168.100.211    180.218.86.140     TCP    66 49932 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2906 270.362505    192.168.100.211    180.218.201.190    TCP    66 49933 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2907 270.362537    192.168.100.211    180.218.4.172      TCP    66 49934 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2908 270.362610    192.168.100.211    180.218.100.153    TCP    66 49935 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2909 270.362670    192.168.100.211    180.218.248.98     TCP    66 49936 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2910 270.362707    192.168.100.211    180.218.166.132    TCP    66 49938 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2911 270.362743    192.168.100.211    180.218.87.209     TCP    66 49939 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2912 270.362821    192.168.100.211    180.218.57.85      TCP    66 49940 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2913 270.362851    192.168.100.211    180.218.186.117    TCP    66 49942 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2914 270.362877    192.168.100.211    180.218.143.24     TCP    66 49943 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
```

If the authentication is successful, the malware will create a service whose name matches the regex AC0[0-9]{1} — e.g. AC01, AC02, AC05 (as mentioned before) that will download the MSI installation package from one of the many HTTP servers and thus will complete the infection loop.

Indicators of Compromise

IOCs are available in our IOC github repository