No money, but Pony! From a mail to a trojan horse | Malwarebytes Labs

By hasherezade

Published: 2015-11-18 · Archived: 2026-04-05 18:22:14 UTC

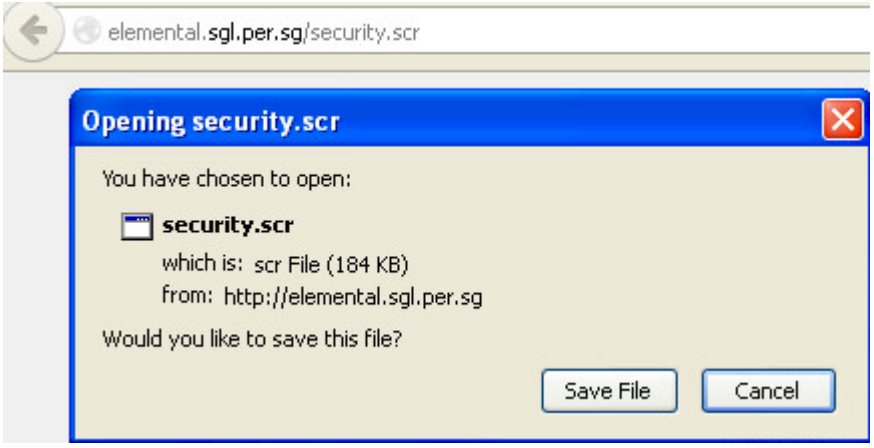
In this post, we will take a high and low-level look at the Pony Trojan, delivered through a recent spam campaign.

During our case study we showed some malicious samples being distributed in spam campaigns. Using this distribution method, malware is often found attached to the e-mail as either:

- an executable (also compressed, i.e. ***zip***, ***rar*** or ***cab*** archive), sometimes pretending to be a different file format, like Dyreza
- a document (commonly PDF or some MS Office format) – like this Dridex downloader

This time we will present a sample with a bit different delivery method. Instead of attaching the malicious file, attackers decide to just send a link and convince users to download the malware:

The scam is to make users curious about an unexpected money transfer, leading them to click on one of the links and download the associated file. It doesn't really matter which link they click on, since they both deliver the same sample.



During download the browser may show a typical executable icon. The unusual extension is just another trick to confuse users, who might be more wary of ***exe*** but not as much when encountering ***scr.*** The ***scr*** extension is used for screensavers – but despite the different name, they are normal executables, and they can be run by Windows in the same way.

The downloaded file tries to look trustworthy by using a well-known Adobe Reader icon and the filename "security" or "infos".

Once executed, it deploys the Pony Trojan on the system. For more information about detection of this malware, click the link below:

md5=8a55ecad10a7cf3dad3630ac40e420a1

For those of you, who are satisfied just by knowing that the file is malicious, you can stop reading after seeing the VirusTotal report. But if you are interested in features of this malware family and in tricks that it uses to hide its real mission, keep reading!

Elements involved

- 8a55ecad10a7cf3dad3630ac40e420a1 – original, packed sample (*security.scr*)
  - b60d3a994a9074cc59d1e065d2583411 – Pony Loader
    - 9a822a6232b932187cd1857a740dfb85 – payload downloaded by Pony Loader (url format: *http://(…)/wp.php*)

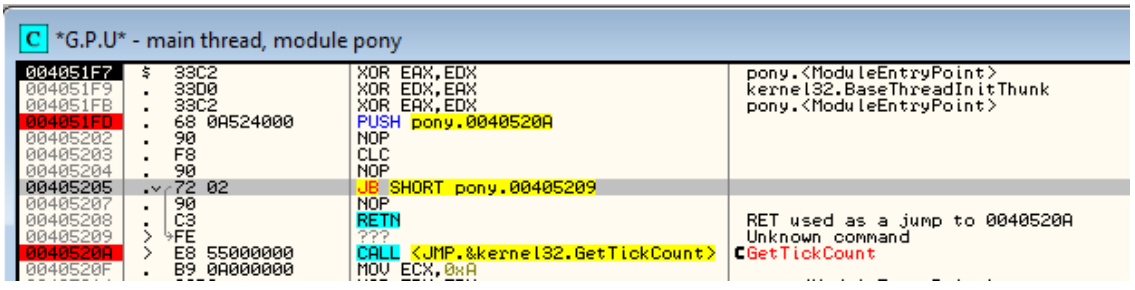
The original sample – *security.scr* is just an outer packing, used for the purpose of obfuscation. It loads into memory another fully independent executable: Pony Loader.

Pony Loader

Some years ago, the sourcecode of Pony Loader (bot) ***1.9*** along with Pony Builder (bot configurator) leaked online. Later the same happened with version ***2.0***. Both sets became available to download on various forums. During this analysis, I will compare the current sample with the leaked material in order to identify changes made by the attackers.

Obfuscation Tricks

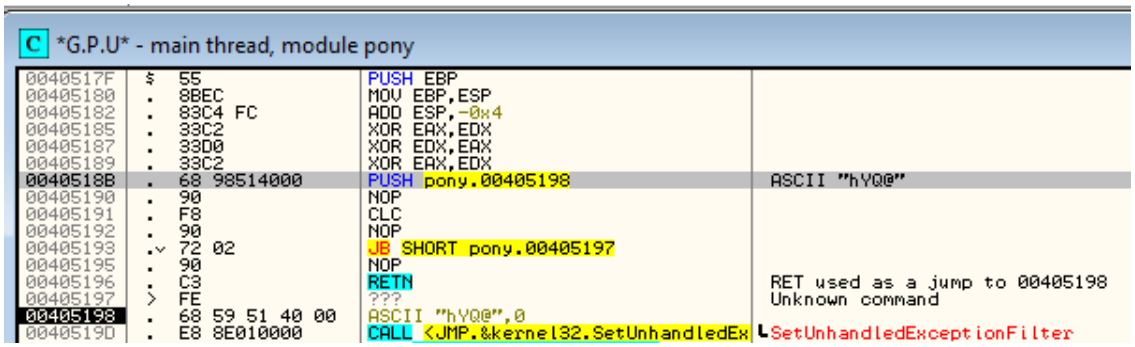
Let's take a look at the Entry Point:



As we can see, the flow is obfuscated. Transitions between basic blocks are made using the well known trick: PUSH-to-RET, which emulates a CALL to an address that is pushed on to stack. But in Pony this technique is

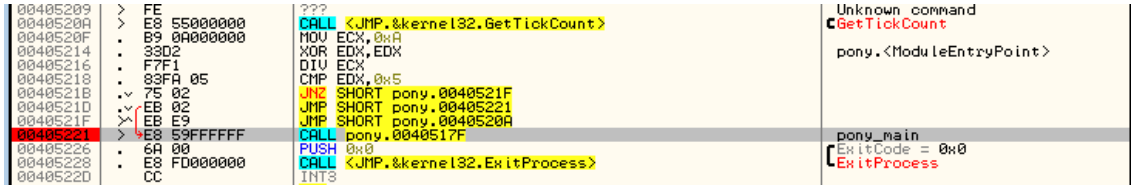
used in more sophisticated way because there are some junk instructions added between the PUSH and the RET in addition to a never executed bogus conditional jump.

Due to these tricks, sometimes common tools fail to correctly interpret the code. Example below:



OllyDbg interpreted the pushed address as a string

Another trick used by this malware is delaying execution. For example, the malware executes *GetTickCount* in a loop till it gets a value satisfying specific condition. The algorithm behind this trick is simple. The value returned by *GetTickCount* is divided by a predefined number. When the remainder equals another predefined value, the loop terminates. As a result *GetTickCount* runs pseudo-random number of times before the execution can continue.



This particular functionality matches the pattern found in Pony 1.9:

[code firstline="1004" highlight="1017" title="Pony.asm"] MainEntryPoint: AntiDisasmTrick

.WHILE TRUE invoke GetTickCount mov ecx, 10 xor edx, edx div ecx .IF edx == 5 .BREAK .ENDIF .ENDW

invoke DoWork

invoke ExitProcess, 0 [/code]

Strings

The authors of the malware didn't took care about obfuscating strings or API calls. At this stage, we can see all of them clearly. Some of the strings are the same (or suggesting equivalent functionality) to those from the sample analyzed by MalwareMustDie in 2013. However, the current sample seems not as offensive, for example it doesn't include as many strings that reference password stealing as the previous one did.

You can see complete (and commented) list of strings here:
https://gist.github.com/hasherezade/1f3199b7b752db5d46c6

Target Identification

Specific modules in the sourcecode are included or excluded according to defined flags. The currently analysed sample have the following module included – being used to target 'NetSarang XFTP':

[code firstline="8985" highlight="8991,9011" title="PasswordModules.asm"] ; XFTP ; http://www.netsarang.com/forum/xftp/list ; Tested: Xftp 4 (Build 0077) ; Tested: Xftp 4 (Build 0083) ; SFTP: implemented

IFDEF COMPILE_MODULE_XFTP

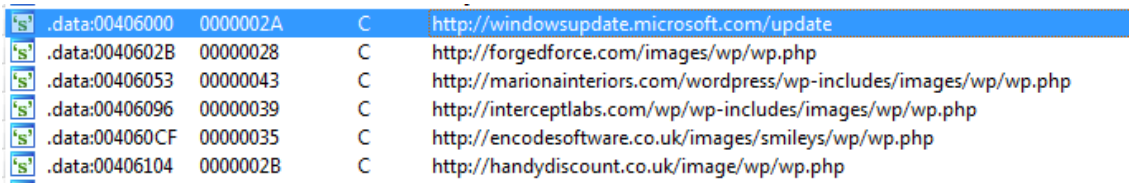
.data CXftpAppDataDir db 'NetSarang',0 CXftpConfigFile db '.xfp',0 .code

GrabXFTP proc stream LOCAL hdr_ofs: DWORD invoke StreamWriteModuleHeader, stream, MODULE_XFTP, 0 mov hdr_ofs, eax invoke AppDataCommonFileScan, stream, offset CXftpAppDataDir, offset CXftpConfigFile, ITEMHDR_ID or 0 invoke StreamUpdateModuleLen, stream, hdr_ofs ret GrabXFTP endp

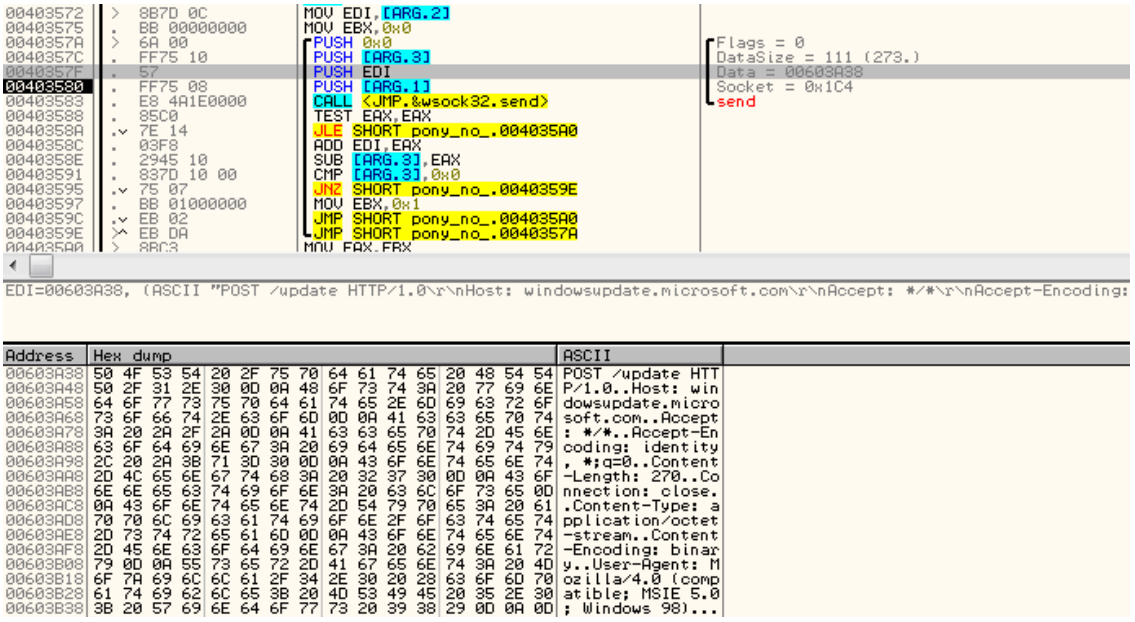
ENDIF [/code]

Network Communications

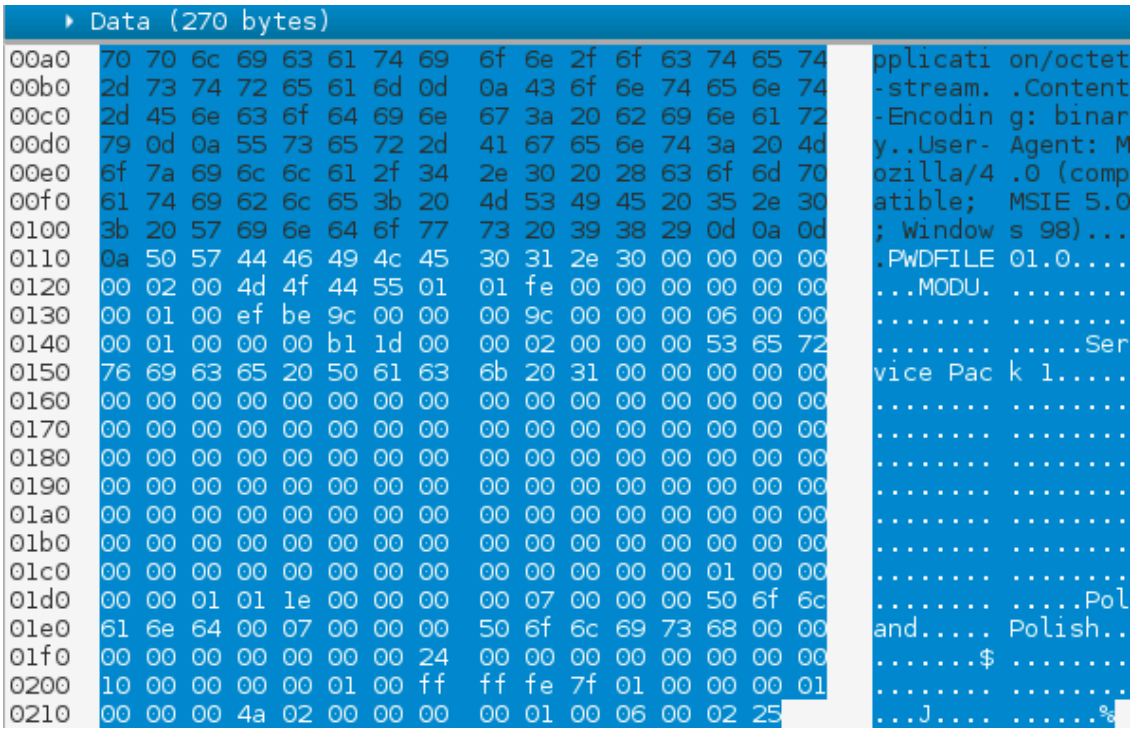
It didn't take long to locate URLs queried by our Pony sample:



The First URL, windows update, is used just after collecting information about the system. The malware sends a POST request to the address as seen below.



The actual data being sent is an unencrypted report created by Pony, listing information about the infected system. This traffic contains the keyword “PWDFILE0” and “MODU” as well as any stolen credentials the malware might have extracted.

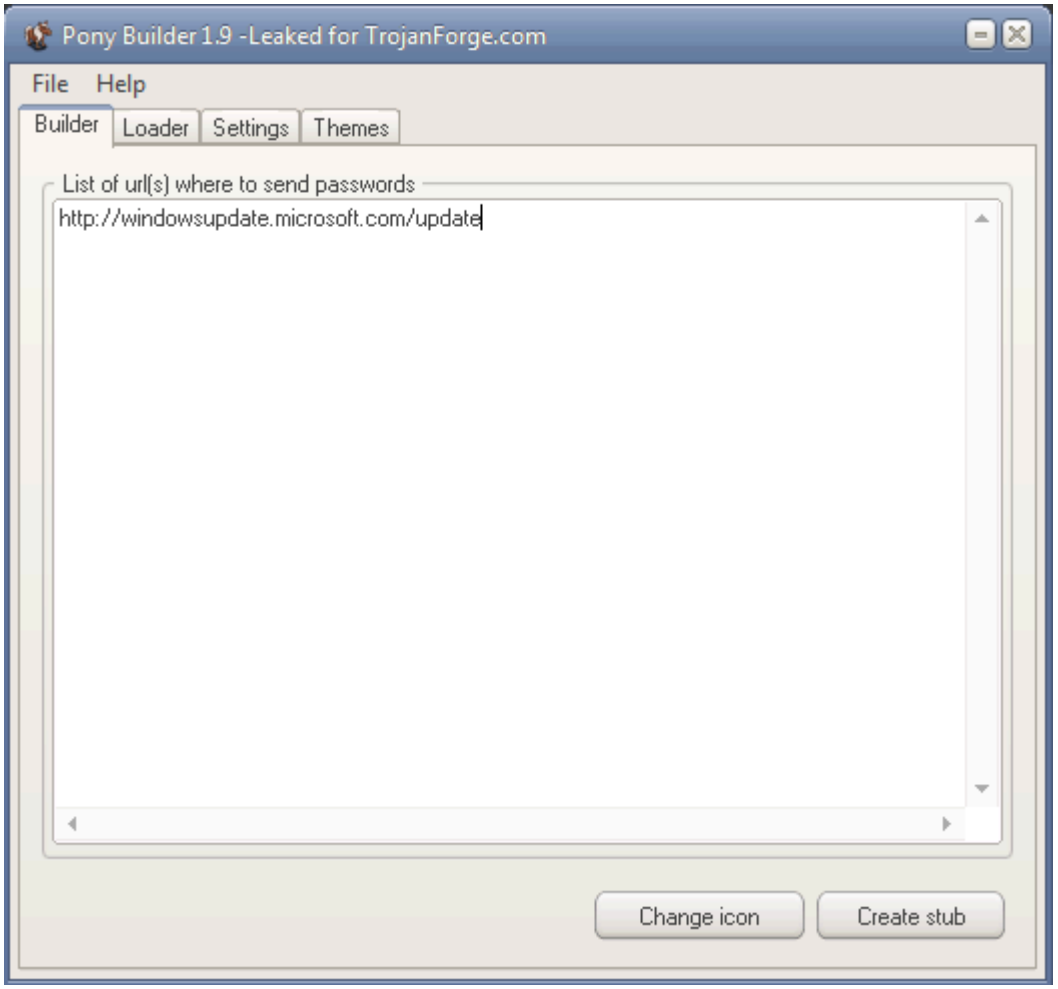


If you are wondering why this type of report was sent to Windows Update server, I wondered the same thing? To find out, I referred to the original code in order to check the intention behind it. As the code states, this function is supposed to send the stolen credentials to the C&C!

[code title=”Pony.asm, function: DoWork” firstline=”961″] ; Scan and send passwords invoke ScanAndSend [/code]

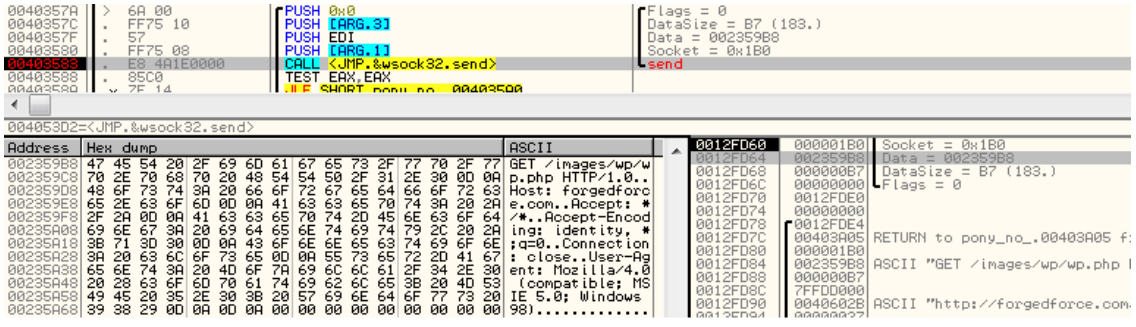
It seems that distributors of this piece of malware were not at all interested in collecting credentials, which is why they set the beacon URL to the Windows Update address rather than a C&C which could collect and store the stolen information. This probably happened because of lazy coders – instead of removing this fragment of code they redirected sending to a bogus URL.

We reconstructed how the configuration might have looked using the Pony Builder:

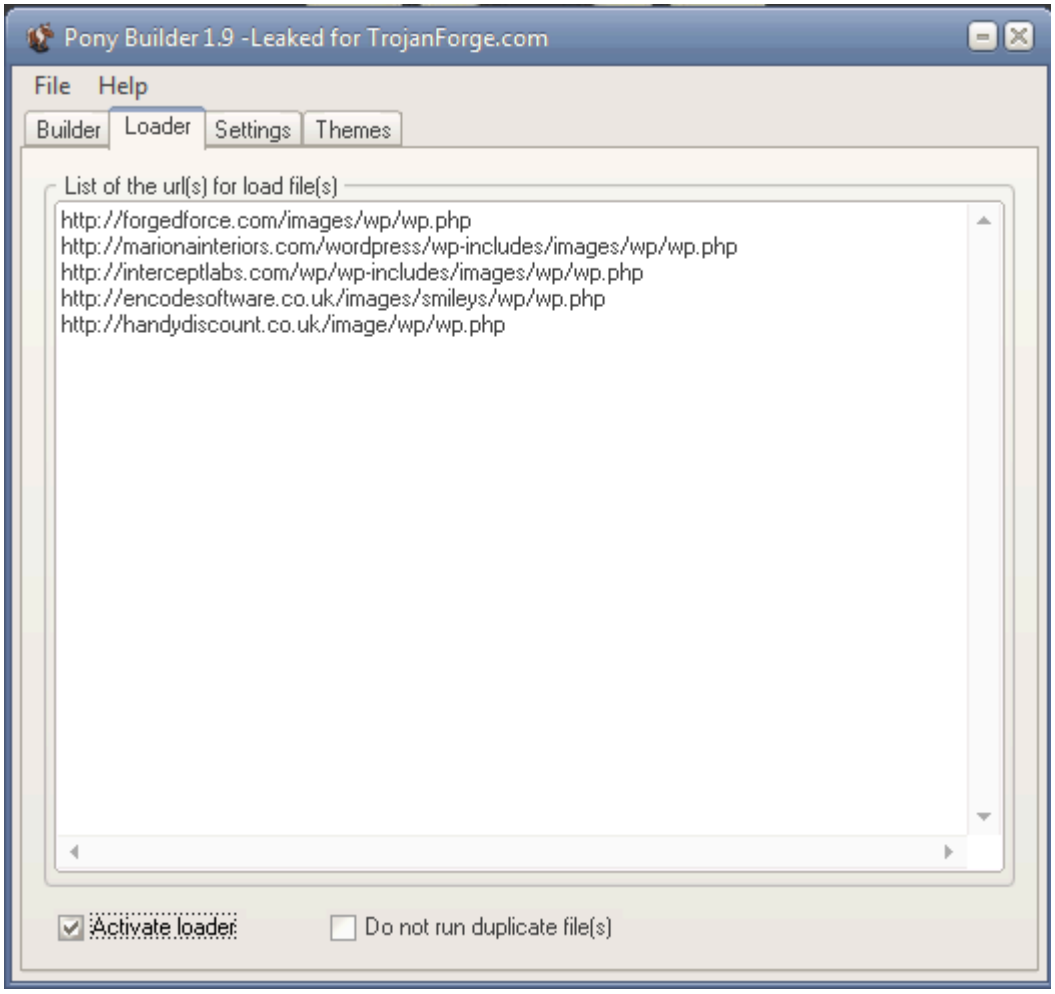


Pony also has the functionality of downloader.

[code title="Pony.asm, function: DoWork" firstline="964"] ; Run loader IFDEF ENABLE_LOADER invoke RunLoader ENDIF [/code] The other URLs (ending *wp.php*) are alternative locations of the second payload. They have extension ***php***, but they serve a malicious executable that is downloaded by Pony, saved as ***exe*** and run. The malware reached out to each of the URLs, in a loop, in order to find an active one. The malware uses a hard-coded GET request to reach out:



Those addresses were set at the “Loader” page in the Pony Builder:



Password Cracking

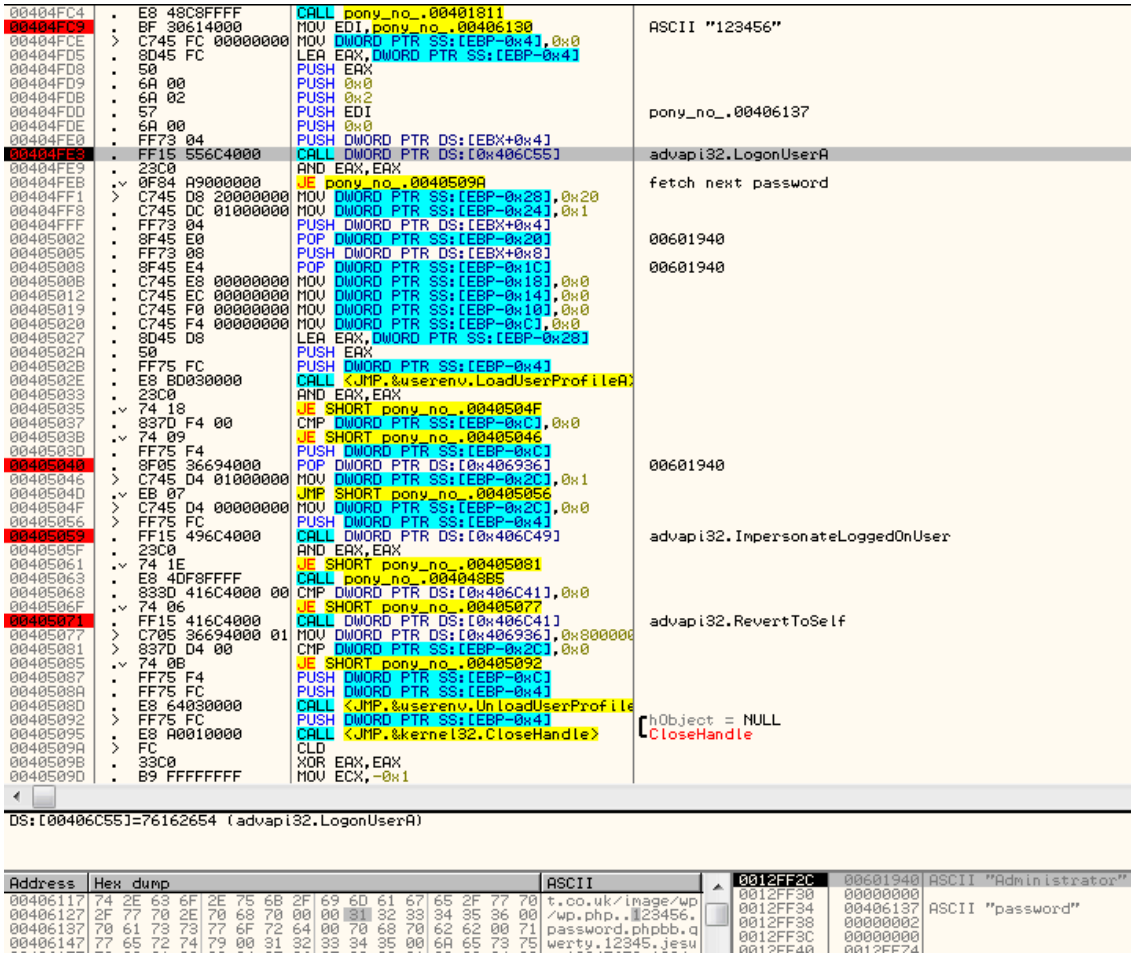
The Pony agent comes with a small dictionary of commonly used passwords.

```
.data:00406130 a123456         db '123456',0           ; DATA XREF: .text:00404FC9↑o
.data:00406137 aPassword       db 'password',0
.data:00406140 aPhpbb          db 'phpbb',0
.data:00406146 aQwerty         db 'qwerty',0
.data:0040614D a12345          db '12345',0
.data:00406153 aJesus          db 'jesus',0
.data:00406159 a12345678       db '12345678',0
.data:00406162 a1234           db '1234',0
.data:00406167 aAbc123         db 'abc123',0
.data:0040616E aLetmein        db 'letmein',0
.data:00406176 aTest           db 'test',0
.data:0040617B aLove           db 'love',0
.data:00406180 a123            db '123',0
.data:00406184 aPassword1      db 'password1',0
.data:0040618E aHello          db 'hello',0
.data:00406194 aMonkey         db 'monkey',0
.data:0040619B aDragon         db 'dragon',0
.data:004061A2 aTrustno1       db 'trustno1',0
.data:004061AB a111111         db '111111',0
.data:004061B2 aIloveyou       db 'iloveyou',0
.data:004061BB a1234567        db '1234567',0
.data:004061C3 aShadow         db 'shadow',0
.data:004061CA a123456789      db '123456789',0
.data:004061D4 aChrist         db 'christ',0
.data:004061DB aSunshine       db 'sunshine',0
.data:004061E4 aMaster         db 'master',0
.data:004061EB aComputer       db 'computer',0
.data:004061F4 aPrincess       db 'princess',0
```

The list matches a list found in the leaked sourcecode of Pony 1.9:

```
; Password list used in windows user logon bruteforcer
```

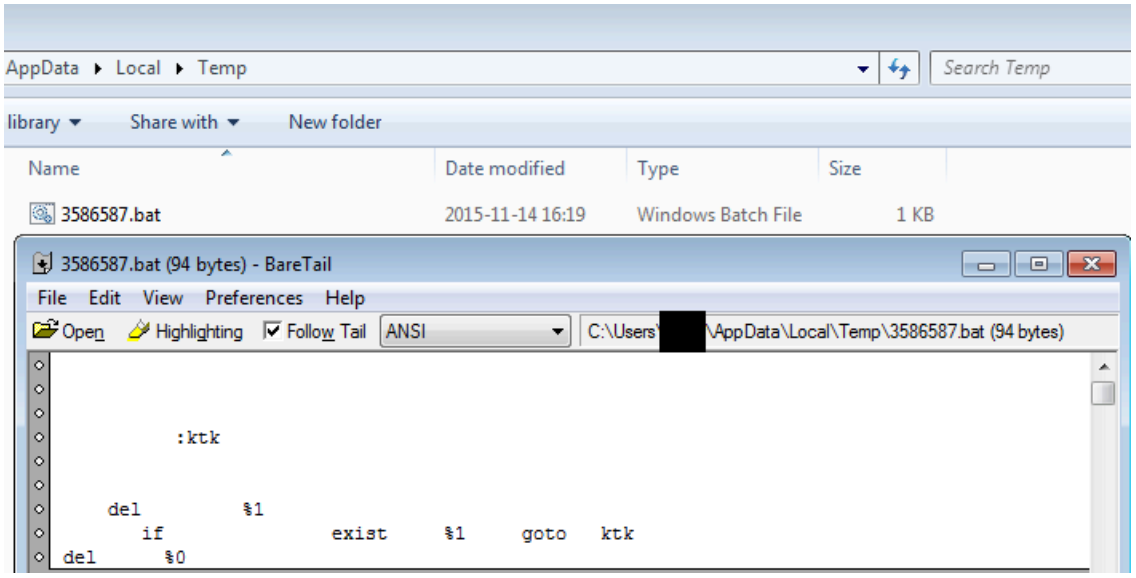
This dictionary is used in attack against local accounts retrieved by function NetUserEnum.



Example: the malware tries to login as "Administrator" checking all the passwords from the dictionary.

Auto deleting

Finally, Pony creates a batch script in %TEMP% with weirdly formatted content:



This script is meant to delete the Pony Loader after execution (works in a loop, in order to wait for the sample to terminate). The same can be found in Pony 1.9 code:

```
.data   szBatchFmt                              db      '%d.bat',0      szSelfDelQuoteFmt       db
```

Conclusion

This sample seems to be compiled from the source of Pony 1.9 – the old one, without recent additions and improvements. Moreover, some features of the original source are removed (i.e. related to credentials stealing). It seems that in this case, Pony Loader is used mainly as a downloader.

As the current example shows, sometimes “new” malware samples are not so new – only they are packed by new packers/crypters.

Attackers often use leaked sourcecode as a base – but they neglect the fact, that the same material is also available to malware analysts – allowing them to easily reveal everything what they wanted to hide.

Appendix

http://blog.malwaremustdie.org/2013/06/case-of-pony-downloaded-zeus-via.html – description of Pony Loader by @malwaremustdie

About the author

Unpacks malware with as much joy as a kid unpacking candies.
