

XOR Analysis & Deobfuscation Part 1

MICHAEL BARR (@mbarr63)

MAY 2013



What is this talk about?

Tools and techniques for understanding information that has been hidden from casual view using XOR obfuscation techniques.

“There are two kinds of cryptography in this world: cryptography that will stop your kid sister from reading your files, and cryptography that will stop major governments from reading your files.”

Bruce Schneier



In The Press...

“Attackers use obfuscation to make it harder to analyze malicious software and stymie security tools ... Currently, most obfuscation is simple, using operations such as XOR-ing bits or rotating through alphanumeric characters”

From article written by: Robert Lemos (April 13, 2011)

<http://www.darkreading.com/applications/malware-writers-making-code-tougher-to-d/229401546>

In The Press Continued...

“In this instance, the malware has used a number of tricks, including evading sandbox analysis by detecting human behavior, evading network-level binary extraction technology by performing multibyte XOR encryption of executable files...”

From article written by: Lucian Constanti (April 2, 2013)

<http://www.infoworld.com/d/security/researchers-find-apt-malware-monitors-mouse-clicks-evade-detection-215641>

What does XOR mean?



“A logical operator that returns a true value if one, but not both, of its operands is true. Also called exclusive OR.”

(<http://www.thefreedictionary.com/XOR>)

X	Y	Output
0	0	0
0	1	1
1	0	1
1	1	0

0	1	0	0	0	0	0	1
0	0	1	1	0	1	1	1
0	1	1	1	0	1	1	0

0x41 = 'A'

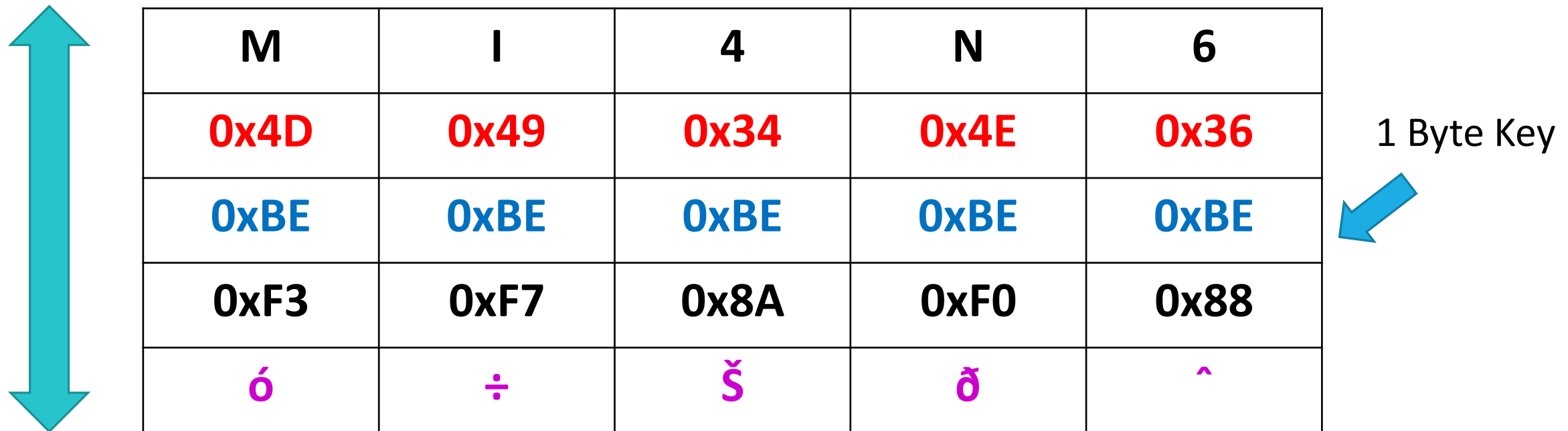
0x37 = '7'

0x76 = 'v'

Obfuscation/encryption with XOR

XOR ciphers can be easily created in software to obfuscate data

Example:



M	I	4	N	6
0x4D	0x49	0x34	0x4E	0x36
0xBE	0xBE	0xBE	0xBE	0xBE
0xF3	0xF7	0x8A	0xF0	0x88
ó	÷	š	ð	^

1 Byte Key

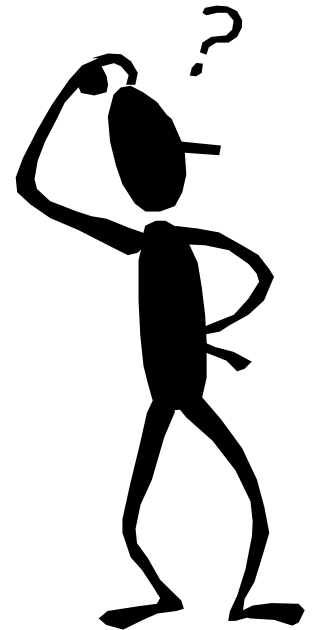
How does this apply to us?

String/file obfuscation (malware analysis, etc...)

Computer forensics

Mobile and Web

Network Communications





File Analysis Tools



String Analysis

“Strings” application sometimes used as first step to analyze malware

Can lead to indicators for incident response

**** Demo ****

File1 = Mal/Dwnldr-Y



SHA256: 64640b1027a45884227554e5aebbbf706278ab1c9254cfe181fcc9324f4bbde2

File name: Downloader.exe

Detection ratio: 35 / 46

Analysis date: 2013-04-12 10:03:13 UTC (1 month ago)




More details

File2 = Troj/LdMon-A



SHA256: d3dcf3912a2675c2fc766c6855c181aeee70cadde8956b1bb90950c5e3324190

File name: hamachi.exe

Detection ratio: 24 / 46

Analysis date: 2013-04-11 14:25:06 UTC (1 month ago)


More details



XORSearch

Developed by Didier Stevens

Additional information at -> <http://blog.didierstevens.com/programs/xorsearch/>

```
Usage: XORSearch [-siuh] [-l length] [-n length] [-f  
search-file] file string
```

```
XORSearch V1.8, search for a XOR, ROL, ROT or SHIFT  
encoded string in a file
```

**** Demo ****

XORTool

Tool written by “hellman” (<https://github.com/hellman/xortool>)

Features:

- Guess multibyte key length

- Guess key based on most used character knowledge

Additional information here:

<http://www.aldeid.com/wiki/Xortool>



**** Demo ****

XOR Analyze

Developed by Thomas Habets

XOR encryption, decryption and analysis features

Utilizes coincidence counting and frequency tables for analysis

References:

<http://www.habets.pp.se/synscan/programs.php?prog=xor-analyze>

<https://github.com/ThomasHabets/xor-analyze>

**** Demo ****

Hex Editors

Hex editors can be useful for analyzing obfuscated files.

Two good commercial editors with XOR capabilities:

- <http://www.sweetscape.com/010editor>
- <http://www.hhdsoftware.com/hex-editor>

**** Demo ****



Additional Tools

- XORStrings - <http://blog.didierstevens.com/2013/04/15/new-tool-xorstrings/>
- xorBruteForcer - <http://eternal-todo.com/var/scripts/xorbruteforcer>
- iheartxor - <http://hooked-on-mnemonics.blogspot.com/p/iheartxor.html>
- NoMoreXOR - <https://github.com/hiddenillusion/NoMoreXOR>
- unXor - <https://github.com/tomchop/unxor>
- Converter - <http://www.kahusecurity.com/tools/>

Recent blog post by Lenny Zeltser: <http://computer-forensics.sans.org/blog/2013/05/14/tools-for-examining-xor-obfuscation-for-malware-analysis>

Malware

Trojan.Cookies

"Communication with the Command & Control (C2) servers uses a combination of single-byte XOR and Base64 encoded data in the Cookie and Set-Cookie HTTP header fields."

<http://contagiodump.blogspot.com/2013/03/mandiant-apt1-samples-categorized-by.html>

Neutrino Exploit Kit

<http://malforsec.blogspot.com/2013/05/neutrino-exploit-kit-analysis.html>

Spyeye

Evidence of xor obfuscated configuration file

Computer Forensic Tools



Windows Forensics

XOR (& ROT) obfuscation plays a role (however, probably not the biggest role)

EnScripts

<http://www.forensickb.com/2011/05/encase-encrypt-to-search-for-keyword.html>

<http://www.forensickb.com/2008/03/xor-entire-file-or-selected-text.html>

Windows Search forensics

<http://www.forensicfocus.com/Content/pid=371/>

UserAssist Key

<http://www.aldeid.com/wiki/Windows-userassist-keys>



Search Selected files for keyword in ROT13...

Select what type of search:

☒ XOR

☐ ROT13

Keyword to search for:

OK Cancel

	Bookmark Type	Preview	Comment
<input type="checkbox"/> 26	Highlighted Data	B . \9]vv, F / 1 (+ (:	XOR Key: 124
<input type="checkbox"/> 27	Highlighted Data	\ : ?F, \: ?/580#,.3;.=1	XOR Key: 124
<input type="checkbox"/> 28	Highlighted Data	B . \9]vv, F / 1 (+ (:	XOR Key: 124
<input type="checkbox"/> 29	Highlighted Data	= \ / \1 , / ?/580#?31132#/(=.	XOR Key: 124
<input type="checkbox"/> 30	Highlighted Data	\Q\2 , , \1 / #(+	XOR Key: 124
<input type="checkbox"/> 31	Highlighted Data	= \ / \1 , / ?/580#?31132#/(=.	XOR Key: 124
<input type="checkbox"/> 32	Highlighted Data	\dó\ < 5÷ } 0 CC ?F, \: ? \: (*	XOR Key: 124
<input type="checkbox"/> 33	Highlighted Data	= \ / \1 , ?/580#?31132#,.3;.=1/ ?	XOR Key: 124
<input type="checkbox"/> 34	Highlighted Data	?F, \: Y, \: Y ?F (91,Y(91,Y ?F	XOR Key: 124
<input type="checkbox"/> 35	Highlighted Data	6 8 / \1 , / ?/580#/(=.0,	XOR Key: 124
<input type="checkbox"/> 36	Highlighted Data	Y/%/(91.33(Y ?F, \: Y, :	XOR Key: 124
<input type="checkbox"/> 37	Highlighted Data	6 8 / \1 , ?/580#,.3;.=1/ ?F 2	XOR Key: 124
<input type="checkbox"/> 38	Highlighted Data	Y/%/(91.33(Y ?F, \: Y, :	XOR Key: 124
<input type="checkbox"/> 39	Highlighted Data	\ . \9]vv, F @ \ \	XOR Key: 124
<input type="checkbox"/> 40	Highlighted Data	\dó\ < 5÷ } 0 CC ?F, \: ? \: (*	XOR Key: 124
<input type="checkbox"/> 41	Highlighted Data	\ . \9]vv, F @ \ \	XOR Key: 124
<input type="checkbox"/> 42	Highlighted Data	, "/o€€€-£ÄøääððéññßÚéæððöçöäí€€€" jEVo,18€Ä€€-iÎÓÉÓß	XOR Key: 128
<input type="checkbox"/> 43	Highlighted Data	fff{fffoe9f7ŠNç;İNç×ëëððöññäññäEääññ÷EääEññóíEéíEQİĐEñç	XOR Key: 131
<input type="checkbox"/> 44	Highlighted Data	fff{fffoe9f7ŠNç;İNç×ëëððöññäññäEääññ÷EääEññóíEéíEQİĐEñç	XOR Key: 131
<input type="checkbox"/> 45	Highlighted Data	fff{fffoe9f7ŠNç;İNç×ëëððöññäññäEääññ÷EääEññóíEéíEQİĐEñç	XOR Key: 131
<input type="checkbox"/> 46	Highlighted Data	fff{fffoe9f7ŠNç;İNç×ëëððöññäññäEääññ÷EääEññóíEéíEQİĐEñç	XOR Key: 131
<input type="checkbox"/> 47	Highlighted Data	fff{fffoe9f7ŠNç;İNç×ëëððöññäññäEääññ÷EääEññóíEéíEQİĐEñç	XOR Key: 131
<input type="checkbox"/> 48	Highlighted Data	fffsfffoe9f7ŠNç;İNç×ëëððöññäññäEääññ÷EääEññóíEéíEQİĐEñç	XOR Key: 131

Images from: <http://www.forensickb.com/2011/05/encase-enscript-to-search-for-keyword.html>

Mobile and Web



Mobile and Web

Sometimes XOR is used poorly in mobile applications...

Storing usernames and passwords with XOR obfuscation is a bad idea. However, Evernote for Android did this (until recently)

<http://arstechnica.com/security/2013/03/critics-substandard-crypto-needlessly-puts-evernote-accounts-at-risk/>



<Truncated>

Copyright 2013 Adam Caudill <adam@adamcaudill.com>

<Truncated>

```
require "base64"
```

```
if ARGV.count != 2
```

```
  puts 'Usage: ./evernote_pass_decode.rb <pass> <username>'
```

```
end
```

```
pass = Base64.decode64(ARGV[0])
```

```
user = ARGV[1]
```

```
final = ''
```

```
pass.bytes.each_with_index do |byte, index|
```

```
  final += (byte ^ user[index % user.length].unpack('c')[0]).chr
```

```
end
```

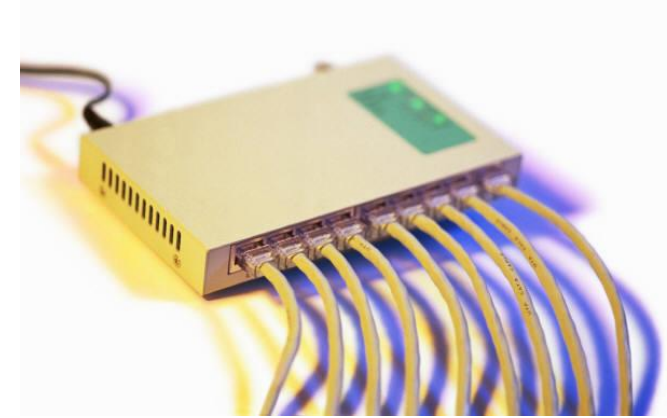
```
puts "Password: #{final}"
```

<https://gist.github.com/adamcaudill/5079342>

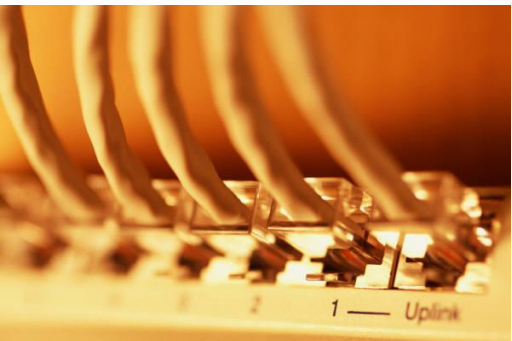
Java Based Malware

Websense's analysis of Flashback malware shows the exploit trigger string is xor'd by single character key

<http://community.websense.com/blogs/securitylabs/archive/2012/04/16/is-the-cve-2012-0507-the-best-toolkit-to-exploit-mac-os-x.aspx>



Network Based Tools



ChopShop

Network based analysis tool

“...framework to aid analysts in the creation and execution of pynids based decoders and detectors...”

Developed and released by Mitre

<https://github.com/MITRECN/ChopShop>

**** DEMO ****

(Demo based on: http://www.mitre.org/work/cybersecurity/blog/cyber_tools_shields4.html)

XOR obfuscated data

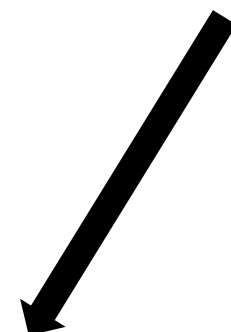
```
HTTP/1.1 200 OK
Server: nginx/1.1.19
Date: Wed, 15 May 2013 00:13:24 GMT
Content-Type: text/html
Content-Length: 16
Last-Modified: Wed, 15 May 2013 00:13:08 GMT
Connection: keep-alive
Accept-Ranges: bytes
```



EYXBXB\]FPCT;

```
[2013-05-15 15:13:24 EDT] 127.0.0.1:80 -> 127.0.0.1:51391 231 bytes
0000000: 79 65 65 61 1e 00 1f 00 11 03 01 01 11 7e 7a 3c |yeea.....~z<|
0000010: 3b 62 54 43 47 54 43 0b 11 5f 56 58 5f 49 1e 00 |;bTCGTC.._VX_I..|
0000020: 1f 00 1f 00 08 3c 3b 75 50 45 54 0b 11 66 54 55 |.....<;uPET..fTU|
0000030: 1d 11 00 04 11 7c 50 48 11 03 01 00 02 11 00 08 |.....|PH.....|
0000040: 0b 00 02 0b 03 05 11 76 7c 65 3c 3b 72 5e 5f 45 |.....v|e<;r^_E|
0000050: 54 5f 45 1c 65 48 41 54 0b 11 45 54 49 45 1e 59 |T_E.eHAT..ETIE.Y|
0000060: 45 5c 5d 3c 3b 72 5e 5f 45 54 5f 45 1c 7d 54 5f |E\]<;r^_ET_E.}T_|
0000070: 56 45 59 0b 11 00 07 3c 3b 7d 50 42 45 1c 7c 5e |VEY....<;}PBE.|^|
0000080: 55 58 57 58 54 55 0b 11 66 54 55 1d 11 00 04 11 |UXWXTU..fTU.....|
0000090: 7c 50 48 11 03 01 00 02 11 00 08 0b 00 02 0b 01 ||PH.....|
00000a0: 09 11 76 7c 65 3c 3b 72 5e 5f 5f 54 52 45 58 5e |..v|e<;r^__TRESX^|
00000b0: 5f 0b 11 5a 54 54 41 1c 50 5d 58 47 54 3c 3b 70 |_..ZTTA.P]XGT<;p|
00000c0: 52 52 54 41 45 1c 63 50 5f 56 54 42 0b 11 53 48 |RRTAE.cP_VTB..SH|
00000d0: 45 54 42 3c 3b 3c 3b 74 68 69 73 20 69 73 20 6d |ETB<;<this is m|
00000e0: 61 6c 77 61 72 65 0a |alware;. |
```

decoded data



StreamDB

“StreamDB is a high-performance framework for storing network streams.”

Some support for XOR de-obfuscation of suspected executables

<http://code.google.com/p/streamdb/>

References

Practical Malware Analysis (chapter 13)

Malware Analyst's Cookbook and DVD (chapter 12)