BlackBerry Blog

BlackBerry Blog › Inside the APT28 DLL Backdoor Blitz

Inside the APT28 DLL Backdoor Blitz

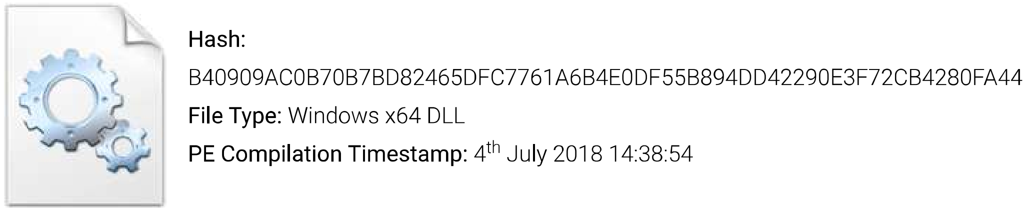
RESEARCH & INTELLIGENCE / 08.28.19 / The BlackBerry Cylance Threat Research Team

Inside the APT28 DLL Backdoor Blitz

This blog post is a follow-up to *Flirting With IDA and APT28*, where we covered generating IDA Pro signatures to identify and eliminate benign library code. In doing so, we can focus our attention on the bespoke code responsible for defining the implant's behavior.

This time around we perform a deep dive into the same APT28 sample by analyzing its capabilities and providing insight into its features.

Overview



Hash:
B40909AC0B70B7BD82465DFC7761A6B4E0DF55B894DD42290E3F72CB4280FA44
File Type: Windows x64 DLL
PE Compilation Timestamp: 4th July 2018 14:38:54

Figure 1: APT28 sample details

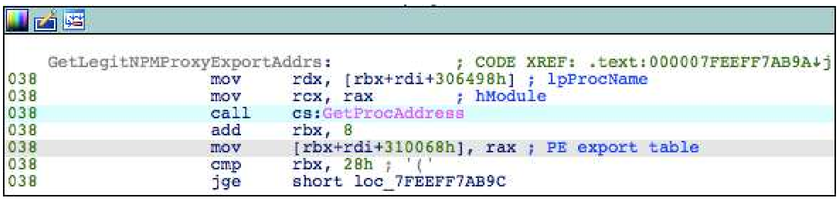
Analysis reveals the implant is a multi-threaded DLL backdoor that gives the threat actor (TA) full access to, and control of, the target host. When commanded by C2, the implant can upload or download files, create processes, interact with the host via a command shell and connect to C2 according to a defined sleep/activity schedule.

As covered in Part 1, the implant is written in C++ and statically linked against OpenSSL and the Poco C++ framework. Since the file is packaged as a DLL, the intention would be to inject it into a long-running process that is granted Internet access (such as a NetSvc service group) or one having local firewall permissions. We do not believe this DLL is intended to operate as a module for a larger tool.

Entry Point

Malicious code entry occurs via the DllMain export. The implant's first task is to load the legitimate Microsoft npmproxy.dll found in "c:\windows\system32". With this loaded, the addresses of the five implant exports, "DllCanUnloadNow", "DllGetClassObject", "DllRegisterServer", "DllUnregisterServer" and "GetProxyDllInfo" are set to the addresses of the same, benign exports found in the Microsoft DLL.

The reason for this is unconfirmed, but most likely serves a defensive measure to evade end point protection that scans export addresses looking for code changes (when compared to a predetermined signature database), or against a database of known-malicious code. Perhaps by setting its own exports to those of the benign copy, the implant is attempting to remain undetected:



```
    GetLegitNPMProxyExportAddrs:                ; CODE XREF: .text:000007FEEFF7AB9A↓j
038                   mov     rdx, [rbx+rdi+306498h] ; lpProcName
038                   mov     rcx, rax            ; hModule
038                   call    cs:GetProcAddress
038                   add     rbx, 8
038                   mov     [rbx+rdi+310068h], rax ; PE export table
038                   cmp     rbx, 28h ; '('
038                   jge     short loc_7FEEFF7AB9C
```

Figure 2: Substitution of PE export addresses

With export addresses replaced, a thread is launched to execute the main code path. Within this, a second thread is launched responsible for send and receive operations with C2. A global variable forms the basis of inter-thread synchronization for processing of C2 messages.

Mutex

The primary thread's first task is the creation of a Globally Unique ID (GUID) mutex "1b8232f6-6806-4733-901d-62bf3ef33e6c". The GUID string can be found as plain text within the binary, offering a potential (albeit trivially replaceable) IoC. As is customary, if mutex creation fails the sample terminates with no further action taken.

Machine Fingerprint

Following mutex creation, the implant generates a unique CRC-32 host fingerprint using the primary network interface's ethernet (MAC) address, the host name, and Windows version string. The final CRC-32 result is then XOR'd with fixed constant 0x64113. The generated host fingerprint is used later to build the C2 beacon URL, and presumably identifies the implant's malicious use on each infected machine.

Command-and-Control (C2)

The main communication protocol with the C2 server is RSA encrypted, Base64 encoded JSON exchanged over HTTPS (TLS, port 443), or as fallback, on plain HTTP using port 80. During our analysis the implant did not attempt to use any other application protocols.

The user-agent string for HTTP requests is populated from either the return value of the "ObtainUserAgentString" API call, or if that fails, a hard-coded alternate (see IoCs below).

Support for egress proxy servers is included through a call to "WinHttpGetIEProxyConfigForCurrentUser". The result is parsed and supplied to the Poco framework's "HTTPClientSession::setProxyConfig" prior to C2 activity.

A 1024-bit RSA key pair is embedded within the implant and used to encrypt and decrypt communication between host and C2. The private key decrypts inbound traffic while the public key encrypts outbound. No stream cipher(s) or session keys are negotiated.

The initial HTTPS beacon to "malaytravelgroup[.]com" is a GET request consisting of a randomly selected URL path concatenation, together with a query string containing the parameter split count, the XOR constant 0x64113 (also used when generating the host CRC-32), and the computed CRC-32 fingerprint. The query string is padded with random data to prevent ad-hoc analysis and finally Base64 encoded:

Field	Name	Bytes	Value	Comment
1	-	1	1-254	Parameter split count as ASCII value.

2	"aid"	4	0x64113	XOR constant used when generating host CRC-32. Possibly a unique implant ID.
3	"bid"	4	*variable*	CRC-32 host fingerprint.

Table 1: C2 beacon query parameters

The URL path component is built as one to three strings ("/" separated), generated from a set of encrypted string tables. The decrypted versions are provided in the Appendix.

The Base64 encoded data is split into a random number of parameters and padded as a means of obfuscation. The query parameter's names are randomly generated using 1-4 upper/lower case characters:

Name	Value	Explanation
<rand_str>	<2_rand_chars> + b64(<n>)	Specifies the number of parameters across which the beacon data is split
<rand_str_1>	<2_rand_chars> + b64(<rand_byte> + <data_part_1>)	Value contains first part of encoded data, preceded by 2 random characters
<rand_str_2>	<2_rand_chars> + b64(<rand_byte> + <data_part_2>)	Value contains second part of encoded data, preceded by 2 random characters
...	...	...
<rand_str_*n*>	<2_rand_chars> + b64(<rand_byte> + <data_part_*n*>)	Value contains last part of encoded data, preceded by 2 random characters

Table 2: Query parameter deconstruction

Examples of beacon request URLs:

```
GET /pricing/training/news/?
GPFi=mLMg&CYlp=Tj9RNBBg%3D%3D&JOM=uJfgAz9Zpw

GET /forum/feedback/switch/?
LnY=YNA&D=TH%2FRM%3D&rMuo=BFXUEG&cs=bkqgA%3D&HZql=bXgTP1mnA%3D
HTTP/1.1

GET /activity/?
FLVE=JNA&Y=vKYxNB&F=Hg6wY%3D&Slq=QBuAAz&ux=lSGfWacA%3D%3D
HTTP/1.1
```

Figure 3: Beacon URL examples

Using the first example in Figure 3, the first query parameter indicating the beacon data split count would be Base64 decoded as:

GPFi=mL**Mg** >> b64_decode(**Mg**) >>0x32 ("2")

All successive query parameters have the first two characters removed prior to Base64 decoding. The first byte of the decoded result is then discarded to arrive at the final value:

CYlp=Tj**9RNBBg==** >> b64_decode(**9RNBBg==)** >> F5 13 41 06
JOM=uJ**fgAz9Zpw** >> b64_decode(**fgAz9Zpw)** >> 7E 00 33 F5 9A 70

0x00064113 >> XOR constant/implant ID

0x709AF533>> Host CRC-32 fingerprint

Once the beacon is received and decoded, the next phase of interaction is to issue commands that drive the implant's functions, such as the provision of an activity schedule/sleep interval limiting implant activity.

Domain Generation Algorithm (DGA)

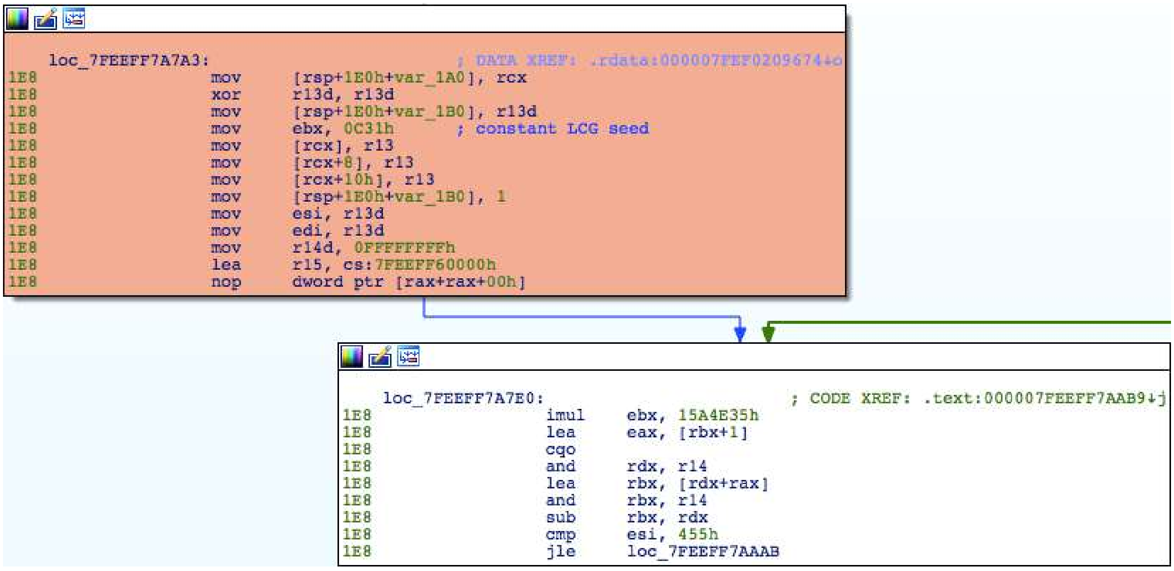
The primary C2 domain of "malaytravelgroup[.]com" is used for the initial beacon. Like all sensitive strings, it is XOR encoded and revealed only when needed.

The implant also contains a Linear Congruential Generator (LCG) based algorithm for generating what appear to be backup C2 domain names. Curiously the implementation is 100% deterministic owing to the use of a fixed seed value (0xC31) as shown in Figure 4. Every invocation of the DGA will produce the same five domains:

schooltillhungryprocess[.]com
reasonwithusefulpolicy[.]com
streetunderrelevantpeople[.]com
experiencewithweakkid[.]com
systembeforeniceparent[.]com

Figure 4: Encoded C2 servers

Once the domains are generated, they are XOR encoded. The LCG implementation can be seen here, together with its constant seed value:



```
    loc_7FEEFF7A7A3:                                     ; DATA XREF: .rdata:000007FEF0209674↓o
1E8                 mov     [rsp+1E0h+var_1A0], rcx
1E8                 xor     r13d, r13d
1E8                 mov     [rsp+1E0h+var_1B0], r13d
1E8                 mov     ebx, 0C31h      ; constant LCG seed
1E8                 mov     [rcx], r13
1E8                 mov     [rcx+8], r13
1E8                 mov     [rcx+10h], r13
1E8                 mov     [rsp+1E0h+var_1B0], 1
1E8                 mov     esi, r13d
1E8                 mov     edi, r13d
1E8                 mov     r14d, 0FFFFFFFFh
1E8                 lea     r15, cs:7FEEFF60000h
1E8                 nop     dword ptr [rax+rax+00h]
```

```
    loc_7FEEFF7A7E0:                                     ; CODE XREF: .text:000007FEEFF7AAB9↓j
1E8                 imul    ebx, 15A4E35h
1E8                 lea     eax, [rbx+1]
1E8                 cqo
1E8                 and     rdx, r14
1E8                 lea     rbx, [rdx+rax]
1E8                 and     rbx, r14
1E8                 sub     rbx, rdx
1E8                 cmp     esi, 455h
1E8                 jle     loc_7FEEFF7AAAB
```

Figure 5: DGA LCG using fixed seed value

Domain names are generated as the concatenation of 4 string-table lookups from 3 encrypted string tables. Table 3 is referenced twice during generation. The Top-Level Domain (TLD) suffix is hardcoded to ".com".

Table	Number of strings
1	99
2	31
3	149

Table 3: String table sizes for domain generation

Assuming the continued adoption of a four-part domain name, the LCG would be capable of generating millions of unique permutations. The reason for limiting its output to the same five, irrespective of date, time, host identity, or other input variable is unclear. A copy of the decrypted string tables is provided in the Appendix.

By manually decreasing the delay between check-in attempts, we accelerated the amount of C2 activity generated by the implant. Curiously, during a 24-hour observation window, no attempts were made to communicate with the generated domains. Different reachability scenarios were tested, ranging from malformed responses to unreachable IP addresses. At no point did the implant generate any DNS or HTTP traffic relating to the generated domains.

It's conceivable the backup domains are accessed via HTTP 302 redirects issued by the primary site, but this would seemingly defeat the purpose of having primary and backup domains to cater for failure, migration, or shutdown. It's also possible the generated C2 domains are active only for certain implant commands such as file upload or download.

Lacking any evidence for their direct use, we can speculate the generated domains may be operating as a honeypot. Access to them from any Internet addresses would immediately be considered "suspicious" by virtue of their hidden existence only within the implant code. The source of such requests could be directly attributable to threat researchers or analysts investigating the implant's inner workings.

Functions

The implant lacks any modularity and is therefore limited to eight functions. Each function is invoked from C2 by supplying a value from which a known CRC-32 is calculated. The computed CRC-32 value serves as a key or look-up to execute the corresponding function.

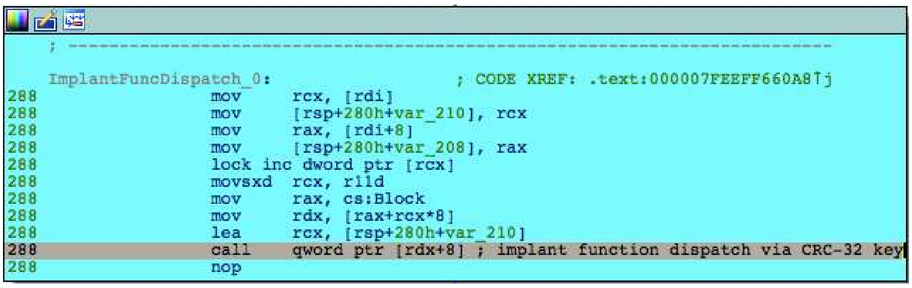
Parameters for each backdoor supported function are supplied with each command and detailed in Table 4:

CRC-32	Function	Parameters	Description
15512FB9	Spawn process	file_name, file_path, args	Spawn a process using the supplied arguments.
76BCDC67	Create file	file_name, file_path, body	Creates a file with the supplied body, i.e. file is pushed (downloaded) from C2. Files have the DOS hidden attribute set by default.
B21CEBD4	Delete file/folder	file_name, file_path, recursive	Deletes file or folder. Optional recursive parameter for folder and contents deletion.
406C9E35	Launch shell	start, stop, status, bash	Launches COMSPEC (cmd.exe) shell. Stdin, stdout, stderr relayed over main C2 channel.
72E99D37	Read file	file_name, file_path	Read a file, i.e. upload to C2.
04365407	Set C2 check-in interval	timeout	Sets the interval between C2 check-in. Default is 3 hours.
99598005	Set C2 check-in/activity schedule	day_of_month, day_of_week, hour,	Sets day of month/week and hour that implant should attempt to reach C2. Default setting is 0 for all values, meaning no exclusions; check-in would therefore be every 3 hours – the default sleep interval.
1E46EC18	Get status information	*None*	Returns Build_Id of implant, Computer_Id, Full_Info. Details of last command and status (success/fail).

Table 4: Implant functions

The shell capability is perhaps the most interesting, giving the operators a full-duplex, interactive COMSPEC (cmd.exe) session, with stdin (input), stdout, and stderr (output returned by commands) redirected over the HTTPS C2 channel. Commands to the shell are issued by the TA with the resulting output being returned almost immediately.

Despite the shell pipe creation taking place in (as identified by IDA signatures) the Poco framework's "SMTPChannel" constructor, the implant makes no use of SMTP as a transport. The "bash" parameter, synonymous with Linux environments, hints at a multi-platform capability. However, in this instance it is used to send commands to the shell by writing the accompanying value to stdin. Windows 10 does offer a Linux Subsystem that would make a bash shell available, but we have no evidence to suggest this is what the operators are installing or using:



```
    ; ---------------------------------------------------------------------------

    ImplantFuncDispatch_0:                           ; CODE XREF: .text:000007FEEFF660A8↑j
288                     mov     rcx, [rdi]
288                     mov     [rsp+280h+var_210], rcx
288                     mov     rax, [rdi+8]
288                     mov     [rsp+280h+var_208], rax
288                     lock inc dword ptr [rcx]
288                     movsxd  rcx, r11d
288                     mov     rax, cs:Block
288                     mov     rdx, [rax+rcx*8]
288                     lea     rcx, [rsp+280h+var_210]
288                     call    qword ptr [rdx+8] ; implant function dispatch via CRC-32 key
288                     nop
```

Figure 6: Implant function dispatch

String Encryption

Thirteen unique XOR keys are used throughout the implant for runtime decoding of sensitive strings. Each implant function (as listed in Table 4) is provided with its own XOR key, as are general C2, URL/HTTP, and RSA related operations:

Key:	Used during:
F226E34F0B64528CF0	Embedded RSA key decryption
371F41ABE4C8880BC1	URL/HTTP related operations
40A521B0866411BAC4	General string decryption key
EFEA221C400E2D1227	C2 messaging
82EF8E95992055B6B5	C2 messaging
4F840D589769	Create process function
6EFEAF1AAA6932	Create file/download function
59938EF1C8EFBA79B	Delete file/folder function
722C1B76	Shell handler function
1F4582D80B	Read file/upload function
6A39E0FC68D6	Set C2 check-in interval function
A24B726DFD	Set C2 activity schedule function
597EB47AAD	Get host/install info function

Table 5: XOR encoding/decoding keys

Conclusion

Analysis shows the implant carries a feature set designed to provide the fundamental capabilities of a backdoor: file download, upload, remote command execution, and interactive shell access.

Comparing the functions of this implant to published descriptions of APT-28's "x-tunnel", we find no tunneling, proxying, or VPN-like capabilities. It also lacks credential harvesting, network service scanning, or Windows registry manipulation. Given the lack of modularity only an updated version could provide these features. The alternative (and perhaps more likely) scenario would be for such capabilities to exist in subsequent tools downloaded and executed by this implant.

Indicators of Compromise (IoCs)

SHA256:

b40909ac0b70b7bd82465dfc7761a6b4e0df55b894dd42290e3f72cb4280fa44

C2 beacon domain:

malaytravelgroup[.]com

DGA domains:

schooltillhungryprocess[.]com
reasonwithusefulpolicy[.]com
streetunderrelevantpeople[.]com
experiencewithweakkid[.]com
systembeforeniceparent[.]com

Mutex:

1b8232f6-6806-4733-901d-62bf3ef33e6c

Hard-coded User-Agent string:

"Mozilla/5.0 (Windows NT 6.3; WOW64; rv:28.0) Gecko/20100101 Firefox/28.0"

Yara Signature

```
rule apt28_backdoor_cls
{
        strings:
        $st1 = "AES_256_poco" ascii
                $st2 = "TEncryption" ascii
                $st3 = "shell" ascii
        condition:
                all of them
}

rule apt28_backdoor_crc32
{
          strings:
                    $xor1 = { 48 8B 07 39 48 0C 75 3A 44 8B 70 08 4C 8B 38 4D 85 C0 74 2E 45 85 E4 74 29 }
          condition:
          $xor1
}
```

Appendix

Decrypted String Tables:

Table 1	Table 2	Table 3

different	at	street
used	on	company
important	in	part
every	to	system
large	into	number
available	from	world
popular	before	case
lonely	until	work
basic	till	party
known	about	girl
various	for	house
difficult	of	woman
several	with	life
unite	by	people
historical	after	year
hot	since	day
useful	during	way
mental	between	thing
scare	near	child
additional	nearby	group
emotional	behind	time
old	across	area
political	above	problem
similar	over	place
healthy	under	hand
financial	below	service
medical	along	school
traditional	round	guy
former	around	country
entire	past	point
strong	through	week
actual		relationship
significant		end
successful		word
electrical		family
expensive		fact
pregnant		head
intelligent		month
interesting		information
poor		power
happy		change
responsible		question
cute		business
helpful		development
recent		home
willing		side
nice		night
wonderful		money
impossible		eye
serious		interest
huge		book
rare		teacher
visible		air
typical		court
competitive		water

critical
desperate
immediate
aware
educational
environmental
global
decent
relevant
accurate
capable
dangerous
dramatic
efficient
powerful
foreign
hungry
friendly
psychological
severe
suitable
numerous
sufficient
unusual
anxious
cultural
curious
famous
pure
comprehensive
obvious
careful
impressive
unhappy
acceptable
aggressive
boring
inner
eastern
sudden
reasonable
strict
weak
civil

manager
form
food
ca
moment
level
room
car
story
market
effect
idea
opportunity
result
use
study
job
name
report
body
law
face
authority
friend
parent
minute
door
minister
road
rate
line
hour
war
mother
right
office
person
reason
view
term
period
centre
morning
project
research
figure
society
history
city
police
kind
million
community
need
tree
price

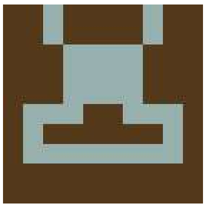
			team game father kid student support program health field man example quality control action process position education age course type manner order decision industry mind condition paper bank century activity table sense building experience staff language plan policy

Table 6: C2 DGA Strings

Table 1

news
nwshp
section
pagead
ads
forum
general
topic
master
explore
features
enterprise
pricing
article
contact
security
about
status
blog
training
help
feedback
terms
activity
pricing
switch

Table 7: URL Path String Table



About The BlackBerry Cylance Threat Research Team

The BlackBerry Cylance Threat Research team examines malware and suspected malware to better identify its abilities, function and attack vectors. Threat Research is on the frontline of information security and often deeply examines malicious software, which puts us in a unique position to discuss never-seen-before threats.

BACK

Corporate

Company

Newsroom

Investors

Careers

Leadership

Corporate Responsibility

Certifications

Customer Success

© 2024 BlackBerry Limited. All rights reserved.