

Informe Código Dañino

CCN-CERT ID-12/21

RansomEXX_Linux ransomware



Septiembre 2021

Edita:



© Centro Criptológico Nacional, 2019

Fecha de Edición: septiembre de 2021

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

Quedan rigurosamente prohibidas, sin la autorización escrita del Centro Criptológico Nacional, bajo las sanciones establecidas en las leyes, la reproducción parcial o total de este documento por cualquier medio o procedimiento, comprendidos la reprografía y el tratamiento informático, y la distribución de ejemplares del mismo mediante alquiler o préstamo públicos.

ÍNDICE

1. SOBRE CCN-CERT.....	4
2. RESUMEN EJECUTIVO.....	5
3. DETALLES GENERALES	6
4. CARACTERÍSTICAS TÉCNICAS	8
5. CIFRADO	10
6. DESINFECCIÓN	14
7. MITIGACIÓN	14
8. REGLAS DE DETECCIÓN	15
8.1. REGLAS DE YARA	15
9. REFERENCIAS	16

1. SOBRE CCN-CERT

El CCN-CERT es la Capacidad de Respuesta a incidentes de Seguridad de la Información del Centro Criptológico Nacional, CCN, adscrito al Centro Nacional de Inteligencia, CNI. 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 CNI, el RD 421/2004 de regulación del CCN y en el RD 3/2010, de 8 de enero, regulador del Esquema Nacional de Seguridad (ENS), modificado por el RD 951/2015 de 23 de octubre.

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, incluyendo la coordinación a nivel público estatal de las distintas Capacidades de Respuesta a Incidentes o Centros de Operaciones de Ciberseguridad existentes.

Todo ello, con el fin último de conseguir un ciberespacio más seguro y confiable, preservando la información clasificada (tal y como recoge el art. 4. F de la Ley 11/2002) y la información sensible, defendiendo el Patrimonio Tecnológico español, formando al personal experto, aplicando políticas y procedimientos de seguridad y empleando y desarrollando las tecnologías más adecuadas a este fin.

De acuerdo a esta normativa y la Ley 40/2015 de Régimen Jurídico del Sector Público es competencia del CCN-CERT la gestión de ciberincidentes que afecten a cualquier organismo o empresa pública. En el caso de operadores críticos del sector público la gestión de ciberincidentes se realizará por el CCN-CERT en coordinación con el CNPIC.

2. RESUMEN EJECUTIVO

El presente documento recoge el análisis de la muestra de código dañino identificada por la firma MD5 **22f2ed67b2aec55d54c9864f98e0ab85** compilada para plataformas Linux (x64). El binario analizado pertenece a la familia de *ransomware* apodada como EXX o RansomEXX, también conocido como Defray777 o RansomX. Este *ransomware*, vinculado con el actor GOLD DUPONT¹, ha estado involucrado en múltiples incidentes desde 2018² y ha afectado a organismos gubernamentales y empresas críticas.

En mayo de 2020 el sistema judicial³ así como el Departamento de Transporte del estado de Texas (TxDOT)⁴ se vio afectado por este *ransomware*. La multinacional japonesa Konica Minolta⁵ así como el sistema judicial de Brasil⁶ fueron también objetivo de RansomEXX en julio y noviembre, respectivamente, de ese mismo año. La empresa brasileña Embraer, considerada como uno de los mayores fabricantes de aviones civiles de la actualidad, no solo se vio afectada⁷ por RansomEXX a finales de 2020, sino que sus datos fueron filtrados después de que el fabricante de aviones se negara a ceder ante la, cada vez más común, doble extorsión⁸. En el último año, determinados grupos de atacantes han utilizado, como vía de entrada, ciertas vulnerabilidades en productos VMWare ESXi para cifrar⁹ sus discos duros virtuales mediante EXX.

RansomEXX se caracteriza por utilizar la librería *open-source mbed TLS*¹⁰, tanto en su variante Windows como en Linux, para llevar a cabo las labores de cifrado. Dicha librería es empleada para generar claves AES con las que cifrar los ficheros de la víctima usando el cifrado por bloques AES-256 en modo ECB (*Electronic CodeBook*). Posteriormente, cada clave AES es cifrada con una clave pública RSA-4096, embebida en el cuerpo del binario, que será adjuntada a cada archivo cifrado (cuya extensión, para la muestra actual, es *“.31gs1-”* seguido de 8 caracteres en hexadecimal). A diferencia de la variante EXX para entornos Windows¹¹, la muestra analizada no implementa funcionalidades adicionales más allá que el propio cifrado de los ficheros (no finaliza el espacio libre en disco, no emplea técnicas anti-análisis y no finaliza servicios y procesos).

3. DETALLES GENERALES

El fichero analizado se corresponde con una muestra de *ransomware* para entornos Linux (ELF64) de la familia RansomEXX cuya firma se muestra a continuación:

MD5	22f2ed67b2aec55d54c9864f98e0ab85
SHA1	3f78e2c6fa101d03b338ea9a750c0b4ccfe8bd95
SHA256	63de4a85b9d2cc3670987b284614ed54f3ac6c7d2e09c0fff7b0747d79200281

Cabe destacar que este *sample* había sido remitido, por primera vez, a las plataformas de análisis de malware¹² el 23 de junio de 2021.

El binario presenta un tamaño de 212.264 bytes y ha sido compilado (*dynamically linked*) con GCC para arquitecturas de 64 bits. La siguiente imagen muestra su *ELF Header* así como algunas de las propiedades estáticas del mismo.

```
(root@CCNLAB)~# file exx
exx: ELF 64-bit LSB shared object, x86_64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=ed2011c4f89d51d0c86833f66a866b12929ab5e5, for GNU/Linux 3.2.0, not stripped
(root@CCNLAB)~# ldd exx
linux-vdso.so.1 (0x00007ffc8056c000)
libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f3840726000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f3840561000)
/lib64/ld-linux-x86-64.so.2 (0x00007f3840790000)
(root@CCNLAB)~# readelf -h exx
ELF Header:
  Magic:   7f 45 4c 46 02 01 00 00 00 00 00 00 00 00 00 00
  Class:             ELF64
  Data:              2's complement, little endian
  Version:           1 (current)
  OS/ABI:            UNIX - System V
  ABI Version:       0
  Type:              DYN (Shared object file)
  Machine:           Advanced Micro Devices X86-64
  Version:           0x1
  Entry point address: 0x3320
  Start of program headers: 64 (bytes into file)
  Start of section headers: 210280 (bytes into file)
  Flags:             0x0
  Size of this header: 64 (bytes)
  Size of program headers: 56 (bytes)
  Number of program headers: 11
  Size of section headers: 64 (bytes)
  Number of section headers: 31
  Section header string table index: 30
(root@CCNLAB)~# objdump -s --section .comment exx
exx:      file format elf64-x86-64

Contents of section .comment:
 0000 4743433a 20284465 6269616e 20392e33  GCC: (Debian 9.3
0010 2e302d31 38292039 2e332e30 00          .0-18) 9.3.0.
(root@CCNLAB)~#
```

OVERVIEW		Name	Size	Address	End Address	Virtual Size	Permissions	Entropy
Info		.comment	0x1d	0x00000000	0x0000001d	0x1d	----	4.14229522
File: /home/kali/exx		.shstrtab	0x114	0x00000000	0x00000114	0x114	----	4.24761899
Format: elf64		.strtab	0x24f	0x00000000	0x0000024f	0x24f	----	4.69677418
Bits: 64		.symtab	0x3738	0x00000000	0x00003738	0x3738	----	2.82438905
Base addr: 0x00000000		.interp	0x2c	0x00000248	0x00000274	0x2c	-r-x	3.94075983
Virtual addr: True		.note.gnu.build-id	0x24	0x00000244	0x00000268	0x24	-r--	4.24716720
OS: linux		.note.ABI-tag	0x20	0x00000248	0x00000308	0x20	-r--	1.56127812
Cnary: False		.gnu.hash	0x28	0x00000308	0x00000330	0x28	-r--	2.40207841
Crypto: False		.dynsym	0x4e0	0x00000330	0x00000810	0x4e0	-r--	0.89462690
Stripped: False		.dynstr	0x211	0x00000810	0x00000a21	0x211	-r--	4.59950626
Size: 207 kB		.gnu.version	0x68	0x00000a22	0x00000a8a	0x68	-r--	1.25292760
NX bit: True		.gnu.version_r	0x50	0x00000a90	0x00000ae0	0x50	-r--	2.48029886
Type: DYN (Shared object file)		.rela.dyn	0x2c8	0x00000ae0	0x00000748	0x2c8	-r--	2.38602026
PIE: True		.rela.plt	0x450	0x00000748	0x00000b98	0x450	-r--	1.68312045
Static: False		.init	0x7	0x00000300	0x00000307	0x7	-r-x	3.91486630
Relro: Partial		.plt	0x20	0x00000300	0x00000330	0x20	-r-x	4.11145119
Compiler: GCC (Debian 9.3.0-18)		.plt.got	0x8	0x00000330	0x00000338	0x8	-r-x	3.00000000
		.text	0x1feb0	0x00000330	0x00002310	0x1feb0	-r-x	5.78046672
		.fini	0x9	0x00002310	0x00002319	0x9	-r-x	2.50325833
		.rodata	0x387f	0x00002400	0x00002787	0x387f	-r--	6.29779965
		.eh_frame_hdr	0x84	0x00002780	0x00002844	0x84	-r--	3.47406091
		.eh_frame	0x2918	0x00002848	0x00002b40	0x2918	-r--	4.71468790
		.init_array	0x8	0x00002c50	0x00002c58	0x8	-r-w	0.54356444
		.fini_array	0x8	0x00002c58	0x00002c60	0x8	-r-w	1.06127812
		.data.rel.ro	0x778	0x00002c60	0x00002d48	0x778	-r-w	2.29701567
		.dynamic	0x1f0	0x00002d48	0x00002f18	0x1f0	-r-w	1.56095803
		.got	0x28	0x00002f18	0x00002f80	0x28	-r-w	0.00000000
		.got.plt	0x188	0x00002f80	0x00002e18	0x188	-r-w	1.76183505
		.data	0x108	0x00002e18	0x00002e28	0x108	-r-w	0.98557417
		.bss	0x0	0x00002e28	0x00003078	0x248	-r-w	
Hashes								
MD5: 22f2ed67b2aec55d54c9864f98e0ab85								
SHA1: 3f78e2c6fa101d03b338ea9a750c0b4ccfe8bd95								
SHA256: 63de4a85b9d2cc3670987b284614ed54f3ac6c7d2e09c0fff7b0747d79200281								
Entropy: 5.757405								
Analysis info								
Functions: 428								
X-Refs: 6818								
Calls: 4546								
Strings: 303								
Symbols: 450								
Imports: 48								
Analysis coverage: 143743 bytes								
Code size: 131545 bytes								
Coverage percent: 109%								

Figura 1. Información estática ELF64

El código dañado ha sido compilado estáticamente con la librería *open-source mbed TLS* y, a diferencia de su versión para Windows, la información del binario que genera el compilador no ha sido eliminada (*stripped*) facilitando en gran medida la ingeniería inversa. Entre la información de *debug* se identifican nombres de funciones, nombres de ficheros, así como variables globales que permiten atribuir fácilmente la muestra a la familia de *ransomware EXX*.

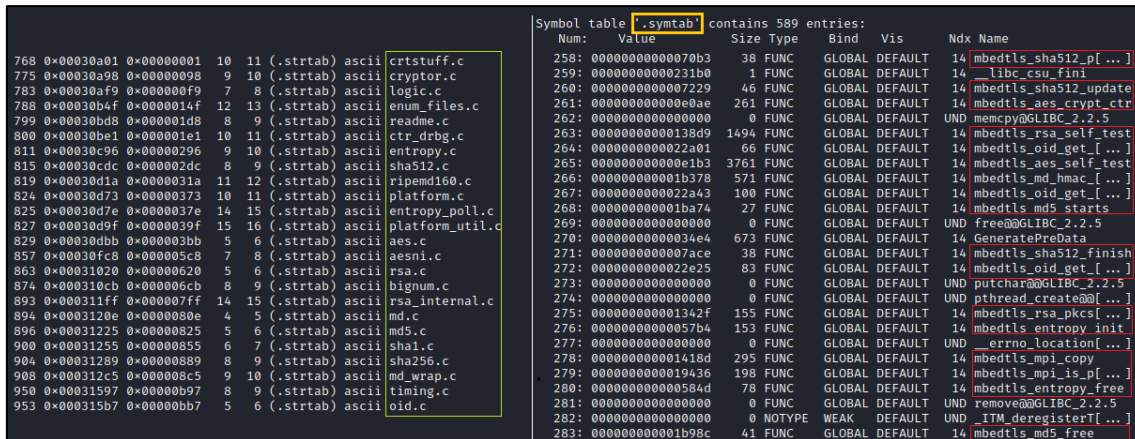


Figura 2. .symtab (Símbolos de mbed TLS)

La entropía reflejada por los *loadable segments* muestra un binario que no ha sido empaquetado/cifrado.

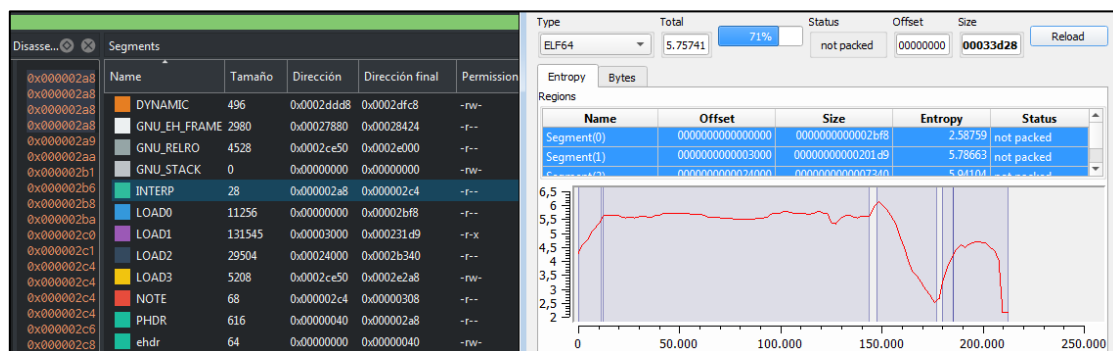


Figura 3. Segmentos / Entropía (LOAD)

El código dañado debe ejecutarse con uno o varios argumentos indicando los directorios que desean cifrarse. Los ficheros resultantes serán renombrados siguiendo el formato *“.31gs1-<8_hex_string>”* donde *<8_hex_string>* indica un valor aleatorio de 8 caracteres en hexadecimal.

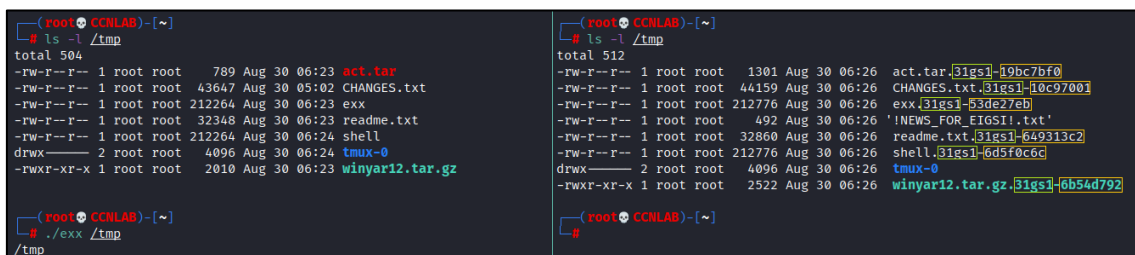
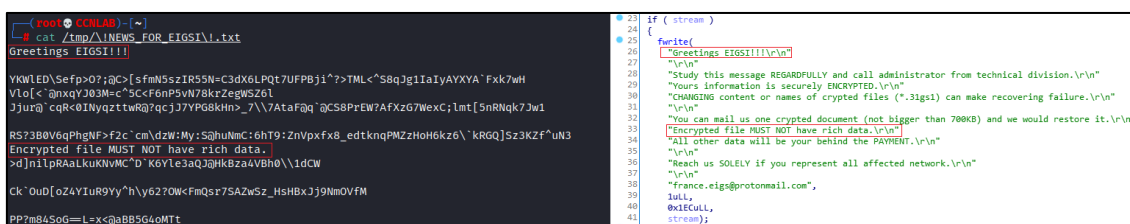


Figura 4. Ficheros cifrados

En cada directorio afectado se creará un fichero con el nombre “!NEWS_FOR_EIGSI!.txt” en donde se indican las instrucciones para recuperar los ficheros cifrados. Cabe destacar que el *sample* actual únicamente muestra de manera legible algunas de las cadenas de dicho *txt*. El cuerpo del mensaje, donde se detallan las instrucciones de recuperación, está en su mayoría codificado posiblemente por algún error en el *builder* encargado de gestionar dicha lógica. En la siguiente imagen se muestra, a la izquierda, el mensaje de texto generado por la muestra actual y, en la derecha, el generado por otro *sample*¹³, de características muy similares (misma *public key*).



```

root@CCNLAB:~# cat /tmp/!NEWS_FOR_EIGSI!.txt
Greetings EIGSI!!!

YKwLEd\SeFp>07;@C<[sfmH5szIR55N=C3dX6LPqt7UFPBji^?>TML<^S8qJg1IaIyAYXYA`Fxx7wH
Vlo[<^@nxqY3B3M=c^5c<F6nPSvN78krzegWS26l
Jjurq`cqR<0IlyqzttwR@q;cJ7YPQ8kHn>_7\7AtaF@q`@CS8PrEW7AfXzG7WexC;1mt[5nRNqk7Jw1

RS?3B0V6qPhgNF>F2c`cm\dzw:My:SqhuNmc:6hT9:ZnVpxfx8_edtkngPMZzHoH6kz6`kRQq]Sz3Kzf^uN3
Encrypted file MUST NOT have rich data.
>d]ni1pRAaLkuKNVMC`D`K6Y1e3aQ3qHKBza4VBh0\1dCW

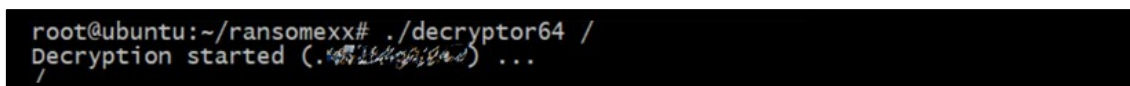
Ck`Oud[oZ4YIur9Yy`h\y6220w<FmQsr7SAZwSz_HsHBxJ9NmOVfM
PP?m8ASoG=L=x<@aBB5G4oMTT

23 if ( stream )
24 {
25     fwrite(
26         "Greetings EIGSI!!!\r\n"
27         "\r\n"
28         "Study this message REGARDFULLY and call administrator from technical division.\r\n"
29         "Yours information is securely ENCRYPTED.\r\n"
30         "CHANGING content or names of crypted files (*.3igs1) can make recovering failure.\r\n"
31         "\r\n"
32         "You can mail us one crypted document (not bigger than 700KB) and we would restore it.\r\n"
33         "Encrypted file MUST NOT have rich data.\r\n"
34         "All other data will be your behind the PAYMENT.\r\n"
35         "\r\n"
36         "Reach us SOLELY if you represent all affected network.\r\n"
37         "\r\n"
38         "france.eigs@protonmail.com",
39         1uLL,
40         0x1fCuLL,
41         stream);

```

Figura 5. Fichero !NEWS_FOR_EIGSI!.txt

Si las víctimas pagan el rescate¹⁴, recibirán una herramienta (*decryptor64*) con la clave privada correspondiente con la que podrán recuperar los ficheros cifrados.



```

root@ubuntu:~/ransomexx# ./decryptor64 /
Decryption started (.3igs1e3aQ3qHKBza4VBh0\1dCW) ...

```

Figura 6. Decryptor64. Fuente: BleedPingComputer

El grupo cibercriminal mantiene, en uno de los dominios *onion* ([http://rns777cdsjrsdlbs4v5qoeppu3px6sb2igmh53jzrx7ipcrbjz5b2ad\[.\]onion](http://rns777cdsjrsdlbs4v5qoeppu3px6sb2igmh53jzrx7ipcrbjz5b2ad[.]onion)), un portal Web donde publican de forma regular los *leaks* vinculados con algunas de las víctimas que han rechazado el pago del rescate. A fecha de realización del presente análisis, los últimos *leaks* publicados estarían relacionados con *Gigabyte Technology* y *American Megatrends International*¹⁵.

4. CARACTERÍSTICAS TÉCNICAS

El binario dañino acepta como parámetros de entrada cada uno de los directorios que desean cifrarse (de forma recursiva). El código no implementa técnicas antianálisis, no trata de finalizar procesos/servicios y no establece comunicaciones con ningún servidor de control. Asimismo, no sobrescribe el *free space* tras finalizar el cifrado tal como ejecuta la variante para plataformas Windows.

Figura 7. Ejecución RansomEXX

Figura 8. Función *GeneratePreData()*

Figura 9. *EnumFiles / init workers / encrypt worker*

Desde *EnumFiles* también se invocará la función *list_dir* encargada de crear el fichero *!NEWS_FOR_EIGSI!.txt* en cada directorio afectado por medio de la función *ReadMeStoreForDir*.

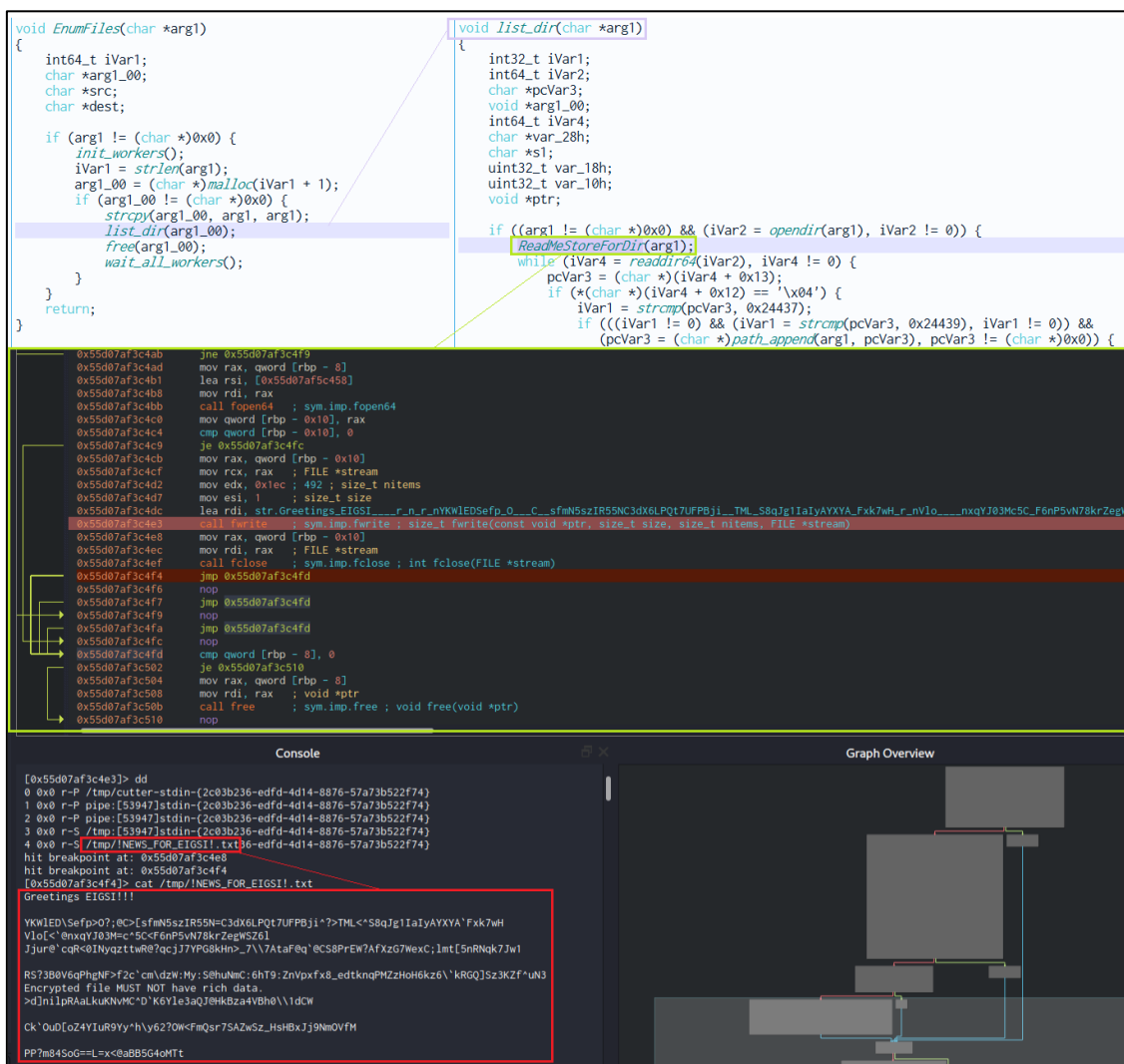
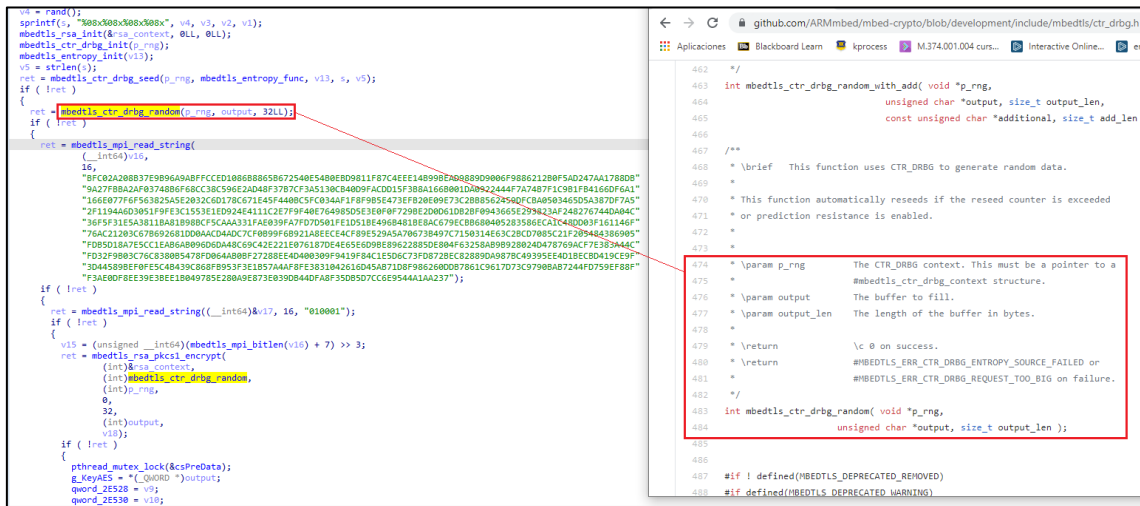


Figura 10. Función *ReadMeStoreForDir*

5. CIFRADO

El binario ha sido compilado estáticamente con la librería *mbed TLS*, una librería *open-source* [\[Error! Marcador no definido.\]](#) que implementa funcionalidades criptográficas (manipulación de certificados X.509, protocolos SSL/TLS y DTLS, etc.). A diferencia de la versión para Windows, el binario no ha sido *stripped* facilitando de este modo la ingeniería inversa gracias a los símbolos disponibles en su *symtab*.

Desde la función *GeneratePreData* se hace uso de *mbed TLS* para generar claves AES por medio de la función *mbedtls_crt_drbg_random*, encargada de generar 32 bytes aleatorios.



```

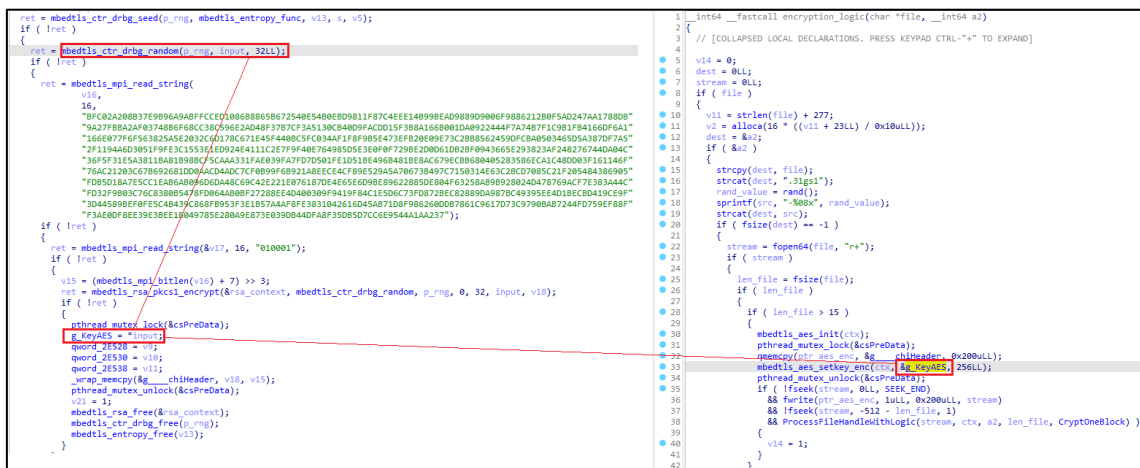
141  ret = mbedtls_ctr_drbg_random(p_rng, output, 32);
142  if ( !ret )
143  {
144      return mbedtls_mpi_read_string(
145          (int64_t)v10,
146          16,
147          "f0c02a08b37e9b96a9ffcced108688658672548e5480e09811f87c4ee148998e9d9890906f988621280f5a0247a17880b"
148          "9a2778b2af0374806f6c8c3b962d4d8f73787c3a5130c34809facc015f38b166801da892244af7a7487f1c91f84160f6a1"
149          "166877f6f53b25a5e2032c0178c7e1e45f440b3f0344f1f8f95e473ef280e9973c28b8562540fcb6a569465d5a3870f7a5"
150          "2f119446d3051f9f3c15531e0924e411c2e7f9f48e764985d5e3e0f8f7298e2d0061d028f0943665e298a23af24827644d04c"
151          "36f3f315a3b11b81898b7c5caaa31fae039f7d70501fe1d518e49684818e8ac679cc8b80405283586ca1c4d0d3f161148f"
152          "76ac2128c678626d1004c4d40c7f809f6892148eecc4cf89529a5470b738497c715013453c29c278b5c21f29a46386085"
153          "f0d5018a75c1eab6a8960d0a48c69c4221e0761870e4e65e6098e8962885d084f63258a8989280240478769ac7e38344c"
154          "f0d3298037c6c8885478f0d6a480f27288e4d400309f9419f84c1e506c73f08728c282890a0878c49395e4d18ec0419c9f"
155          "3045898f0f5c48439c86f89537e18574a4f8f38310a216045a871d0f986260007861c9617d73c37980a87244fd759ef88f"
156          "f3a60f8e39e38e18049785e280a9e873e308040f48f3508507cc69544a1a237");
157  }
158  if ( !ret )
159  {
160      ret = mbedtls_mpi_read_string((int64_t)v17, 16, "010001");
161      if ( !ret )
162      {
163          v15 = (unsigned_int64)(mbedtls_mpi_bitlen(v10) + 7) >> 3;
164          ret = mbedtls_rsa_pkcs1_encrypt(
165              (int)&rsa_context,
166              (int)mbedtls_ctr_drbg_random,
167              (int)p_rng,
168              0,
169              32,
170              (int)output,
171              v18);
172      }
173      if ( !ret )
174      {
175          pthread_mutex_lock(&csPreData);
176          g_KeyAES = "(0000 *)output;
177          qword_2E528 = v19;
178          qword_2E530 = v10;
179      }
180  }
  
```

```

462  /*
463  int mbedtls_ctr_drbg_random_with_add( void *p_rng,
464      unsigned char *output, size_t output_len,
465      const unsigned char *additional, size_t add_len )
466  {
467  /**
468   * \brief This function uses CTR_DRBG to generate random data.
469   *
470   * This function automatically reseeds if the reseed context is exceeded
471   * or prediction resistance is enabled.
472   *
473   *
474   * \param p_rng The CTR_DRBG context. This must be a pointer to a
475   *               #mbedtls_ctr_drbg_context structure.
476   * \param output The buffer to fill.
477   * \param output_len The length of the buffer in bytes.
478   *
479   * \return \c 0 on success.
480   * \return #MBEDTLS_ERR_CTR_DRBG_ENTROPY_SOURCE_FAILED or
481   *         #MBEDTLS_ERR_CTR_DRBG_REQUEST_TOO_BIG on failure.
482   */
483  int mbedtls_ctr_drbg_random( void *p_rng,
484      unsigned char *output, size_t output_len );
485  }
486  #if !defined(MBEDTLS_DEPRECATED_REMOVED)
487  #if defined(MBEDTLS_DEPRECATED_WARNING)
  
```

Figura 11. Generación claves AES

Las API `mbedtls_aes_init` y `mbedtls_aes_setkey_enc` se encargarán de inicializar el contexto AES y establecer, el grupo de 32 bytes previamente creado, como Key para cifrar, en modo ECB (Electronic CodeBook), el contenido de los ficheros.



```

141  ret = mbedtls_ctr_drbg_random(p_rng, output, 32);
142  if ( !ret )
143  {
144      return mbedtls_mpi_read_string(
145          (int64_t)v10,
146          16,
147          "f0c02a08b37e9b96a9ffcced108688658672548e5480e09811f87c4ee148998e9d9890906f988621280f5a0247a17880b"
148          "9a2778b2af0374806f6c8c3b962d4d8f73787c3a5130c34809facc015f38b166801da892244af7a7487f1c91f84160f6a1"
149          "166877f6f53b25a5e2032c0178c7e1e45f440b3f0344f1f8f95e473ef280e9973c28b8562540fcb6a569465d5a3870f7a5"
150          "2f119446d3051f9f3c15531e0924e411c2e7f9f48e764985d5e3e0f8f7298e2d0061d028f0943665e298a23af24827644d04c"
151          "36f3f315a3b11b81898b7c5caaa31fae039f7d70501fe1d518e49684818e8ac679cc8b80405283586ca1c4d0d3f161148f"
152          "76ac2128c678626d1004c4d40c7f809f6892148eecc4cf89529a5470b738497c715013453c29c278b5c21f29a46386085"
153          "f0d5018a75c1eab6a8960d0a48c69c4221e0761870e4e65e6098e8962885d084f63258a8989280240478769ac7e38344c"
154          "f0d3298037c6c8885478f0d6a480f27288e4d400309f9419f84c1e506c73f08728c282890a0878c49395e4d18ec0419c9f"
155          "3045898f0f5c48439c86f89537e18574a4f8f38310a216045a871d0f986260007861c9617d73c37980a87244fd759ef88f"
156          "f3a60f8e39e38e18049785e280a9e873e308040f48f3508507cc69544a1a237");
157  }
158  if ( !ret )
159  {
160      ret = mbedtls_mpi_read_string((int64_t)v17, 16, "010001");
161      if ( !ret )
162      {
163          v15 = (mbedtls_mpi_bitlen(v10) + 7) >> 3;
164          ret = mbedtls_rsa_pkcs1_encrypt(&rsa_context, mbedtls_ctr_drbg_random, p_rng, 0, 32, input, v18);
165      }
166      if ( !ret )
167      {
168          pthread_mutex_lock(&csPreData);
169          g_KeyAES = "input;
170          qword_2E528 = v10;
171          qword_2E530 = v11;
172          _wrap_memcpy(&_ch1header, v18, v15);
173          pthread_mutex_unlock(&csPreData);
174          v11 = 1;
175          mbedtls_rsa_free(&rsa_context);
176          mbedtls_ctr_drbg_free(p_rng);
177          mbedtls_entropy_free(v13);
178      }
179  }
  
```

```

1  int64 __fastcall encryption_logic(char *file, __int64 a2)
2  {
3  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]
4
5  v14 = 0;
6  dest = 0LL;
7  stream = 0LL;
8  if ( !file )
9  {
10     v11 = strlen(file) + 277;
11     v2 = allocate(16 * ((v11 + 23LL) / 0x100LL));
12     dest = &v2;
13     if ( !&v2 )
14     {
15         strcpy(dest, file);
16         strcat(dest, ".31gsl");
17         rand_value = rand();
18         sprintf(src, "%080x", rand_value);
19         strcat(dest, src);
20         if ( !fsize(dest) )
21         {
22             stream = fopen64(file, "r+");
23             if ( !stream )
24             {
25                 len_file = fsize(file);
26                 if ( len_file )
27                 {
28                     if ( len_file > 15 )
29                     {
30                         mbedtls_aes_init(ctb);
31                         pthread_mutex_lock(&csPreData);
32                         g_keyenc = &_ch1header, 0x200uLL);
33                         mbedtls_aes_setkey_enc(ctb, g_KeyAES, 256LL);
34                         pthread_mutex_unlock(&csPreData);
35                         if ( !fseek(stream, 0LL, SEEK_END)
36                             && fwrite(ptr_aes_enc, 1uLL, 0x200uLL, stream)
37                             && ifseek(stream, -512, len_file, 1)
38                             && ProcessFileHandleWithLogic(stream, ctb, a2, len_file, CryptOneLock) )
39                         {
40                             v14 = 1;
41                         }
42                     }
43                 }
44             }
45         }
46     }
47 }
  
```

Figura 12. Generación claves AES

La clave AES será cifrada utilizando una clave RSA de 4096 bits (embebida en el código) y el *ciphertext* resultante (clave AES cifrada con RSA) será añadido al final de cada fichero cifrado. La lógica empleada para fijar la clave AES y para copiar dicho *ciphertext* es coordinada por medio de objetos Mutex (`pthread_mutex_lock` y

pthread_mutex_unlock) durante el proceso de cifrado para garantizar la exclusión mutua (véase el recuadro verde en la siguiente imagen).

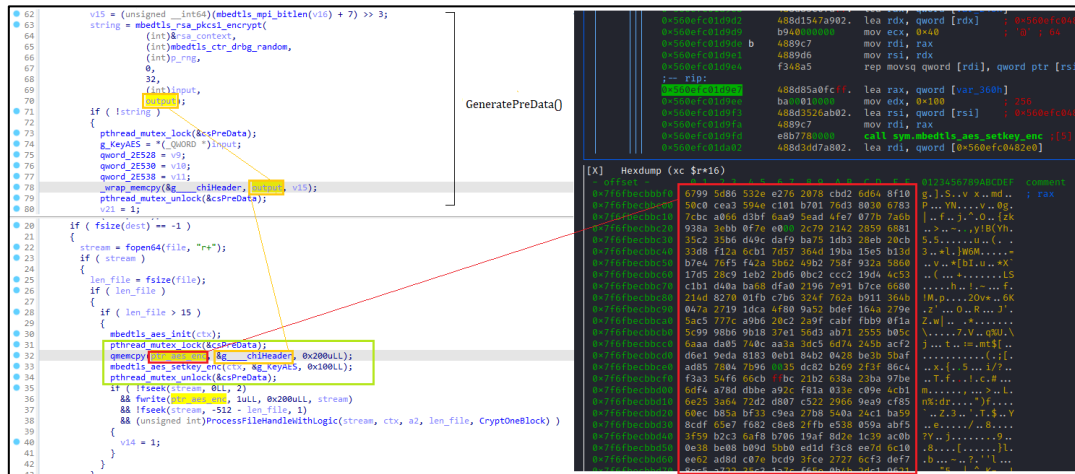


Figura 13. Generación claves AES / Ciphertext (clave AES cifrada con RSA)

Nota: Es posible que los binarios para ambas plataformas (Windows/Linux) compartan el mismo código fuente en aquellas funciones que interactúan con *mbedtls TLS* dada la similitud existente en la lógica empleada para la gestión y creación de claves.

A cada fichero cifrado se le añadirá la extensión *“.31gs1”* seguido de un valor en hexadecimal de 8 caracteres derivado de la API *rand()* y formateado con *sprintf* ("%08x").

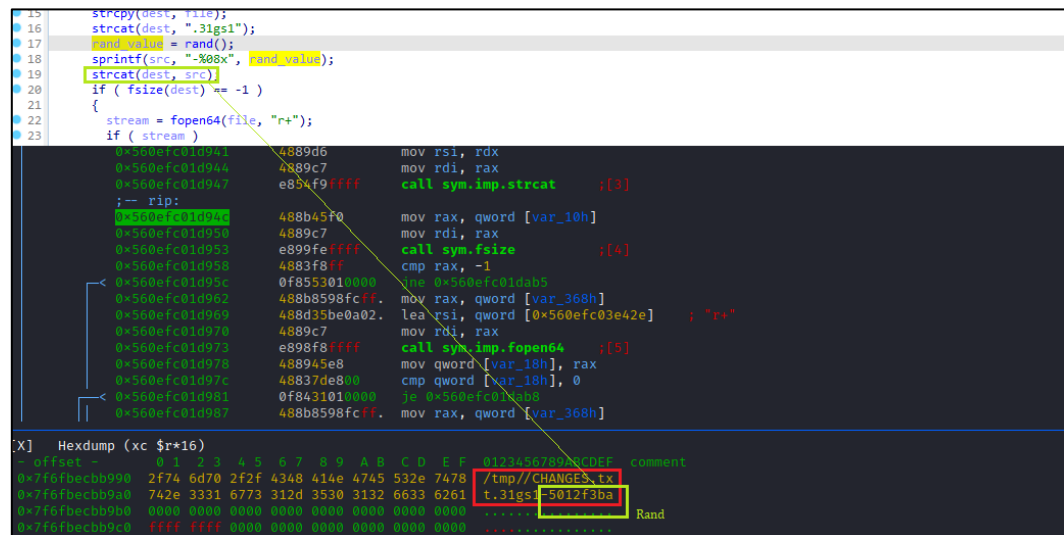


Figura 14. Renombrado de ficheros

ProcessFileHandleWithLogic será la función encargada de cifrar el contenido de cada fichero. Esta función recibe como argumento un puntero a la función *CryptOneBlock* responsable de utilizar la API de *mbedtls_aes_crypt_ecb* (perteneciente a *mbedtls*) para cifrar los bloques de datos de cada fichero. A modo de ejemplo, en la siguiente imagen se muestra el antes y después del cifrado de determinado fichero ordinario.

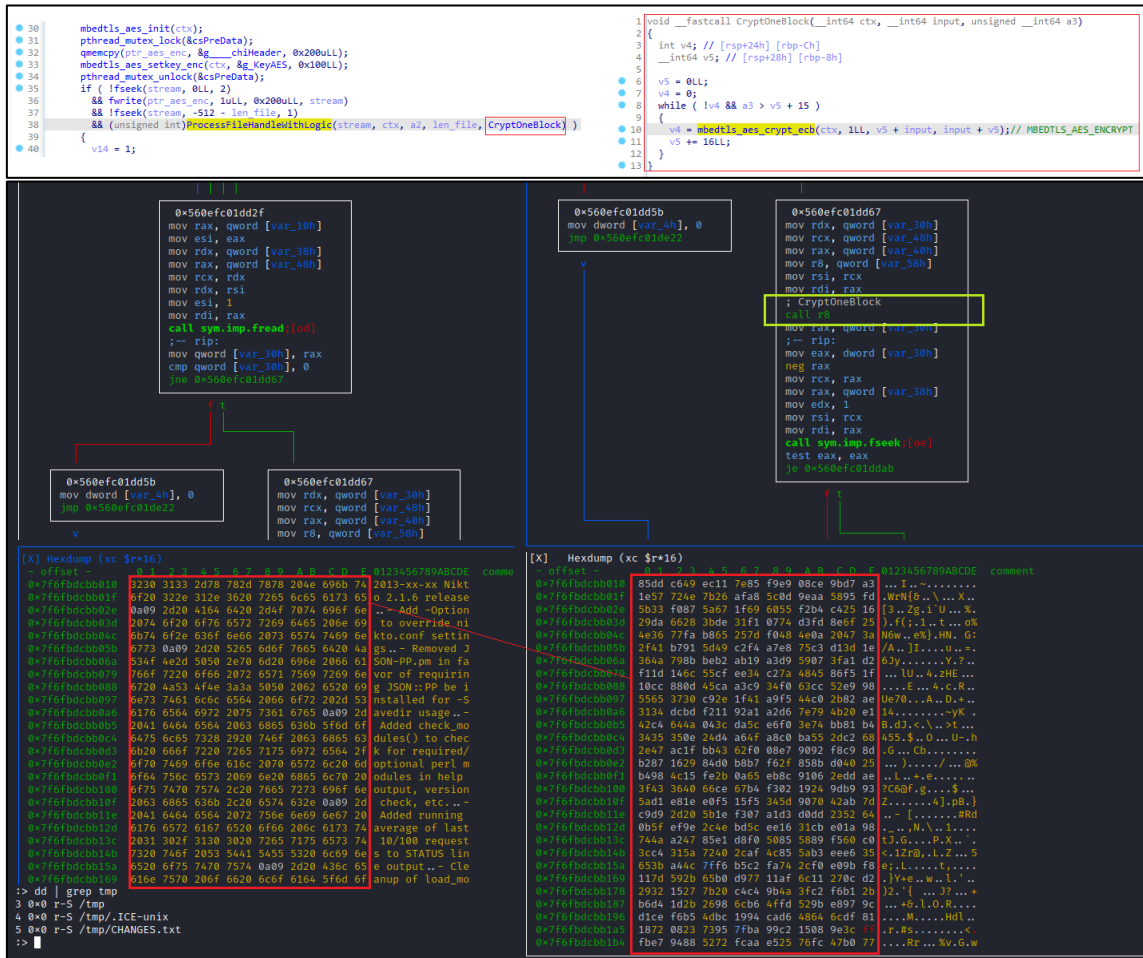


Figura 15. Cifrado de archivos

A cada fichero se añadirá la clave AES cifrada con RSA al final del mismo garantizando de este modo que el único método factible para descifrar el mismo sea mediante la clave privada RSA (en manos del grupo cibercriminal).



Figura 16. Escritura clave AES (cifrada con RSA)

El siguiente *call graph* muestra el *path* de ejecución previamente descrito y recoge el conjunto de funciones más destacables que intervienen en la operativa del cifrado de ficheros.

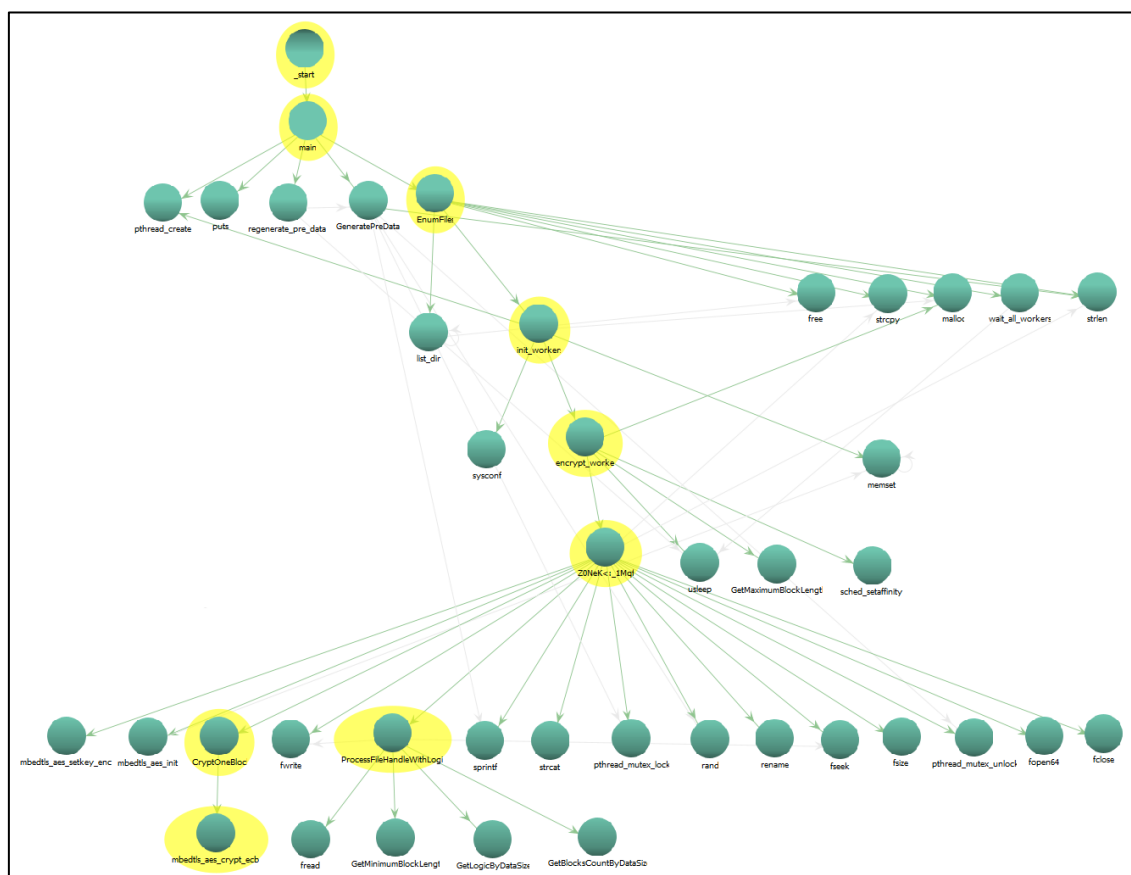


Figura 17. Call Graph

6. DESINFECCIÓN

Dado que la variante analizada no adquiere persistencia, los equipos afectados con RansomEXX no requieren de una rutina de desinfección más allá de reiniciar el sistema o finalizar el proceso vinculado con el *ransomware*. Cabe destacar que, aunque el propio binario dañino no implementa funcionalidades de persistencia, el conjunto de TTP empleadas por los atacantes para proceder con el despliegue del *ransomware* sí que puede involucrar otros componentes o técnicas para garantizar la persistencia del actor; por ejemplo, otros binarios dañinos, cuentas de usuarios a modo de *backdoor*, vulnerabilidades¹⁶ en un servicio expuesto, etc. El proceso de respuesta a incidentes tendrá que investigar y analizar de forma meticulosa el vector de entrada, así como el conjunto de TTP empleado por los atacantes para garantizar que la mitigación ha sido efectiva y para evitar la repetición del ataque.

Respecto al descifrado de los ficheros, el modelo de criptografía utilizado asegura que el único método factible para el descifrado sea mediante el uso de la clave privada RSA.

7. MITIGACIÓN

La manera más estratégica para contener y reducir el impacto del *ransomware* es trabajar de forma paralela en el aislamiento de redes/VLAN que conforman la entidad

afectada (con el objetivo de contener los segmentos de red con equipos infectados y evitar su expansión) y en el rotado de contraseñas en el dominio. Dicho rotado debe incluir las cuentas locales de administrador (es importante que sean únicas para cada equipo) así como la cuenta KRBTGT del dominio. Téngase en cuenta que estas medidas pueden interferir en el correcto funcionamiento del dominio o redes afectadas y pueden causar diversos imprevistos (por ejemplo, el reinicio de máquinas); no obstante, permitiría contener la amenaza de una manera resolutiva.

8. REGLAS DE DETECCIÓN

8.1. REGLAS DE YARA

```
rule RansomEXX_Lin {
  meta:
    description = "Ransomware ransomEXX (Linux)"
    date = "2021-09-14"
    author = "CCN-CERT"

  strings:
    $x1 = "ProcessFileHandleWithLogic" fullword ascii
    $x2 = "mbedtls_aes_crypt_ecb" fullword ascii
    $x3 = "ReadMeStoreForDir" fullword ascii
    $x4 = "!NEWS_FOR" fullword ascii
    $x5 = "enum_files.c" fullword ascii
    $x6 = {45 6E 63 72 79 70 74 65 64 20 66 69 6C 65 20 4D 55 53 54 20 4E 4F 54 20 68 61 76 65 20 72 69 63
    68 20 64 61}
    $x7 = "Encrypted file MUST NOT have rich data" fullword ascii
    $x8 = "ReadMeStoreForDir" fullword ascii
    $x9 = { C7 00 2E 33 31 67 66 C7 40 04 73 31 C6 40 06 00 } // .31gs1 extension
    $x10 = { 42 46 43 30 32 41 32 30 38 42 33 37 45 39 42 } // Embeded RSA public key}
    $x11 = "regenerate_pre_data" fullword ascii
    $x12 = "ProcessFileHandleWithLogic" fullword ascii

  condition:
    (uint32(0) == 0x464C457F) and ((filesize > 100KB) and (filesize < 350KB)) and (3 of ($x*))
}
```

9. REFERENCIAS

¹ **Malpedia: GOLD DUPONT**

https://malpedia.caad.fkie.fraunhofer.de/actor/gold_dupont

² **Last, but Not Least: Defray777**

<https://unit42.paloaltonetworks.com/vatet-pyxie-defray777/3/>

³ **Texas Courts hit by ransomware, network disabled to limit spread**

<https://www.bleepingcomputer.com/news/security/texas-courts-hit-by-ransomware-network-disabled-to-limit-spread/>

⁴ **Ransomware attack impacts Texas Department of Transportation**

<https://www.bleepingcomputer.com/news/security/ransomware-attack-impacts-texas-department-of-transportation/>

⁵ **Business technology giant Konica Minolta hit by new ransomware**

<https://www.bleepingcomputer.com/news/security/business-technology-giant-konica-minolta-hit-by-new-ransomware/>

⁶ **Brazil's court system under massive RansomExx ransomware attack**

<https://www.bleepingcomputer.com/news/security/brazils-court-system-under-massive-ransomexx-ransomware-attack/>

⁷ **Hackers leak data from Embraer, world's third-largest airplane maker**

<https://www.zdnet.com/article/hackers-leak-data-from-embraer-worlds-third-largest-airplane-maker/>

⁸ **Double extortion ransomware**

<https://www.darktrace.com/en/blog/double-extortion-ransomware/>

⁹ **Ransomware gangs are abusing VMWare ESXi exploits to encrypt virtual disks**

<https://www.zdnet.com/article/ransomware-gangs-are-abusing-vmware-esxi-exploits-to-encrypt-virtual-hard-disks/>

¹⁰ **Mbed TLS**

<https://github.com/ARMmbed/mbedtls>

¹¹ **CCN-CERT: Informe Código Daño: RansomEXX (Windows)**

<https://www.ccn-cert.cni.es/en/reports/public/6158-ccn-cert-id-11-21-ransomexx-win-ransomware-1/file.html>

¹² **RansomEXX Trojan attacks Linux systems**

<https://securelist.com/ransomexx-trojan-attacks-linux-systems/99279/>

¹³ **MalwareBazaar Database: cb408d45762a628872fa782109e8fcfc3a5bf456074b007de21e9331bb3c5849**

<https://bazaar.abuse.ch/sample/cb408d45762a628872fa782109e8fcfc3a5bf456074b007de21e9331bb3c5849/>

¹⁴ **RansomExx ransomware also encrypts Linux systems:**

<https://www.bleepingcomputer.com/news/security/ransomexx-ransomware-also-encrypts-linux-systems/>

¹⁵ **Computer hardware giant GIGABYTE hit by RansomEXX ransomware:**

<https://www.bleepingcomputer.com/news/security/computer-hardware-giant-gigabyte-hit-by-ransomexx-ransomware/>