

Escenario de Entrenamiento - Modificación de Memoria

Nombre del Grupo

Datos de Entrenamiento - Process Hollowing SQL (Hive)

Escenario

En este escenario de entrenamiento, ejecutaremos una carga útil de ransomware mediante process hollowing, una técnica de inyección/evasión en la que se inicia un proceso y su código ejecutable es modificado para realizar algún otro comportamiento.

Las técnicas en memoria se encuentran entre las más difíciles de detectar, e incluso más difíciles de responder de forma automatizada, por lo que a menudo se ignoran. La memoria es, por definición, volátil y está en constante cambio. Identificar cambios dentro de la memoria de un proceso requiere identificar las secciones relevantes y evaluarlas en distintos momentos del ciclo de vida de un proceso para ver si ha sido manipulada.

Identificándolo

Number of Incidents	Machine Name	Program Path	Program Arguments
▼ 1	\\CC-NOAH-TEST	C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\...	SQLCMD.EXE

Created	Response Name	Untrusted Remote Threads	Modified Memory	No Trusted Cert
October 18, 2022 at 6:46:59 PM EDT	    Moonflower Razzmatazz Zonkey 74	False	True	False

Las respuestas de memoria modificada son, por mucho, las más difíciles de inspeccionar. Dado que la memoria es efímera, se trata como una “caja negra” y a menudo se mantiene a distancia. La buena noticia es que, a partir de Cyber Crucible 4.4.1.3, ahora tenemos la capacidad de recopilar automáticamente telemetría de muestras de memoria modificada, identificando qué rangos de memoria fueron modificados y los cambios exactos realizados en ellos.

Sin embargo, ese tipo de análisis es muy engorroso de realizar, por lo que un buen primer paso es revisar los procesos relacionados en el panel de control para hacerse una idea del comportamiento relacionado. Dado que el proceso en cuestión aquí es un SQLCMD.exe firmado, debería resultar

rápidamente obvio si se trata de un proceso SQL benigno o de algo peor.

Parent Path	Parent Arguments	Child Path
C:\Windows\System32\cmd.exe	"C:\Windows\system32\cmd.exe"	C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\
C:\Windows\explorer.exe	C:\Windows\Explorer.EXE	C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\
C:\Program Files\Microsoft SQL Server\Client S...	"C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\...	C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\
C:\Program Files\Microsoft SQL Server\Client S...	SQLCMD.EXE	C:\Windows\System32\net.exe

Hasta ahora todo bien, solo procesos SQL normales, nada necesariamente sospechoso ni tampoco no sospechoso. Aunque, nunca nos gusta ver CMDs involucrados en respuestas automatizadas. Vamos a investigar un poco más.

Parent Path	Child Path	Child Arguments
C:\Windows\System32\net.exe	C:\Windows\System32\net1.exe	C:\Windows\system32\net1 stop "vmicvss" /y
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\net.exe	net.exe stop "VSS" /y
C:\Windows\System32\net.exe	C:\Windows\System32\net1.exe	C:\Windows\system32\net1 stop "VSS" /y
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\net.exe	net.exe stop "WRSVC" /y
C:\Windows\System32\net.exe	C:\Windows\System32\net1.exe	C:\Windows\system32\net1 stop "WRSVC" /y
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\net.exe	net.exe stop "UnistoreSvc_5205e" /y
C:\Windows\System32\net.exe	C:\Windows\System32\net1.exe	C:\Windows\system32\net1 stop "UnistoreSvc_5205e" /y
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\sc.exe	sc.exe config "LanmanWorkstation" start= disabled
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\sc.exe	sc.exe config "MsDtsServer130" start= disabled
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\sc.exe	sc.exe config "MSSQLMSSQLSERVER01" start= disabled
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\sc.exe	sc.exe config "MSSQLSERVER" start= disabled
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\sc.exe	sc.exe config "sacsvr" start= disabled
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\sc.exe	sc.exe config "SamSs" start= disabled
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\sc.exe	sc.exe config "SQLAgent\$MSSQLSERVER01" start= disabled
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\130\Tools\Binn\SQLCMD.exe	C:\Windows\System32\sc.exe	sc.exe config "SQLBrowser" start= disabled

¡La prueba irrefutable!

¡Y ahí está! Podemos seguir desplazándonos y ver cada vez más comportamientos como estos. El ejecutable SQLCMD está ejecutando todo tipo de comandos para deshabilitar servicios, cambiar configuraciones, y todo lo que un ransomware haría de manera preventiva.

```

33 2E 39 35 00 55 50 58 21 0D 24 0E 0A 2F 0A 72 3.95.UPX!.$../.r
3C A0 BF E2 D5 BC 78 37 00 89 EF 0D 00 00 22 32 <?????x7...?"2
00 49 39 00 34 1A 03 00 32 92 0D 60 C1 31 9B FD .I9.4...2..`?1.?
DA 0A 6D 23 7E 10 FA 7A 45 17 EE D3 C1 7B 5A 29 ?.m#~.?.zE.???{Z)
46 0D FE AB FF 75 95 FD 0B 82 78 55 EA B0 16 36 F.???u.?.xU???.6
8E DE D9 92 0E 16 2E 17 D8 AE 46 29 60 2B 87 B8 .??.....??F)`+.?
B1 44 40 E3 9C 00 B2 40 66 01 1B AF 99 05 4E EE ?D@?...?@f...N?
10 B1 7C C8 8F A3 1D A3 1D 65 70 A4 95 43 77 6A .?|?..?.ep?.Cwj
E0 EC 83 F0 30 4B BB EE 4C 82 D1 AA CC BF 75 A7 ???.?0K??L.???u?
D5 92 0A 68 4F 32 E0 2E 40 19 B7 9D 23 4D 13 6C ?..h02?..@..#M.1
B9 3D D2 FF 82 25 BF A0 91 F0 99 22 F4 9B 3C F4 ?=???.%???.?"?.<?
B1 80 EC 52 08 B5 95 4B F5 90 CD 1F BF E9 24 C2 ?.?R.?.K?..??$?
33 2E 7F 4F E1 8E E9 00 DB 2A 43 B9 FF 97 63 74 3..0?.?.*C???.ct
80 01 B3 F6 39 BC C5 13 59 D9 06 B2 B7 33 C2 D0 ...??9???.Y?..??3??
56 C3 75 A0 0A BB 28 0D AA BA 7F BC E4 15 90 88 V?u?..?(.??..??...
AD CD FC FF BC 7E 45 4E D7 32 60 90 A5 6F 8D D8 ?????~EN?2`.?o.?
6E 44 E0 39 02 45 2B 31 0D EE 25 D6 53 5E A4 17 nD?9.E+1.??%?S^?.
B9 0F 05 22 82 68 8A 26 7F 5B 8D EB 95 11 AF 05 ?..".h.&.[.??..?.
F6 D4 3D 7B 38 D0 1E 47 03 2A BE C7 28 84 19 B3 ??={8?.G.*??(..?
9A 64 3E AD 5F 7A 36 8B D8 D7 7C 43 B6 5F DE DB .d>?_z6.??|C?_??
4F F4 16 C0 9E DE 92 F8 97 0D 99 50 AF E0 AD CD O?..?..?....P?????
C2 97 5B C8 BF D6 3E 0F C7 BC D0 89 9D 21 13 AF ?..[???>..???..!..?
E6 49 31 51 3D 8A 22 36 6D DD A1 A6 E8 F7 6A 75 ?I1Q="6m?????ju
D5 22 77 21 D0 67 64 B3 2E 96 A6 81 6E 4F 49 E6 ?"w!gd?...?.nOI?
59 81 5F D9 44 E3 9F 2D C6 45 97 5F 0B CA 95 39 Y._?D?..-?E._.?.9
89 78 3A BC 23 CE A4 64 34 01 29 31 05 29 4B 36 .x:??d4.)1.)K6
76 A3 4D 4A 73 33 B2 23 17 D7 0C E4 D6 4B 79 62 v?MJs3?#.?.??Kyb

```

```

332e      xor      ebp, dword ptr [rsi]
393500555058  cmp      dword ptr [58D95708h], esi
210d240e0a2f  and      dword ptr [2F931032h], ecx
0a723c      or       dh, byte ptr [rdx+3Ch]
a0bfe2d5bc78370089 mov      al, byte ptr [89003778BCD5E2BFh]
ef         out      dx, eax
0d00002232  or       eax, 32220000h
004939      add      byte ptr [rcx+39h], cl
00341a      add      byte ptr [rdx+rbx], dh
0300      add      eax, dword ptr [rax]
32920d60c131  xor      dl, byte ptr [rdx+31C1600Dh]
9b         wait
fd         std
da0a      fimul    dword ptr [rdx]
6d         ins     dword ptr [rdi], dx
237e10     and      edi, dword ptr [rsi+10h]
fa         cli
7a45      jp       000000000089027E
17        ???
ee         out      dx, al
d3c1      rol     ecx, cl
7b5a      jnp     0000000000890299
29460d     sub     dword ptr [rsi+0Dh], eax
fe        ???
ab        stos   dword ptr [rdi]
ff7595     push    qword ptr [rbp-6Bh]
fd        std

```

Aquí podemos ver una pequeña idea de lo que nos espera cuando empezamos a profundizar y analizar las diferencias de memoria. La mayoría de las técnicas lo suficientemente sofisticadas

como para ejecutar malware completamente en memoria también van a ofuscar su código en memoria. Esto hace que su análisis sea extremadamente difícil, pero el acceso a la telemetría adicional ya ha demostrado ser invaluable para la identificación de incidentes, así como para el ajuste de comportamientos

Documentación relevante

- **Inyección de Procesos**

- [Mitre T1559](#) - La comunicación entre procesos puede proporcionar control sobre el proceso objetivo desde el inyector una vez que se completa la inyección.
- [Mitre T1106](#) - Las API nativas suelen ser el acceso más directo a la funcionalidad del sistema operativo para acceder a archivos, ejecutar procesos y más.
- [Mitre T1569](#) - Inyectar en un servicio del sistema como un `svchost` existente puede disfrazar código malicioso para que se reporte como ejecutándose desde un proceso conocido y confiable.
- [Mitre T1055](#) - Inyección de procesos, generalmente utilizada para ejecutar código malicioso en un proceso objetivo mientras se permite que el proceso original continúe.
- [Mitre T1543](#) - Crear o modificar un proceso del sistema puede disfrazar código malicioso como un proceso del sistema normal y confiable ante la detección de malware.

Revision #1

Created 2026-07-24 01:15:01 UTC by Dennis Underwood

Updated 2026-07-24 01:15:01 UTC by Dennis Underwood