Análisis técnico externo de la causa raíz : Channel File 291

INTRODUCCIÓN

Este informe ahonda en la información compartida anteriormente en nuestra revisión preliminar posterior al incidente, profundizando en los hallazgos, mitigaciones, detalles técnicos y análisis de la causa raíz del incidente. Al 29 de julio a las 5pm., hora del Pacífico, comparando semana a semana, el 99% de los sensores de Windows estaban en línea en comparación con antes de la actualización de contenidos. Por lo general, vemos una variación de alrededor del 1% semana tras semana en las conexiones de sensores.

A lo largo de este RCA, empleamos terminología generalizada para describir la plataforma CrowdStrike Falcon a fin de mejorar la legibilidad. La terminología en otra documentación puede ser más específica y técnica.

¿QUÉ OCURRIÓ?

El sensor CrowdStrike Falcon ofrece potentes modelos de IA y aprendizaje automático (machine learning) en el sensor para proteger los sistemas de los clientes mediante la identificación y la corrección de las últimas amenazas avanzadas. Estos modelos se mantienen actualizados y se fortalecen con los aprendizajes de la telemetría de amenazas más reciente del sensor y la inteligencia humana de los ingenieros de detección de amenazas de Falcon Adversary OverWatch, Falcon Complete y CrowdStrike. Este amplio conjunto de telemetría de seguridad comienza como datos filtrados y agregados en cada sensor en un repositorio de gráficos local.

Cada sensor correlaciona el contexto de su repositorio gráfico local con la actividad en tiempo real del sistema en comportamientos e indicadores de ataque (IOA) en un proceso continuo de perfeccionamiento. Este proceso de perfeccionamiento incluye un Motor de Detección de Sensores que combina el Contenido de Sensores incorporado con el contenido de respuesta rápida (Rapid Response Content) suministrado desde la nube. El contenido de respuesta rápida se emplea para recopilar telemetría, identificar indicadores de comportamiento del adversario y aumentar las nuevas detecciones y prevenciones en el sensor sin necesidad de cambiar el código del sensor. El contenido de respuesta rápida es una investigación de comportamiento, separada y distinta de las capacidades de prevención y detección de inteligencia artificial (IA) en el sensor de CrowdStrike.

El contenido de respuesta rápida se entrega a través de archivos de canal (Channel Files) y es interpretado por el intérprete de contenido del sensor, empleando un motor basado en expresiones regulares. Cada *channel file* de contenido de respuesta rápida está asociado a un Template Type (tipo de plantilla) específico integrado en una versión del sensor. El

Template Type proporciona al intérprete de contenido datos de actividad y contexto gráfico que se comparan con el contenido de respuesta rápida.

Con el lanzamiento de la versión 7.11 del sensor en febrero de 2024, CrowdStrike introdujo un nuevo Template Type para permitirte la visibilidad y la detección de nuevas técnicas de ataque que abusan de las Named Pipes y otros mecanismos de comunicación entre procesos (“IPC”) de Windows. Como se describe en la revisión preliminar posterior al incidente (PIR), el nuevo Template Type de IPC se desarrolló y probó de acuerdo con nuestros procesos estándar de desarrollo de contenido de sensor y se integró en el sensor para prepararlo para su uso en el terreno. Las Template Instances (instancias de plantilla) de IPC se entregan como contenido de respuesta rápida a los sensores a través del Channel File correspondiente numerado 291.

El nuevo Template Type de IPC definió 21 campos de parámetros de entrada, pero el código de integración que invocó al intérprete de contenido con las Template Instances del Channel File 291 proporcionó solo 20 valores de entrada con los que coincidir. Esta falta de coincidencia en el recuento de parámetros evadió varias capas de validación y pruebas de compilación, ya que no se descubrió durante el proceso de prueba de la versión del sensor, las pruebas de estrés del Template Type (mediante una Template Instance de prueba) o las primeras implementaciones exitosas de Template Instances de IPC en el campo. En parte, esto se debió al uso de criterios de coincidencia comodín (wildcards) para la entrada 21 durante las pruebas y en las Template Instances de IPC iniciales.

El 19 de julio de 2024 se desplegaron dos Template Instances de IPC adicionales. Uno de ellos introdujo un criterio de coincidencia sin comodines (wildcards) para el parámetro de entrada número 21. Estas nuevas Template Instances dieron como resultado una nueva versión del Channel File 291 que ahora requeriría que el sensor inspeccionara el parámetro de entrada 21. Hasta que este channel file se entregara a los sensores, ninguna Template Instance de IPC en versiones anteriores del canal había utilizado el campo de parámetro de entrada 21. El validador de contenido (Content Validator) evaluó las nuevas Template Instances, pero basó su evaluación en la expectativa de que el Template Type de IPC se proporcionara con 21 entradas.

Los sensores que recibieron la nueva versión del Channel File 291 con el contenido causante del problema se expusieron a un problema de lectura latente fuera de los límites en el intérprete de contenido. En la siguiente notificación de IPC del sistema operativo, se evaluaron las nuevas Template Instances de IPC, especificando una comparación con el valor de entrada 21. El intérprete de contenido solo esperaba 20 valores. Por lo tanto, el intento de acceder al valor 21 produjo una lectura de memoria fuera de los límites más allá del final de la matriz de datos de entrada y provocó un bloqueo del sistema.

En resumen, fue la confluencia de estos problemas lo que resultó en un bloqueo del sistema: la falta de coincidencia entre las 21 entradas validadas por el validador de contenido frente a las 20 proporcionadas al intérprete de contenido, el problema de lectura latente fuera de los límites en el intérprete de contenido y la falta de una prueba específica para los criterios de coincidencia no comodín (sin wildcard) en el campo 21. Si bien este escenario con Channel File 291 ahora no puede repetirse, también informa las mejoras del proceso y los pasos de mitigación que CrowdStrike está desplegando para garantizar una mayor resiliencia.

HALLAZGOS Y MITIGACIONES

1. El número de campos en el Template Type de IPC no se validó en el momento de la compilación del sensor

Conclusiones: En el momento del incidente, el código del sensor para el Template Type de IPC describía 20 fuentes de entrada diferentes para uso de la Template Instance. Esto significa que, cuando el sensor quería tomar una decisión de detección basada en el Template Type de IPC, el código del sensor suministraba 20 fuentes de entrada distintas al intérprete de contenido. Sin embargo, la definición del Template Type de IPC en el archivo Definiciones de Template Type establecía que esperaba 21 campos de entrada. Esta definición dio lugar a Template Instances en el Channel File 291 que esperaban operar con 21 entradas. Este desajuste no se detectó durante el desarrollo del Template Type de IPC. Los casos de prueba y el contenido de respuesta rápida empleados para probar el Template Type de IPC no provocaron ningún fallo durante el desarrollo de la función ni durante las pruebas de la versión 7.11 del sensor.

Mitigación: validar el número de campos de entrada en el Template Type en el momento de compilación del sensor

Un parche para el compilador de contenido del sensor que valida el número de entradas proporcionadas por un Template Type se desarrolló el 19 de julio de 2024 y entró en producción el 27 de julio de 2024, como parte de las herramientas de compilación internas de CrowdStrike. El parche del compilador de contenido del sensor también verificó que ningún otro Template Type proporcionara un número incorrecto de entradas en ninguna plataforma.

2. No existía una verificación de límites de matriz de tiempo de ejecución para los campos de entrada del intérprete de contenido en el Channel File 291

Hallazgos: El contenido de respuesta rápida para el Channel File 291 instruyó al intérprete de contenido que lea la entrada 21 de la matriz de punteros de entrada. Sin embargo, el Template Type de IPC solo genera 20 entradas. Como resultado, una vez que se entregó el contenido de respuesta rápida que empleó un criterio de coincidencia no comodín (sin wildcard) para la entrada número 21, el intérprete de contenido realizó una lectura fuera de los límites de la matriz de entrada. Este no es un problema de escritura arbitraria en memoria y se revisó de forma independiente.

Mitigación: agregar comprobaciones de límites de matriz de entrada en tiempo de ejecución al intérprete de contenido para contenido de respuesta rápida en el Channel File 291

El 25 de julio de 2024 se agregó la comprobación de límites a la función de intérprete de contenido que recupera las cadenas de entrada. Al mismo tiempo, se agregó una comprobación adicional de que el tamaño de la matriz de entrada coincide con el número de entradas esperadas por el contenido de respuesta rápida. Estas correcciones se están adaptando a todas las versiones 7.11 de sensores de Windows y posteriores a través de una versión de revisión de software de sensor. Esta versión estará disponible al público general el 9 de agosto de 2024.

La comprobación de límites agregada evita que el intérprete de contenidos realice un acceso fuera de los límites de la matriz de entrada y bloquee el sistema. La comprobación adicional agrega una capa extra de validación en tiempo de ejecución de que el tamaño de la matriz de entrada coincida con el número de entradas esperado por el contenido de respuesta rápida.

Completamos las pruebas de fuzz del Template Type del Channel 291 y lo estamos ampliando a gestoresde contenido de respuesta rápida adicionales en el sensor.

Corregir el número de entradas proporcionadas por el Template Type de IPC

Se actualizó el código del sensor que define el Template Type de IPC para proporcionar el número correcto de entradas (21). Esta corrección se está aplicando a todas las versiones 7.11 y superiores del sensor de Windows mediante una revisión del software del sensor. Esta versión estará disponible al público general el 9 de agosto de 2024.

3. Las pruebas de Template Type deben cubrir una variedad más amplia de criterios de coincidencia

Resultados: Se realizaron pruebas manuales y automatizadas durante el desarrollo del Template Type de IPC. Estas pruebas se centraron en la validación funcional del Template Type, incluido el flujo correcto de datos relevantes para la seguridad a través de él, y la

evaluación de esos datos para generar alertas de detección adecuadas basadas en criterios creados en casos de prueba de desarrollo.

Las pruebas automatizadas se apoyaron en herramientas internas y externas para crear los datos relevantes para la seguridad necesarios para ejercer el Template Type de IPC en todas las versiones de Windows compatibles dentro de un amplio subconjunto de casos de uso operativo esperados. Para las pruebas automatizadas, se seleccionó un conjunto estático de 12 casos de prueba para que fuera representativo de las expectativas operativas más amplias y para validar la creación de alertas de telemetría y detección. Parte de estas pruebas incluyó definir un channel file para su uso dentro de los casos de prueba. La selección de datos en el channel file se realizó manualmente e incluyó un criterio de coincidencia de comodín regex en el campo 21 para todas las Template Instances, lo que significa que la ejecución de estas pruebas durante el desarrollo y las compilaciones de versiones no expuso la lectura latente fuera de los límites en el intérprete de contenido cuando se le proporcionaron 20 entradas en lugar de 21.

Mitigación: aumentar la cobertura de la prueba durante el desarrollo del Template Type

Para confirmar que estamos validando todos los campos de cada Template Type, se crearon pruebas automatizadas que no coinciden con criterios comodín para cada campo. Este paso se realizó para todos los Template Types existentes y es necesario para todos los Template Types futuros. Además, todos los Template Types futuros incluyen casos de prueba con escenarios adicionales que reflejan mejor el uso en producción.

4. El validador de contenido contenía un error lógico

Resultados: El validador de contenido evaluó las nuevas Template Instances. Sin embargo, basó su evaluación en la expectativa de que el Template Type de IPC contaría con 21 entradas. Esto dio lugar a que la Template Instance causante del problema se enviara al intérprete de contenido.

Mitigación: crear controles adicionales en el validador de contenido

El validador de contenido se está modificando para agregar nuevas comprobaciones a fin de garantizar que el contenido de las Template Instances no incluya criterios coincidentes que coincidan con más campos de los que se proporcionan como entrada al intérprete de contenido. Esta corrección se lanzará a producción el 19 de agosto de 2024.

Evitar la creación de archivos problemáticos de Channel Files 291

El validador de contenido se modificó para permitir solo criterios de coincidencia de comodines en el campo 21, lo que evita el acceso fuera de los límites en los sensores que solo proporcionan 20 entradas.

5. La validación de la Template Instance debería expandirse para incluir pruebas dentro del intérprete de contenido

Resultados: Los nuevos Template Types se someten a pruebas de estrés en muchos aspectos, como la utilización de recursos, el impacto en el rendimiento del sistema y el volumen de detección. Para muchos Template Types, incluido el Template Type de IPC, se emplea una instancia específica con el fin de realizar una prueba de estrés comparándola con cualquier valor posible de los campos de datos asociados para identificar interacciones adversas del sistema.

Se ejecutó una prueba de esfuerzo del Template Type de IPC con una Template Instance de prueba en nuestro entorno de prueba, que consta de una variedad de sistemas operativos y cargas de workloads. El Template Type de IPC pasó la prueba de estrés y se validó para su uso, y se lanzó una Template Instance a producción como parte de una actualización de contenido de respuesta rápida.

Sin embargo, la Template Instance probada por el validador de contenido no observó que el número no coincidente de entradas causaría un fallo del sistema cuando el Template Type de IPC se lo proporcionara al intérprete de contenido.

Mitigación: actualizar los procedimientos de prueba del sistema de configuración de contenidos

El sistema de configuración de contenido se actualizó con nuevos procedimientos de prueba para garantizar que se pruebe cada nueva Template Instance, independientemente de que la Template Instance inicial se pruebe con el Template Type al crearse. Esto proporciona a las Template Instances pruebas adicionales antes de la implementación en producción.

6. Las Template Instances deben tener una implementación por etapas

Hallazgos: cada Template Instance debe implementarse en una implementación por etapas.

El sistema de configuración de contenido se actualizó con capas de implementación adicionales y comprobaciones de aceptación

La implementación por etapas mitiga el impacto si una nueva Template Instance provoca errores como bloqueos del sistema, picos de volumen de detección de falsos positivos o problemas de rendimiento. Las nuevas Template Instances que pasaron las pruebas canary se promoverán sucesivamente a anillos (grupos) de implementación más amplios o se revertirán si se detectan problemas. Cada anillo (grupo) está diseñado para identificar y mitigar posibles problemas antes de una implementación más amplia. La promoción de una Template Instance al siguiente anillo sucesivo va seguida de un tiempo de horneado adicional, en el que se recopilan los datos de telemetría para determinar el impacto general de la Template Instance en el endpoint.

Mitigación: proporcionar control al cliente sobre la implementación de actualizaciones de contenido de respuesta rápida

La plataforma Falcon se actualizó para proporcionar a los clientes un mayor control sobre la entrega de contenido de respuesta rápida. Los clientes pueden elegir dónde y cuándo se implementan las actualizaciones de contenido de respuesta rápida. Continuamos mejorando esta capacidad para proporcionar un control más granular sobre las implementaciones de contenido de respuesta rápida junto con detalles de actualización de contenido a través de notas de versión, a las que los clientes pueden suscribirse.

REVISIÓN INDEPENDIENTE DE TERCEROS

CrowdStrike contrató a dos proveedores independientes de seguridad de software externos para llevar a cabo una revisión adicional del código del sensor Falcon para garantizar la seguridad y la calidad. Además, estamos llevando a cabo una revisión independiente del proceso de calidad de principio a fin, desde el desarrollo hasta la implementación. Ambos proveedores comenzaron las revisiones con un enfoque inmediato en el código y el proceso afectados el 19 de julio.

DETALLES TÉCNICOS

Antecedentes y terminología

CrowdStrike envía contenido de seguridad a nuestro sensor de dos maneras: contenido del sensor (Sensor Content) que se envía con nuestro sensor directamente y contenido de respuesta rápida (Rapid Response Content) que está diseñado para responder al cambiante panorama de amenazas a velocidad operativa.

El procesamiento del contenido de respuesta rápida basado en expresiones regulares en el sensor implica varios componentes:

- **Intérprete de contenido:** parte del código C++ del sensor, que puede probar las cadenas de entrada con expresiones regulares.
- **Template Types:** contiene campos predefinidos para que los ingenieros de detección de amenazas los aprovechen en el contenido de respuesta rápida. Los Template Types se expresan en código y se compilan en el sensor en el momento del desarrollo.
- **Archivo de definiciones de Template Type:** define los parámetros de cada Template Type. Las definiciones de este archivo incluyen información sobre qué channel file entregará el contenido de respuesta rápida para cada Template Type, cuántas entradas debe usar el Template Type y qué tipo de datos se requieren para cada entrada.
- **Contenido del sensor:** determina cómo combinar los datos relevantes para la seguridad con el contenido de respuesta rápida para tomar ciertas decisiones de detección. El contenido del sensor incluye modelos de inteligencia artificial (IA) y aprendizaje automático (machine learning) en el sensor, así como Template Types. Se compila como parte de la versión del sensor.
- **Template Instances:** criterios de coincidencia desarrollados por los ingenieros de detección. Las Template Instances consisten en contenido regex destinado a ser empleado con un Template Type específico. Las Template Instances identifican datos específicos para su uso en operaciones de seguridad. Las Template Instances se definen mediante una interfaz de usuario controlada por el archivo Definiciones de Template Type.
- **Contenido de respuesta rápida:** consta de varias Template Instances agrupadas. El contenido de respuesta rápida se entrega por channel file.
- **Validador de contenido**: verifica la validez de los channel files con su definición en el archivo de definiciones de Template Types.
- **Sistema de configuración de contenidos**: se utiliza para crear Template Instances, que se validan y despliegan en el sensor a través de un mecanismo denominado **archivos de canal** (Channel Files).

Uso del controlador del núcleo en un producto de seguridad

Como señaló David Weston en el blog de seguridad de Microsoft, los productos de seguridad del ecosistema de Windows, incluido el sensor Falcon, suelen aprovechar los controladores (drivers) del kernel como componentes principales de una oferta de seguridad estable.

La presencia en el núcleo (kernel) ofrece una amplia visibilidad de las actividades relevantes para la seguridad de todo el sistema, como la creación de procesos e hilos, o los archivos que se escriben, borran y modifican en el disco. Las interfaces expuestas por el

núcleo permiten a los controladores (drivers) de CrowdStrike aplicar controles críticos para un producto de seguridad, como la prevención en línea de procesos maliciosos o el bloqueo de archivos de malware que se escriben en el disco.

El controlador del kernel de CrowdStrike se carga desde una fase inicial del arranque del sistema para permitir que el sensor observe y se defienda contra el malware que se inicia antes de que se pongan en marcha los procesos del modo de usuario (user mode).

Proporcionar contenido de seguridad actualizado (por ejemplo, el contenido de respuesta rápida de CrowdStrike) a estas capacidades del kernel permite al sensor defender los sistemas contra un panorama de amenazas en rápida evolución sin realizar cambios en el código del kernel. El contenido de respuesta rápida son datos de configuración; no es código ni un controlador del kernel.

CrowdStrike certifica cada nueva versión del sensor de Windows a través del programa Windows Hardware Quality Labs (WHQL), que incluye pruebas exhaustivas a través de todas las pruebas requeridas en el Kit de laboratorio de hardware de Windows (HLK) y el Kit de certificación de hardware de Windows (HCK) de Microsoft. El proceso de certificación WHQL marca el final de una completa serie de pruebas internas que incluyen pruebas funcionales, pruebas de longevidad, pruebas de estrés con inyección de fallos, fuzzing y pruebas de rendimiento. Durante las pruebas requeridas para el programa WHQL, los sensores emplean las últimas versiones de los archivos de canal en el momento de la certificación.

A medida que las nuevas versiones de Windows introducen soporte para realizar más de estas funciones de seguridad en el espacio del usuario, CrowdStrike actualiza su agente para utilizar este soporte. Queda mucho trabajo por hacer para que el ecosistema de Windows admita un producto de seguridad robusto que no dependa de un controlador de kernel para al menos algunas de sus funcionalidades. Nos comprometemos a trabajar directamente con Microsoft de forma continua a medida que Windows sigue agregando más soporte para las necesidades de productos de seguridad en el espacio de usuario.

Análisis de volcado de memoria

Para ilustrar cómo las nuevas Template Instances en el Channel File 291 provocaron un bloqueo del sistema, examinamos brevemente un volcado de memoria del kernel de un sistema afectado por el contenido causante del problema. Esto amplía el análisis del fallo compartido por David Weston en el blog de seguridad de Microsoft.

Abriendo el crash dump en el depurador del kernel de Windows y usando el comando estándar !analyze -v para obtener un resumen rápido, vemos que se produjo una comprobación de errores de memoria (también conocida como “violación de acceso”).
(Nota: Los detalles de depuración no relacionados se omiten por brevedad, y aquí se

analiza una descarga de fallos representativa. Existen variaciones de la descarga, dependiendo de los detalles del estado de la máquina.)

```
1: kd> !analyze -v
*******************************************************************************
*                                                                             *
*                        Bugcheck Analysis                                    *
*                                                                             *
*******************************************************************************

PAGE_FAULT_IN_NONPAGED_AREA (50)
Invalid system memory was referenced.  This cannot be protected by try-except.
Typically the address is just plain bad or it is pointing at freed memory.
Arguments:
Arg1: ffffd6030000006a, memory referenced.
Arg2: 0000000000000000, X64: bit 0 set if the fault was due to a not-present PTE.
        bit 1 is set if the fault was due to a write, clear if a read.
        bit 3 is set if the processor decided the fault was due to a corrupted PTE.
        bit 4 is set if the fault was due to attempted execute of a no-execute PTE.
        - ARM64: bit 1 is set if the fault was due to a write, clear if a read.
        bit 3 is set if the fault was due to attempted execute of a no-execute PTE.
Arg3: fffff8020ebc14ed, If non-zero, the instruction address which referenced the bad memory
        address.
Arg4: 0000000000000002, (reserved)

READ_ADDRESS:  ffffd6030000006a Paged pool

MM_INTERNAL_CODE:  2

IMAGE_NAME:  csagent.sys

MODULE_NAME: csagent

FAULTING_MODULE: fffff8020eae0000 csagent

PROCESS_NAME:  System

TRAP_FRAME:  ffffae035f57eca0 -- (.trap 0xffffae035f57eca0)
NOTE: The trap frame does not contain all registers.
Some register values may be zeroed or incorrect.
rax=ffffae035f57f280 rbx=0000000000000000 rcx=0000000000000003
rdx=ffffae035f57f250 rsi=0000000000000000 rdi=0000000000000000
rip=fffff8020ebc14ed rsp=ffffae035f57ee30 rbp=ffffae035f57ef30
 r8=ffffd6030000006a  r9=0000000000000000 r10=0000000000000000
r11=0000000000000014 r12=0000000000000000 r13=0000000000000000
r14=0000000000000000 r15=0000000000000000
iopl=0         nv up ei ng nz na po nc
csagent+0xe14ed:
fffff802`0ebc14ed 458b08          mov     r9d,dword ptr [r8] ds:ffffd603`0000006a=????????
Resetting default scope

STACK_TEXT:
ffffae03`5f57ea78 fffff802`05add2da     : 00000000`00000050 ffffd603`0000006a 00000000`00000000
ffffae03`5f57eca0 : nt!KeBugCheckEx
ffffae03`5f57ea80 fffff802`05947efc     : ffffd603`000ed454 00000000`00000000 00000000`00000000
ffffd603`0000006a : nt!MiSystemFault+0x1bc19a
```

```
ffffae03`5f57eb80 fffff802`05a2707e     : 00000000`00000000 ffffd603`e33a019e ffffae03`5f57f0a0
ffffae03`5f57f0a0 : nt!MmAccessFault+0x29c
ffffae03`5f57eca0 fffff802`0ebc14ed     : 00000000`00000000 ffffae03`5f57ef30 ffffd603`f208200c
ffffd603`f207a05c : nt!KiPageFault+0x37e
ffffae03`5f57ee30 fffff802`0eb9709e     : 00000000`00000000 00000000`e01f008d ffffae03`5f57f202
fffff802`0ed6aaf8 : csagent+0xe14ed
ffffae03`5f57efd0 fffff802`0eb98335     : 00000000`00000000 00000000`00000010 00000000`00000002
ffffd603`f207a01c : csagent+0xb709e
ffffae03`5f57f100 fffff802`0edd20c7     : 00000000`00000000 00000000`00000000 ffffae03`5f57f402
00000000`00000000 : csagent+0xb8335
ffffae03`5f57f230 fffff802`0edcec44     : ffffae03`5f57f6e8 fffff802`060abae0 ffffd603`ed408580
00000000`00000003 : csagent+0x2f20c7
ffffae03`5f57f4b0 fffff802`0eb47a31     : 00000000`0000303b ffffae03`5f57f770 ffffd603`edc908a0
ffffc189`7fcd4098 : csagent+0x2eec44
ffffae03`5f57f670 fffff802`0eb46aee     : ffffd603`edc908a0 fffff802`0ebf1e7e 00000000`00006820
fffff802`0ed3f8f0 : csagent+0x67a31
ffffae03`5f57f7e0 fffff802`0eb4685b     : ffffae03`5f57fa58 ffffd603`edc97830 ffffd603`edc908a0
ffffc189`7f90f4b8 : csagent+0x66aee
ffffae03`5f57f850 fffff802`0ebe99ea     : 00000000`f047f4ef ffff49ac`ca0f55d4 00000000`00000000
ffffd603`ec18fc30 : csagent+0x6685b
ffffae03`5f57f8d0 fffff802`0eb3efbb     : 00000000`00000000 ffffae03`5f57fad9 ffffc189`7f90f010
ffffc189`7f7ea470 : csagent+0x1099ea
ffffae03`5f57fa00 fffff802`0eb3edd7     : ffffc189`7ab79000 00000000`00000000 ffffc189`7f90f010
ffffc189`00000001 : csagent+0x5efbb
ffffae03`5f57fb40 fffff802`0ebde681     : 00000000`00000000 00000000`00000000 ffffc189`7f5a97d0
ffffc189`7f7ea470 : csagent+0x5edd7
ffffae03`5f57fb70 fffff802`05879ca7     : ffffc189`7faa8040 00000000`00000080 fffff802`0ebde510
00000000`00000000 : csagent+0xfe681
ffffae03`5f57fbb0 fffff802`05a1af64     : ffffe601`bcf51180 ffffc189`7faa8040 fffff802`05879c50
00000000`00000000 : nt!PspSystemThreadStartup+0x57
ffffae03`5f57fc00 00000000`00000000     : ffffae03`5f580000 ffffae03`5f579000 00000000`00000000
00000000`00000000 : nt!KiStartSystemThread+0x34
```

Este comando de clasificación automatizado identifica a `csagent.sys` como el controlador que realiza el acceso a la memoria fuera de los límites. `csagent.sys` es el controlador de filtro del sistema de archivos de CrowdStrike, un tipo de controlador del kernel que se registra con componentes del sistema operativo Windows para recibir notificaciones de actividades del sistema relevantes para la seguridad en tiempo real.

Entre las notificaciones para las que se registra el controlador de CrowdStrike se encuentra una notificación para la creación de Named Pipes. Cuando el controlador recibe una notificación de Named Pipe, estos datos se combinan con otra información contextual sobre el sistema. Estos datos combinados se presentan para su evaluación con respecto a las Template Instances transmitidas en el Channel File 291.

Para observar más de cerca este proceso, vemos el estado del registro en el punto de la lectura de la memoria fuera de los límites restaurando el trap frame y desensamblando las instrucciones anteriores para orientarnos. *(Nota: Esta lista de desensamblado se modificó a partir de la salida del depurador estándar para anotar el código con nombres de símbolos ilustrativos).*

```
1: kd> .trap 0xffffae035f57eca0
NOTE: The trap frame does not contain all registers.
Some register values may be zeroed or incorrect.
rax=ffffae035f57f280 rbx=0000000000000000 rcx=0000000000000003
rdx=ffffae035f57f250 rsi=0000000000000000 rdi=0000000000000000
rip=fffff8020ebc14ed rsp=ffffae035f57ee30 rbp=ffffae035f57ef30
 r8=ffffd6030000006a  r9=0000000000000000 r10=0000000000000000
r11=0000000000000014 r12=0000000000000000 r13=0000000000000000
r14=0000000000000000 r15=0000000000000000
iopl=0         nv up ei ng nz na po nc
csagent+0xe14ed:
fffff802`0ebc14ed 458b08          mov     r9d,dword ptr [r8]
ds:ffffd603`0000006a=????????

1: kd> u @rip-16 L0n10
csagent!TemplateGetString+0xe:
fffff802`0ebc14d7 4e8b04d8        mov     r8,qword ptr [rax+r11*8]
fffff802`0ebc14db 750b            jne     csagent!TemplateGetString+0x1f
(fffff802`0ebc14e8)
fffff802`0ebc14dd 4d85c0          test    r8,r8
fffff802`0ebc14e0 7412            je      csagent!TemplateGetString+0x2b
(fffff802`0ebc14f4)
fffff802`0ebc14e2 450fb708        movzx   r9d,word ptr [r8]
fffff802`0ebc14e6 eb08            jmp     csagent!TemplateGetString+0x27
(fffff802`0ebc14f0)
fffff802`0ebc14e8 4d85c0          test    r8,r8
fffff802`0ebc14eb 7407            je      csagent!TemplateGetString+0x2b
(fffff802`0ebc14f4)
fffff802`0ebc14ed 458b08          mov     r9d,dword ptr [r8]
fffff802`0ebc14f0 4d8b5008        mov     r10,qword ptr [r8+8]
```

Antes de este fragmento de código, los datos de contexto de la notificación de canalización nombrada se habían preparado para el Template Type de IPC como una matriz de 20 punteros de entrada, cada uno apuntando a una estructura de cadena que contiene una dirección de búfer y un valor de tamaño. Este fragmento tiene la intención de seleccionar una de las entradas para devolver su dirección y tamaño de búfer, de acuerdo con un índice especificado por Channel File 291.

A medida que ingresamos este código, la dirección de la matriz de punteros de 20 entradas se mantiene en el registro rax, y el registro r11 indica que la entrada a recuperar está en el índice 0x14, es decir, el elemento 21.

Examinando la matriz de entrada, encontramos una matriz de 20 punteros a estructuras de cadenas de entrada, seguidas de un valor 21 que no *apunta* a una memoria válida:

```
1: kd> dp @rax l0n21
ffffae03`5f57f280  ffffae03`5f57f320 ffffae03`5f57f330
ffffae03`5f57f290  ffffae03`5f57f340 ffffae03`5f57f350
ffffae03`5f57f2a0  ffffae03`5f57f360 ffffae03`5f57f370
ffffae03`5f57f2b0  ffffae03`5f57f380 ffffae03`5f57f390
ffffae03`5f57f2c0  ffffae03`5f57f3a0 ffffae03`5f57f3b0
ffffae03`5f57f2d0  ffffae03`5f57f3c0 ffffae03`5f57f3d0
ffffae03`5f57f2e0  ffffae03`5f57f3e0 ffffae03`5f57f3f0
ffffae03`5f57f2f0  ffffae03`5f57f400 ffffae03`5f57f410
ffffae03`5f57f300  ffffae03`5f57f420 ffffae03`5f57f430
ffffae03`5f57f310  ffffae03`5f57f440 ffffae03`5f57f450
ffffae03`5f57f320  ffffd603`0000006a
1: kd> !pte ffffd603`0000006a
                                           VA ffffd6030000006a
PXE at FFFFFE7F3F9FCD60    PPE at FFFFFE7F3F9AC060    PDE at FFFFFE7F3580C000
PTE at FFFFFE6B01800000
contains 0A00000107A00863  contains 0000000000000000
pfn 107a00    ---DA--KWEV  contains 0000000000000000
not valid
```

Luego de leer este puntero no válido en el registro r8, el flujo de control en el fragmento de código anterior da el primer salto para `abordar fffff802`0ebc14e8`, realiza una comprobación de puntero NULL y, a continuación, intenta una lectura a través del puntero no válido, lo que da como resultado una lectura fuera de los límites y una comprobación de errores posterior.

RECURSOS ADICIONALES

Inserta referencias y enlaces a recursos técnicos adicionales.

Centro de remediación y orientación: actualización de contenido de Falcon para hosts con sistema operativo Windows

Blog: Detalles técnicos: actualización de contenido Falcon para hosts con sistema operativo Windows

Centro de remediación: glosario de términos

Este documento es una traducción de la siguiente versión en inglés https://www.crowdstrike.com/wp-content/uploads/2024/08/Channel-File-291-Incident-Root-Cause-Analysis-08.06.2024.pdf. Esta versión traducida se proporciona únicamente para facilitar su comprensión y por motivos de conveniencia. En caso de conflicto o ambigüedad, la versión en inglés siempre prevalecerá y tendrá prioridad.