Análisis técnico externo de la causa raíz — Channel File 291

INTRODUCCIÓN

Este informe detalla la información previamente compartida en nuestro Informe Preliminar Posterior al Incidente y profundiza en los hallazgos, las mitigaciones, los detalles técnicos y el análisis de la causa raíz del incidente. El 29 de julio a las 17:00 horas (hora del Pacífico), utilizando una comparación entre semanas, el 99 % de los sensores de Windows estaban conectados en comparación con antes de la actualización de contenidos. Normalmente vemos una variación de ~1 % semana a semana en las conexiones de los sensores.

A lo largo de este Análisis de la Causa Raíz (RCA), hemos utilizado terminología general para describir la plataforma CrowdStrike Falcon con el fin de que resulte más comprensible. La terminología en otra documentación puede ser más específica y técnica.

QUÉ HA OCURRIDO

El sensor CrowdStrike Falcon ofrece una potente inteligencia artificial y modelos de aprendizaje automatizado (machine learning) en el sensor para proteger los sistemas de los clientes mediante la identificación y corrección de las amenazas avanzadas más recientes. Estos modelos se mantienen actualizados y se refuerzan con lo aprendido 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 cuando los datos se filtran y agregan en cada sensor en un repositorio de gráficos local.

Cada sensor correlaciona el contexto de su repositorio de gráficos local con la actividad del sistema en tiempo real en los comportamientos e indicadores de ataque (IOA) en un proceso continuo de refinamiento. Este proceso de refinamiento incluye un Motor de Detección del Sensor que combina el Contenido del sensor integrado con el Contenido para Respuesta Rápida emitido desde la nube. El Contenido para Respuesta Rápida se utiliza para recopilar datos de telemetría, identificar indicadores de comportamiento del adversario (IOA) y aumentar las nuevas detecciones y prevenciones en el sensor sin necesidad de cambiar el código del sensor. El Contenido para Respuesta Rápida son investigaciones de comportamiento, separadas y distintas de las capacidades de prevención y detección mediante inteligencia artificial en el sensor de CrowdStrike.

El Contenido para Respuesta Rápida se entrega a través de Archivos de Canal (Channel Files) y es interpretado por el Intérprete de Contenido del sensor, utilizando un motor basado en expresiones regulares. Cada archivo de canal de Contenido para Respuesta

Rápida está asociado a un Tipo de Plantilla (Template Type) específico integrado en una versión del sensor. El Template Type proporciona al Intérprete de Contenido los datos de actividad y contexto gráfico para ser mapeados con el Contenido para Respuesta Rápida.

Con el lanzamiento de la versión 7.11 del sensor en febrero de 2024, CrowdStrike introdujo un nuevo Template Type para facilitar la visibilidad y la detección de nuevas técnicas de ataque que abusan de Named Pipes y otros mecanismos de comunicación entre procesos (InterProcessCommunication o"IPC") de Windows. Como se describe en el PIR, el nuevo Template Type IPC se desarrolló y probó de acuerdo con nuestros procesos estándar de desarrollo deContenido del Sensor y se integró en el sensor para prepararlo para su uso sobre el terreno. Las Instancias de Plantilla IPC (IPC Template Instances) se entregan como Contenido para Respuesta Rápida a los sensores a través de un Channel File correspondiente con el número 291.

El nuevo Template Type IPC definía 21 campos de parámetros de entrada, pero el código de integración que invocaba al Intérprete de Contenido con las Template Instances del Channel File 291 solo proporcionaba 20 valores de entrada con los que hacerlos 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 de Template Type (mediante el uso de una Template Instance de prueba) o las primeras implementaciones exitosas de Template Instances IPC en el terreno. En parte, esto se debió al uso de criterios de coincidencia de wildcards para la entrada 21 tanto durante las pruebas como en las IPC Template Instances iniciales.

El 19 de julio de 2024, se implementaron dos IPC Template Instances adicionales. Uno de ellos introdujo un criterio de coincidencia sin wildcards para el parámetro de entrada número 21. Estas nuevas Template Instances dieron lugar a 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 entregó a los sensores, ninguna IPC Template Instance en versiones anteriores del canal había hecho uso del campo del parámetro de entrada 21. El Validador de Contenido evaluó las nuevas Template Instances, pero basó su evaluación en la expectativa de que el IPC Template Type tuviera 21 entradas.

Los sensores que recibieron la nueva versión del Channel File 291 que contenía el contenido problemático se expusieron a un problema latente de lectura fuera de límites en el Intérprete de Contenido. En la siguiente notificación de IPC del sistema operativo, se evaluaron las nuevas IPC Template Instances y se especificó 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 límites más allá del final de la matriz de datos de entrada y dio lugar a un bloqueo del sistema (BSOD o pantalla azul).

En resumen, fue la confluencia de estos problemas lo que provocó un bloqueo del sistema (BSOD o pantalla azul): la falta de coincidencia entre las 21 entradas validadas por el Validador de Contenido y las 20 proporcionadas al Intérprete de Contenido, el problema latente de lectura fuera de límites en el Intérprete de Contenido y la falta de una prueba específica para comprobar los criterios de coincidencia sinwildcard en el campo 21. Si bien este escenario con Channel File 291 ya no puede repetirse, también informa de las mejoras del proceso y las medidas de mitigación que CrowdStrike está implementando para garantizar una mayor resiliencia.

CONCLUSIONES Y MITIGACIONES

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

Conclusiones: En el momento del incidente, el código del sensor del IPC Template Type describía 20 fuentes de entrada diferentes que utilizaba la Template Instance. Esto significa que cuando el sensor quería tomar una decisión de detección basándose en el IPC Template Type, el código del sensor proporcionaba 20 fuentes de entrada diferentes al Intérprete de Contenido. Sin embargo, la definición del IPC Template Type en el archivo Template Types Definitions decía que esperaba 21 campos de entrada. Esta definición dio lugar a Template Instances en el Channel File 291 que se esperaba que funcionaran con 21 entradas. Esta falta de coincidencia no se detectó durante el desarrollo del IPC Template Type. Los casos de prueba y el Contenido para Respuesta Rápida utilizados para probar el IPC Template Type no provocaron ningún error 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 del Template Type en el momento de compilar el sensor

El 19 de julio de 2024 se desarrolló un parche para el compilador de Contenido del Sensor que valida el número de entradas proporcionadas por un Template Type, que 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 proporcionar un número incorrecto de entradas en ninguna plataforma.

2. Se carecía de una comprobación de los límites de la matriz de tiempo de ejecución para los campos de entrada del Intérprete de Contenido en el Channel File 291.

Conclusiones: el Contenido para Respuesta Rápida para el Channel File 291 instruyó al Intérprete de Contenido a leer la entrada 21 de la matriz de punteros de entrada. Sin embargo, el IPC Template Type solo genera 20 entradas. Como resultado, una vez que se entregó el Contenido para Respuesta Rápida que utilizó un criterio de coincidencia sin wildcards para la entrada número 21, el Intérprete de Contenido realizó una lectura fuera de límites de la matriz de entrada. Este no es un problema de escritura arbitraria en memoria y se ha revisado de forma independiente.

Mitigación: Añadir comprobaciones de límites de matriz de entrada en tiempo de ejecución al Intérprete de Contenido para obtener Contenido para Respuesta Rápida en el Channel File 291

El 25 de julio de 2024 se añadió la comprobación de límites a la función Intérprete de Contenido que recupera las cadenas de entrada. Al mismo tiempo, se añadió una comprobación adicional de que el tamaño de la matriz de entradas coincide con el número de entradas esperado por el Contenido para Respuesta Rápida. Estas correcciones se están adaptando a todas las versiones de sensores de Windows 7.11 y superiores a través de una versión de revisión del software del sensor. Esta versión estará disponible con carácter general el 9 de agosto de 2024.

La comprobación de límites añadida impide que el Intérprete de Contenido acceda fuera de límites a la matriz de entrada y bloquee el sistema (BSOD o pantalla azul). La comprobación adicional añade una capa de validación en tiempo de ejecución de que el tamaño de la matriz de entrada coincide con el número de entradas esperado por el Contenido para Respuesta Rápida.

Hemos completado pruebas de exploración de vulnerabilidades mediante datos aleatorios (fuzzing) del Template Type del channel file 291 y las estamos ampliando para incluir más gestores de Contenido para Respuesta Rápida en el sensor.

Mitigación: Corregir el número de entradas que proporciona el IPC Template Type

El código del sensor que define el IPC Template Type se actualizó para proporcionar el número correcto de entradas (21). Esta corrección se está adaptando a todas las versiones de sensores de Windows 7.11 y superiores a través de una versión de revisión del software del sensor. Esta versión estará disponible con carácter general el 9 de agosto de 2024.

3. Las pruebas de Template Types deberían cubrir una variedad más amplia de criterios de coincidencia

Conclusiones: Durante el desarrollo del IPC Template Type se realizaron pruebas manuales y automatizadas. 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 los criterios creados en los 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 IPC Template Type 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 telemetría y alertas de detección. Parte de estas pruebas incluyó definir un Channel Filepara 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 basado en una regex con wildcards
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 límites en el Intérprete de Contenido cuando se le proporcionaron 20 entradas en lugar de 21.

Mitigación: Aumentar la cobertura de las pruebas durante el desarrollo del Template Type

Para confirmar que estamos validando todos los campos en cada Template Type, se han creado pruebas automatizadas que prueban con criterios de coincidencia sin wildcards para cada campo. Este paso se ha hecho 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

Conclusiones: El Validador de Contenido evaluó las nuevas Template Instances. Sin embargo, basó su evaluación en la expectativa de que el IPC Template Type se proporcionara con 21 entradas. Esto provocó que la Template Instance problemática se enviara al Intérprete de Contenido.

Mitigación: Crear comprobaciones adicionales en el Validador de Contenido

El Validador de Contenido se está modificando para añadir nuevas comprobaciones y garantizar que el contenido de las Template Instances no incluya criterios de coincidencia que coincidan en más campos de los que se proporcionan como entrada al Intérprete de Contenido. Esta solución se lanzará a producción el 19 de agosto de 2024.

Mitigación: Evitar la creación de Channel Files 291 problemáticos

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

5. La validación de Template Instances debería ampliarse para incluir las pruebas en el Intérprete de Contenido

Conclusiones: Los Template Types recién lanzados 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 IPC Template Type , se utiliza una Template Instance específica para someter a una prueba de estrés a la Template Type, comparándola con cualquier valor posible de los campos de datos asociados para identificar interacciones adversas del sistema.

Se ejecutó una prueba de estrés del IPC Template Type con una Template Instance de prueba en nuestro entorno de pruebas, que consta de diversos sistemas operativos y cargas de trabajo. El IPC Template Type pasó la prueba de estrés y se validó para su uso, y se lanzó una Template Instance a producción como parte de una actualización del Contenido para 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 IPC se lo proporcionó al Intérprete de Contenido.

Mitigación: Actualizar los procedimientos de prueba del Sistema de Configuración del Contenido

El Sistema de Configuración del Contenido se ha actualizado con nuevos procedimientos de pruebas para garantizar que se pruebe cada nueva Template Instance, independientemente de que la Template Instance inicial se pruebe con el Template Type en el momento de la creación. Esto proporciona a las Template Instances pruebas adicionales antes de la implementación en producción.

6. Las Template Instances deben tener una implementación escalonada

Conclusiones: Cada Template Instance debe desplegarse por etapas.

Mitigación: El Sistema de Configuración del Contenido se ha actualizado con capas de despliegue y comprobaciones de aceptación adicionales

La implementación por etapas mitiga el impacto si una nueva Template Instance provoca errores como bloqueos del sistema (BSOD o pantalla azul), picos en el volumen de detecciones de falsos positivos o problemas de rendimiento. Las nuevas Template Instances que hayan pasado 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. A la promoción de una Template Instance al siguiente anillo sucesivo le sigue un tiempo de preparación adicional, donde se recopilan datos de telemetría para determinar el impacto general de la Template Instance en el endpoint.

Mitigación: Proporcionar al cliente control sobre la implementación de actualizaciones de Contenido para Respuesta Rápida

La plataforma Falcon se ha actualizado para proporcionar a los clientes un mayor control sobre la entrega de Contenido para Respuesta Rápida. Los clientes pueden elegir dónde y cuándo se despliegan las actualizaciones del Contenido para Respuesta Rápida. Seguimos mejorando esta capacidad para proporcionar un control más detallado de las implementaciones de Contenido para Respuesta Rápida, junto con los detalles de actualización de contenidos mediante notas de publicación, a las que pueden suscribirse los clientes.

REVISIÓN POR TERCEROS INDEPENDIENTES

CrowdStrike ha contratado a dos proveedores independientes de software de seguridad para que revisen más a fondo el 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 han comenzado las revisiones con un enfoque inmediato en el código y el proceso afectados el 19 de julio.

DETALLES TÉCNICOS

Antecedentes y terminología

CrowdStrike proporciona actualizaciones de la configuración del contenido de seguridad a nuestros sensores de dos maneras: como Contenido del Sensor, que se envía directamente con nuestro sensor, y como Contenido para Respuesta Rápida, que está diseñado para responder al panorama cambiante de amenazas a una velocidad operativa.

El procesamiento del Contenido para 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:** Contienen campos predefinidos para que los ingenieros de detección de amenazas los aprovechen en el Contenido para Respuesta Rápida. Los Template Types se expresan en código y se compilan en el sensor en el momento de la creación.
- **Archivo Template Type Definitions:** Define los parámetros de cada Template Type. Las definiciones de este archivo incluyen información sobre qué archivo de canal entregará el Contenido para 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 para Respuesta Rápida para tomar ciertas decisiones de detección. El Contenido del Sensor incluye modelos de inteligencia artificial y aprendizaje automático (machine learning) en el sensor, así como Template Types. Se compila como parte de la versión del sensor.
- **Template Instances: C**riterios de coincidencia desarrollados por ingenieros de detección. Las Template Instances constan de contenido regex destinado a su uso 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 Template Type Definitions.
- **Contenido para Respuesta Rápida: C**onsta de varias Template Instances agrupadas. El Contenido para Respuesta Rápida se entrega por Channel Files.
- **Validador de Contenido**: Comprueba la validez de los Channel Filescomparándola con su definición en el archivo Template Type Definitions.
- **Sistema de Configuración del Contenido**: Se utiliza para crear Template Instances, que se validan y se implementan en el sensor a través de un mecanismo llamado **Channel Files**.

Uso del controlador del núcleo (kernel)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 sólida.

La presencia en el kernel ofrece una amplia visibilidad de las actividades relevantes para la seguridad de todo el sistema, como la creación de procesos e hilos, o la escritura, eliminación y modificación de los archivos en el disco. Las interfaces expuestas por el kernel permiten a los driversde CrowdStrike aplicar controles críticos para un producto de seguridad, como la prevención en líneade procesos maliciosos o el bloqueo de archivos de malware que se escriben en el disco.

El driverdel kernel de CrowdStrike se carga desde una fase temprana del arranque del sistema para permitir que el sensor observe y se defienda del malware que se inicia antes de que comiencen los procesos en user mode.

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

CrowdStrike certifica cada nueva versión de sensores de Windows a través del programa Windows Hardware Quality Labs (WHQL), que incluye pruebas exhaustivas con todas las pruebas necesarias en los kits de laboratorio de hardware (HLK) y Windows Hardware Certification Kit (HCK) de Microsoft. El proceso de certificación WHQL marca el final de un exhaustivo desafío de pruebas internas que incluye pruebas funcionales, pruebas de longevidad, pruebas de esfuerzo con inyección de fallos, pruebas de borrado y de rendimiento. Durante las pruebas requeridas para el programa WHQL, los sensores utilizan las últimas versiones de los Channel Files 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. El ecosistema de Windows aún tiene mucho trabajo por hacer para apoyar un producto de seguridad sólido que no dependa de un driver de kernel para al menos algunas de sus funciones. Nos comprometemos a trabajar directamente con Microsoft de forma continua a medida que Windows sigue añadiendo 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 del Channel File 291 provocaron un fallo del sistema, examinamos brevemente un volcado de memoria del kernel de un sistema afectado por el contenido problemático. Esto amplía el análisis del fallo compartido por David Weston en el blog de Seguridad de Microsoft.

Al abrir el volcado de memoria en el depurador del kernel de Windows y usar el comando estándar !analyze -v para obtener un resumen rápido, vemos que se ha producido un código de comprobación de errores (bugcheck) de fallo de memoria (también conocida

como "violación de acceso"). *(Nota: Los detalles de depuración no relacionados se omiten por brevedad y aquí se analiza un volcado de memoria representativo. Existen variaciones del volcado, 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 triaje automatizado identifica a `csagent.sys` como el driverque realiza el acceso a la memoria fuera de límites. `csagent.sys` es el driver de filtro del sistema de archivos de CrowdStrike, un tipo de driver de kernel que se registra con los 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 driver de CrowdStrike hay una notificación para la creación de Namedpes. Cuando el driver recibe una notificación de Named Pipes, estos datos se combinan con otra información contextual sobre el sistema. Estos datos combinados se presentan para su evaluación comparándolos con las Template Instances incluidas en el Channel File 291.

Para ver más de cerca este proceso, visualizamos el estado del registro en el punto de la lectura de memoria fuera de límites restaurando el trap frame y desensamblando las instrucciones precedentes para orientarnos. *(Nota: Este listado de desmontaje se ha*

modificado a partir de la salida estándar del depurador 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 del Named Pipe se preparaban para el IPC Template Type como una matriz de 20 punteros de entrada, cada uno de los cuales apuntaba a una estructura de cadenas que contiene una dirección de búfer y un valor de tamaño. Este fragmento pretende seleccionar una de las entradas para devolver la dirección y el tamaño del búfer, según un índice especificado en el Channel File 291.

Al introducir 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 que se va 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
```

Después de leer este puntero no válido en el registro `r8`, el flujo de control en el fragmento de código anterior da el primer salto a la dirección `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 límites y el consiguiente código de comprobación de errores (bugcheck).

RECURSOS ADICIONALES

Inserta referencias y enlaces a recursos técnicos adicionales.
Centro de corrección y guía: Actualización de contenido de Falcon para hosts de Windows
Blog: Detalles técnicos: Actualización de contenido de Falcon para hosts de Windows
Centro de corrección: 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 para mayor claridad. En caso de conflicto o ambigüedad, la versión en inglés siempre prevalecerá y tendrá prioridad.