External Technical Root Cause Analysis (RCA) — Channel File 291

INTRODUZIONE

Il presente report riporta le informazioni precedentemente condivise nel nostro Esame preliminare post-incidente, fornendo ulteriori dettagli circa le risultanze, le mitigazioni, i dettagli tecnici e l'analisi delle cause principali dell'incidente. Il 29 luglio alle 17:00 PT, in base ad una verifica settimanale circa il 99% dei sensori Windows risultava online rispetto a prima che i contenuti venissero aggiornati. Solitamente le connessioni dei sensori variano di circa l'1% di settimana in settimana.

In questa RCA, adottiamo una terminologia generica per descrivere le componenti della piattaforma CrowdStrike Falcon utilizzando un linguaggio chiaro al fine di agevolare la lettura. La terminologia adottata in altri documenti potrebbe invece essere più specifica e tecnica.

COS'È SUCCESSO

Il sensore CrowdStrike Falcon utilizza potenti modelli di intelligenza artificiale e machine learning per proteggere i sistemi dei clienti identificando e risolvendo le minacce avanzate più recenti. Questi modelli vengono aggiornati e potenziati costantemente grazie all'apprendimento effettuato sulla telemetria fornita dal sensore e grazie all'ingegno umano degli ingegneri dei team di Falcon Adversary OverWatch, Falcon Complete e del team di threat detection di CrowdStrike. Questo ricco set di telemetria di sicurezza inizia quando i dati vengono filtrati e aggregati su ciascun sensore in un archivio locale.

Ogni sensore crea correlazioni tra il contesto del suo archivio locale con l'attività del sistema in tempo reale, trasformando questi dati in comportamenti e indicatori di attacco (IOA), in un processo continuo di perfezionamento. Questo processo di perfezionamento include un Sensor Detection Engine che combina i Sensor Content integrati con i Rapid Response Content forniti dal cloud. I Rapid Response Content vengono utilizzati per raccogliere dati di telemetria, identificare gli indicatori del comportamento degli attori malevoli e aumentare i nuovi rilevamenti e prevenzioni sul sensore senza richiedere modifiche al suo codice. Il Rapid Response Content utilizza euristiche comportamentali, separate e distinte dalle funzionalità di prevenzione e rilevamento dell'AI su sensore di CrowdStrike.

I Rapid Response Content vengono forniti tramite Channel File e interpretati dal Content Interpreter del sensore utilizzando un motore basato su espressioni regolari. Ogni channel file dei Rapid Response Content è associato a un Template Type specifico integrato in una

versione del sensore. Il Template Type fornisce al Content Interpreter i dati delle attività e il contesto del grafico da confrontare con i Rapid Response Content.

Con il rilascio della versione 7.11 del sensore a febbraio 2024, CrowdStrike ha introdotto un nuovo Template Type per consentire la visibilità e il rilevamento di nuove tecniche di attacco che abusano delle c.d. "named pipes" e di altri meccanismi di comunicazione interprocesso ("IPC") di Windows. Come indicato nella PIR, il nuovo Template Type IPC è stato sviluppato e testato secondo i nostri processi standard di sviluppo del Sensor Content ed è stato integrato nel sensore in vista dell'utilizzo sul campo. Le Template Instances IPC vengono fornite come Rapid Response Content ai sensori tramite un Channel File corrispondente, il numero 291.

Il Template Type IPC definiva 21 campi di parametri di input, ma il codice di integrazione che richiamava il Content Interpreter con le Template Instances del Channel File 291 forniva invece solo 20 valori di input da confrontare. Questa discrepanza nel conteggio dei parametri ha eluso diversi livelli di convalida e test, e non è stata rilevata durante il processo di test di distribuzione del sensore, lo stress test del Template Type (effettuato utilizzando una Template Instance di prova) o i primi, numerosi utilizzi che hanno dato esito positivo delle Template Instances IPC sul campo. In parte, ciò è dovuto all'uso di criteri di corrispondenza wildcard per il 21° input durante i test e nelle Template Instance IPC.

Il 19 luglio 2024 sono state distribuite altre due Template Instances IPC. Una di queste ha introdotto un criterio di corrispondenza non wildcard per il 21° parametro di input. Queste nuove Template Instances hanno portato a una nuova versione del Channel File 291 che richiedeva al sensore di ispezionare il 21° parametro di input. Fino a quel momento, nessuna Template Instances IPC nelle versioni precedenti del channel aveva utilizzato il 21° campo del parametro di input. Il Content Validator ha valutato le nuove Template Instances, ma ha basato la sua valutazione sull'aspettativa che il Template Type IPC fosse stato fornito con 21 input.

I sensori che hanno ricevuto la nuova versione del Channel File 291 contenente i contenuti problematici sono stati esposti a un problema latente di lettura out of bounds nel Content Interpreter. Alla notifica IPC successiva da parte del sistema operativo, sono state valutate le nuove Template Instances IPC, specificando un confronto con il 21° valore di input. Il Content Interpreter prevedeva solo 20 valori. Pertanto, il tentativo di accedere al 21° valore ha prodotto una lettura di memoria out of bounds oltre l'array di dati di input e ha provocato un arresto anomalo del sistema.

In sintesi, è stata la combinazione di questi problemi a causare un crash di sistema: la mancata corrispondenza tra i 21 input convalidati dal Content Validator e i 20 forniti al Content Interpreter, il problema di lettura latente out of bounds nel Content Interpreter e la mancanza di un test specifico per i criteri di corrispondenza non wildcard nel 21° campo.

Sebbene quanto accaduto con Channel File 291 non possa più ripetersi, questo ha condotto CrowdStrike ad implementare miglioramenti dei processi e delle misure di mitigazione per garantire una maggiore resilienza.

RISULTANZE E MITIGAZIONI

1. Il numero di campi nel Template Type IPC non era stato convalidato in fase di compilazione del sensore

Risultanze**:** Al momento dell'incidente, il codice del sensore per il Template Type IPC descriveva 20 diverse fonti di input per l'utilizzo da parte della Template Instance. Ciò significa che, quando il sensore voleva prendere una decisione di rilevamento in base al Template Type IPC, il codice del sensore forniva 20 diverse fonti di input al Content Interpreter. Tuttavia, la definizione del Template Type IPC nel file Template Type Definitions affermava che i campi di input attesi fossero 21. Questa definizione ha portato a delle Template Instances nel Channel File 291, che dovevano funzionare su 21 input. La mancata corrispondenza non è stata rilevata durante lo sviluppo del Template Type IPC. I casi di test e i Rapid Response Content utilizzati per testare il Template Type IPC non sono incorsi nella condizione di errore durante lo sviluppo della funzionalità, né durante il test della versione 7.11 del sensore.

Misure di Mitigazione: convalida del numero di campi di input nel Template Type al momento della compilazione del sensore

Il 19 luglio 2024 è stata sviluppata per il Sensor Content Compiler una patch che convalida il numero di input forniti da un Template Type, la quale è entrata in produzione il 27 luglio 2024 come parte degli strumenti di compilazione interni di CrowdStrike. La patch del Sensor Content Compiler ha inoltre verificato che nessun altro Template Type, su nessuna piattaforma, forniva un numero errato di input.

2. Mancanza di un controllo dei limiti dell'array in fase di esecuzione per i campi di input del Content Interpreter nel Channel File 291

Risultanze**:** i Rapid Response Content per il Channel File 291 hanno istruito il Content Interpreter di leggere la 21° voce dell'array di puntatori di input. Tuttavia, il Template Type IPC genera solo 20 input. Di conseguenza, una volta consegnati i Rapid Response Content che utilizzavano un criterio di corrispondenza senza wildcard per il 21° input, il Content Interpreter ha eseguito una lettura out of bounds dell'array di input. Non si tratta di un problema di scrittura arbitraria della memoria ed è stato esaminato in modo indipendente.

Misure di Mitigazione: aggiunta di controlli Content Interpreter ai limiti dell'array di input in fase di esecuzione per i Content Interpreter nel Channel File 291

Il controllo dei limiti è stato integrato alla funzione Content Interpreter che ha recuperato le stringhe di input il 25 luglio 2024. Contemporaneamente, è stato aggiunto un ulteriore controllo volto a convalidare che la dimensione dell'array di input corrisponda al numero di input previsti dal Rapid Response Content. Queste correzioni sono state trasferite su tutte le versioni del sensore Windows dalla 7.11 in poi tramite un rilascio di hotfix del software del sensore. Questa versione sarà disponibile a livello generale entro il 9 agosto 2024.

L'aggiunta del controllo dei limiti impedisce al Content Interpreter di eseguire un accesso out of bounds all'array di input e di mandare in crash il sistema. Il controllo aggiuntivo rappresenta un ulteriore livello di convalida del tempo di esecuzione secondo cui la dimensione dell'array di input corrisponde al numero di input previsto dal Rapid Response Content.

Abbiamo completato il fuzz test del Template Type Channel 291 e lo stiamo estendendo a ulteriori gestori di Rapid Response Content nel sensore.

Misure di Mitigazione: correzione del numero di input forniti dal Template Type IPC

Il codice del sensore che definisce il Template Type IPC è stato aggiornato per fornire il numero corretto di input (21). Questa correzione è stata trasferita su tutte le versioni del sensore Windows dalla 7.11 in poi tramite un rilascio di hotfix del software del sensore. Questa versione sarà disponibile a livello generale entro il 9 agosto 2024.

3. Il test del Template Type deve riguardare una varietà più ampia di criteri di corrispondenza

Risultanze: durante lo sviluppo del Template Type IPC sono stati eseguiti sia test manuali che automatizzati, incentrati sulla convalida funzionale del Template Type, incluso il flusso corretto di dati rilevanti per la sicurezza attraverso di esso, e sulla valutazione dei dati per generare avvisi di rilevamento appropriati in base ai criteri creati nei casi di test di sviluppo.

I test automatizzati hanno sfruttato strumenti interni ed esterni per creare i dati rilevanti per la sicurezza necessari per esercitare il Template Type IPC in tutte le versioni di Windows supportate in un ampio sottoinsieme dei casi d'uso operativi previsti. Per i test automatizzati, è stato selezionato un set statico di 12 casi di test, rappresentativo di aspettative operative più ampie e necessario per convalidare la creazione di avvisi di telemetria e rilevamento. Parte di questo test includeva la definizione di un channel file da usare nei casi di test. La selezione dei dati nel channel file è stata effettuata manualmente e includeva un criterio di corrispondenza regex wildcard nel 21° campo per tutte le Template Instance, il che significa

che l'esecuzione di questi test durante le build di sviluppo e di rilascio non ha rilevato la lettura out of bounds latente nel Content Interpreter quando sono stati forniti 20 invece che 21 input.

Misure di Mitigazione: copertura dei test aumentata durante lo sviluppo del Template Type

Per confermare che stiamo convalidando tutti i campi di ogni Template Type, abbiamo creato test automatici che lavorano con criteri di corrispondenza non wildcard per ogni campo. Questo passaggio è stato eseguito per tutti i Template Types esistenti e sarà obbligatorio per tutti i Template Types futuri. Inoltre, tutti i futuri Template Types includeranno casi di test con scenari aggiuntivi che riflettano meglio l'utilizzo in produzione.

4. Il Content Validator conteneva un errore logico

Risultanze: Il Content Validator ha valutato le nuove Template Instances. Tuttavia, la valutazione si basava sull'aspettativa che il Template Type IPC sarebbe stato fornito con 21 input. Ciò ha comportato l'invio della Template Instance problematica al Content Interpreter.

Misure di Mitigazione: creazione di controlli aggiuntivi nel Content Validator

Il Content Validator è stato modificato per aggiungere nuovi controlli volti a garantire che i contenuti nelle Template Instance non includano criteri che corrispondono a più campi di quelli forniti come input al Content Interpreter. Questa correzione verrà rilasciata in produzione entro il 19 agosto 2024.

Misure di Mitigazione: impedire la creazione di Channel File 291 problematici

Il Content Validator è stato modificato per consentire solo i criteri di corrispondenza wildcard nel 21° campo, il che impedisce l'accesso out of bounds nei sensori che forniscono solo 20 input.

5. La convalida della Template Instance deve essere espansa per includere i test all'interno del Content Interpreter

Risultanze**:** i nuovi Template Type che vengono rilasciati vengono sottoposti a stress test su diversi aspetti, quali l'utilizzo delle risorse, l'impatto sulle prestazioni del sistema e il volume dei rilevamenti. Per molti Template Types, fra cui il Template Type IPC, si usa una Template Instance specifica per sottoporlo a stress test e confrontarlo con tutti i valori possibili dei campi dati associati, al fine di individuare interazioni avverse del sistema.

Uno stress test del Template Type IPC con una Template Instance di test è stato eseguito nel nostro ambiente di test, che consiste in una varietà di sistemi operativi e workload. Il Template Type IPC ha superato lo stress test ed è stato convalidato per l'uso, mentre una Template Instance è stata rilasciata in produzione come parte di un aggiornamento dei Rapid Response Content.

Tuttavia, la Template Instance testata da Content Validator non ha individuato che il numero non corrispondente di input avrebbe causato un arresto anomalo del sistema se fornito al Content Interpreter dal Template Type IPC.

Misure di Mitigazione: aggiornamento delle procedure di test di Content Configuration System

Il Content Configuration System è stato aggiornato con nuove procedure di test per garantire che ogni nuova Template Instance venga testata, indipendentemente dal fatto che la Template Instance iniziale venga testata insieme al Template Type al momento della creazione. In questo modo, le Template Instance possono essere testate ulteriormente prima dell'utilizzo in produzione.

6. Le Template Instance necessitano di avere un deployment graduale.

Risultanze**:** Il deployment di ogni Template Instance deve avere un rollout graduale.

Misure di Mitigazione: il Content Configuration System è stato aggiornato con ulteriori livelli di configurazione e controlli di accettazione

Il deployment graduale mitiga l'impatto nel caso in cui una nuova Template Instance dovesse causare errori come crash di sistema, picchi di volume di rilevamento di falsi positivi o problemi di prestazioni. Le nuove Template Instance che hanno superato il canary test dovranno essere successivamente passate a fasi di deployment più ampie o ritirate se vengono rilevati problemi. Ogni fase è progettata per identificare e mitigare i potenziali problemi prima di una configurazione con un raggio più ampio. Il passaggio di un test alla fase successiva è seguito da un ulteriore periodo di integrazione, in cui la telemetria viene raccolta per determinare l'impatto complessivo della Template Instance sull'endpoint.

Misure di Mitigazione: dare al cliente il controllo sul deployment degli aggiornamenti dei Rapid Response Content

La piattaforma Falcon è stata aggiornata per fornire ai clienti un maggiore controllo sul rilascio di Rapid Response Content. I clienti possono scegliere dove e quando configurare il deployment di Rapid Response Content. Continueremo a migliorare questa capacità per fornire un controllo ancora più granulare sul deploymentdi Rapid Response Content e

dettagli sugli aggiornamenti dei contenuti tramite release notes alle quali i clienti possono iscriversi.

CONTROLLO INDIPENDENTE DI TERZE PARTI

CrowdStrike ha incaricato due fornitori di sicurezza software di terze parti indipendenti perché conducano un'ulteriore revisione del codice del sensore Falcon sia dal punto di vista della sicurezza che per le garanzie rispetto al controllo qualità. Stiamo inoltre conducendo una revisione indipendente del processo di qualità end-to-end, dallo sviluppo fino alla fase di deployment. Entrambi i fornitori hanno iniziato il loro controllo, concentrandosi immediatamente sul codice e sul processo interessati il 19 luglio.

DETTAGLI TECNICI

Contesto e terminologia

CrowdStrike fornisce aggiornamenti di configurazione dei contenuti di sicurezza ai nostri sensori in due modi: tramite il Sensor Content, distribuito direttamente con il nostro sensore, e il Rapid Response Content, progettato per rispondere tempestivamente all'evolversi delle minacce.

L'elaborazione dei Rapid Response Content basati su regex sul sensore prevede diversi componenti:

- **Content Interpreter:** parte del codice C++ del sensore che può testare le stringhe di input rispetto alle espressioni regolari.
- **Template Type:** contengono campi predefiniti che i threat detection engineers possono sfruttare nei Rapid Response Content. I Template Type sono parte del codice dell'agente con esso vengono compilati.
- **File Template Type Definitions:** definisce i parametri di ciascun Template Type. Le definizioni presenti in questo file includono informazioni su quale Channel File fornirà i Rapid Response Content per ogni Template Type, quanti input il Template Type deve utilizzare e che tipo di dati è richiesto per ciascun input.
- **Sensor Content:** determina come combinare i dati rilevanti per la sicurezza con i Rapid Response Content per prendere determinate decisioni di rilevamento. Il Sensor Content include modelli di intelligenza artificiale e machine learning sul sensore, nonché i Template Types. Viene compilato come parte del rilascio del sensore.
- **Template Instances:** criteri di corrispondenza sviluppati dai detection engineers. Le Template Instances sono costituite da contenuto regex destinato all'uso con un

Template Type specifico. Le Template Instances identificano dati specifici da utilizzare nelle operazioni di sicurezza. Le Template Instances vengono definite utilizzando un'interfaccia utente guidata dal file Template Type Definitions.

- **Rapid Response Content:** sono costituiti da più Template Instances raggruppate congiuntamente. I Rapid Response Content vengono forniti attraverso il channel file.
- **Content Validator**: controlla la validità dei channel file rispetto alla loro definizione nel file Template Type Definitions.
- **Content Configuration System**: viene usato per creare le Template Instances, che vengono convalidate e implementate nel sensore tramite un meccanismo chiamato **Channel File**.

Utilizzo del driver del kernel in un prodotto di sicurezza

Come sottolineato da David Weston sul blog di Microsoft Security, i prodotti di sicurezza dell'ecosistema Windows, compreso il sensore Falcon, sfruttano comunemente i driver del kernel come componenti fondamentali per garantire un migliore livello di sicurezza.

La presenza nel kernel offre la necessaria visibilità sulle attività rilevanti per la sicurezza a livello di sistema, come la creazione di processi e thread o la scrittura, l'eliminazione e la modifica di file su disco. Le interfacce esposte dal kernel consentono ai driver di CrowdStrike di imporre controlli critici per un prodotto di sicurezza, come la prevenzione in tempo reale dei processi dannosi o il blocco dei file malware scritti su disco.

Il driver del kernel di CrowdStrike viene caricato in una fase iniziale dell'avvio del sistema, per consentire al sensore di osservare e difendere il sistema contro i malware che si avviano prima del lancio dei processi in modalità utente.

Fornire contenuti di sicurezza aggiornati (come i Rapid Response Content di CrowdStrike) a queste funzionalità del kernel consente al sensore di difendere i sistemi da un panorama di minacce in continua evoluzione, senza apportare modifiche dirette al codice del kernel. I Rapid Response Content sono dati di configurazione, non si tratta né di codici né di driver del kernel.

CrowdStrike certifica ogni nuovo rilascio di sensori Windows tramite il programma Windows Hardware Quality Labs (WHQL), che include prove approfondite per tutti i test richiesti nel Windows Hardware Lab Kit (HLK) e nel Windows Hardware Certification Kit (HCK) di Microsoft. Il processo di certificazione WHQL è l'ultimo di una serie completa di test interni come test funzionali, di longevità, fault injection stress test, fuzzing e performance test. Durante i test richiesti per il programma WHQL, i sensori usano le versioni più recenti dei channel file al momento della certificazione.

Poiché le nuove versioni di Windows introducono il supporto per eseguire più funzioni di sicurezza in modalità utente, CrowdStrike aggiorna il suo agente per utilizzare questo supporto. Per l'ecosistema Windows c'è ancora molto lavoro da fare per supportare un prodotto di sicurezza robusto che non sia basato su un driver del kernel per almeno alcune delle sue funzionalità. Ci impegniamo a lavorare direttamente con Microsoft su base continuativa man mano che Windows continuerà ad aggiungere ulteriore supporto per le esigenze dei prodotti di sicurezza in modalità utente.

Analisi del crash dump

Per illustrare come le nuove Template Instances nel Channel File 291 abbiano portato ad un arresto anomalo del sistema, di seguito verrà esaminato brevemente un crash dump anomalo del kernel, proveniente da un sistema interessato dal contenuto problematico. Si tratta di un ampliamento dell'analisi degli arresti anomali condivisa da David Weston nel blog Microsoft Security.

Aprendo il crash dump nel debugger del kernel di Windows e utilizzando il comando standard !analyze -v per ottenere un rapido riepilogo, vediamo che si è verificato un errore di allocazione della memoria (noto anche come "access violation"). *(Nota: per brevità, verranno omessi i dettagli del debug non correlati e verrà analizzato un crash dump di esempio. Ne esistono di diversi tipi, a seconda dei dettagli dello stato della macchina).*

```
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
```

Questo comando di valutazione automatizzato identifica `csagent.sys` come driver che esegue l'accesso out of bounds alla memoria. `csagent.sys` è il driver filtro del file system di CrowdStrike, un tipo di driver del kernel che si registra con i componenti del sistema operativo Windows per ricevere in tempo reale notifiche rilevanti per la sicurezza riguardo alle attività del sistema.

Tra le notifiche a cui si registra il driver di CrowdStrike ce n'è una per la creazione delle named pipes. Quando il driver riceve una notifica named pipe, i dati vengono combinati con altre informazioni contestuali sul sistema. Questi dati combinati vengono presentati per la valutazione rispetto alle Template Instance trasmesse nel Channel File 291.

Per osservare più da vicino questo processo, analizziamo lo stato del registro nel punto della lettura della memoria out of bounds ripristinando il trap frame e disassemblando le istruzioni precedenti per orientarci. *(Nota: questo elenco disassembly è stato modificato rispetto all'output del debugger standard per annotare il codice con nomi simbolici illustrativi.)*

```
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]
```

Prima di questo frammento di codice, i dati di contesto della notifica named pipe venivano preparati per il Template Type IPC come matrice di 20 puntatori di input, ognuno dei quali punta a una struttura di stringa che contiene un indirizzo buffer e un valore di dimensione. Questo frammento di codice intende selezionare uno degli input per restituire l'indirizzo e le dimensioni del buffer, secondo un indice specificato dal Channel File 291.

Quando inseriamo questo codice, l'indirizzo della matrice di puntatori a 20 input è contenuto nel registro rax, e il registro r11 indica che l'input da recuperare è all'indice 0x14, ossia il 21° elemento.

Esaminando la matrice di input, troviamo infatti un array di 20 puntatori a strutture di stringhe di input, seguito da un ventunesimo valore che *non* punta a una memoria valida:

```
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
```

Dopo aver letto questo puntatore non valido nel registro r8, il flusso di controllo nel frammento precedente esegue il primo salto all'indirizzo `fffff802`0ebc14e8`, esegue un controllo del puntatore NULL e quindi tenta una lettura attraverso il puntatore non valido, dando come risultato una lettura out of bounds e un successivo bugcheck.

RISORSE AGGIUNTIVE

Hub di remediation e indicazioni: aggiornamento dei contenuti Falcon per host Windows
Aggiornamento dei contenuti Falcon per host Windows
Hub di Remediation — Glossario dei termini

Il presente documento è una traduzione della seguente versione inglese https://www.crowdstrike.com/wp-content/uploads/2024/08/Channel-File-291-Incident-Root-Cause-Analysis-08.06.2024.pdf. La versione tradotta è riportata solo per agevolare la consultazione e per comodità dell'utente. In caso di discrepanze o ambiguità, la versione inglese prevarrà e avrà la precedenza in ogni caso.