Externe technische Ursachenanalyse – Channel File 291

EINFÜHRUNG

Dieser Bericht erläutert die Informationen, die wir bereits in unserem Preliminary Post Incident Review geteilt haben, und vertieft die Ergebnisse, Maßnahmen, technischen Details und die Ursachenanalyse des Vorfalls. Mit Stand vom 29. Juli um 17 Uhr PT sind im Wochenvergleich ca. 99 % der Windows-Sensoren online, verglichen mit dem Status vor dem Content-Update. Üblicherweise beobachten wir eine Abweichung von ca. 1 % von Woche zu Woche bei den Sensorverbindungen.

Zum besseren Verständnis verwenden wir in dieser RCA verallgemeinerte Terminologien zur Beschreibung der CrowdStrike Falcon-Plattform. Die Terminologie in anderen Dokumentationen kann daher spezifischer und technischer sein.

WAS PASSIERT IST

Der CrowdStrike Falcon-Sensor bietet leistungsstarke sensorinterne KI- und Machine-Learning-Modelle, um Kundensysteme durch Identifizierung und Beseitigung der neuesten fortschrittlichen Bedrohungen zu schützen. Diese Modelle werden durch Erkenntnisse aus der neuesten Bedrohungstelemetrie des Sensors sowie menschlicher Intelligenz von Falcon Adversary OverWatch, Falcon Complete und Threat-Detection-Ingenieuren von CrowdStrike auf dem neuesten Stand gehalten und gestärkt. Diese umfangreichen Sicherheitstelemetriedaten werden zunächst von den einzelnen Sensoren gefiltert und in einem lokalen Graphspeicher aggregiert.

Jeder Sensor korreliert in einem fortlaufenden Verfeinerungsprozess den Kontext aus seinem lokalen Graphspeicher mit der Live-Systemaktivität zu Verhaltensweisen und Angriffsindikatoren (IOAs). Dieser Verfeinerungsprozess beinhaltet eine Sensor Detection Engine, die integrierten Sensor-Content mit Rapid Response Content aus der Cloud kombiniert. Rapid Response Content wird verwendet, um Telemetriedaten zu sammeln, Indikatoren für das Verhalten von Angreifern zu identifizieren und neuartige Erkennungs- und Schutzmaßnahmen auf dem Sensor zu verbessern, ohne dass der Sensorcode geändert werden muss. Rapid Response Content ist eine verhaltensbasierte Heuristik, die von den KI-Präventions- und Erkennungsfunktionen auf dem Sensor von CrowdStrike getrennt ist und sich davon unterscheidet.

Rapid Response Content wird über Channel Files bereitgestellt und vom Content Interpreter des Sensors mithilfe einer auf regulären Ausdrücken (Regex) basierenden Engine interpretiert. Jedes Rapid Response Content Channel File ist mit einem bestimmten Template-Typ verknüpft, der in einem Sensor-Release integriert ist. Der Template-Typ stellt

dem Content Interpreter Aktivitätsdaten und Graph-Kontext für den Abgleich mit dem Rapid Response Content zur Verfügung.

Mit der Veröffentlichung der Sensorversion 7.11 im Februar 2024 hat CrowdStrike einen neuen Template-Typ eingeführt, um die Visibilität von neuartigen Angriffstechniken und deren Erkennungzu ermöglichen. Diese Angriffstechniken missbrauchen Named Pipes und andere Mechanismen der Windows-Interprozesskommunikation („IPC“). Wie im PIR beschrieben, wurde der neue IPC-Template-Typ gemäß unseren standardmäßigen Entwicklungsprozessen für Sensor-Content entwickelt und getestet und in den Sensor integriert, um ihn für den Einsatz in der Praxis vorzubereiten. IPC-Template-Instanzen werden als Rapid Response Content über ein entsprechendes Channel File mit der Nummer 291 an Sensoren übermittelt.

Der neue IPC-Template-Typ definierte 21 Eingabeparameterfelder, aber der Integrationscode, der den Content Interpreter mit den Template-Instanzen der Channel File 291 aufrief, lieferte nur 20 Eingabewerte zum Abgleich. Diese Nichtübereinstimmung der Parameteranzahl entging mehreren Ebenen von Build-Validierung und -Testing, da sie während des Testprozesses des Sensor-Releases, des Stresstests des Template-Typs (unter Verwendung einer Template-Instanz) oder den ersten erfolgreichen Bereitstellungen von IPC-Template-Instanzen in der Praxis nicht entdeckt wurde. Das lag zum Teil daran, dass für den 21. Eingabewert während der Tests und in den ersten IPC-Template-Instanzen Wildcard-Matching-Kriterien verwendet wurden.

Am 19. Juli 2024 wurden zwei weitere IPC-Template-Instanzen bereitgestellt. Eine davon führte ein Nicht-Wildcard-Matching-Kriterium für den 21. Eingabeparameter ein. Diese neuen Template-Instanzen führten zu einer neuen Version von Channel File 291, bei der der Sensor nun den 21. Eingabeparameter überprüfen musste. Bevor dieses Channel File an Sensoren übermittelt wurde, hatten keine IPC-Template-Instanzen in früheren Channel-Versionen das 21. Eingabeparameterfeld benutzt. Der Content Validator evaluierte die neuen Template-Instanzen, ging dabei aber davon aus, dass der IPC-Template-Typ mit 21 Eingabewerte bereitgestellt werden würde.

Bei Sensoren, die die neue Version des Channel Files 291 mit dem problematischen Inhalt erhalten haben, ist im Content Interpreter ein latenter unzulässiger Speicherzugriff aufgetreten. Bei der nächsten IPC-Benachrichtigung des Betriebssystems wurden die neuen IPC-Template-Instanzen evaluiert, wobei ein Vergleich mit dem 21. Eingabewert spezifiziert wurde. Der Content Interpreter erwartete nur 20 Werte. Daher führte der Versuch, auf den 21. Wert zuzugreifen, zu einem unzulässigen Speicherlesevorgang über das Ende des Eingabedatenbereichs hinaus und verursachte einen Systemabsturz.

Zusammenfassend lässt sich sagen, dass das Zusammentreffen dieser Probleme zu einem Systemabsturz führte: die Diskrepanz zwischen den 21 Eingabewerten, die vom Content

Validator validiert wurden, und den 20 Eingabewerten, die dem Content Interpreter zur Verfügung gestellt wurden, das Problem mit dem latenten unzulässigen Speicherzugriff im Content Interpreter und das Fehlen eines spezifischen Tests für Non-Wildcard-Matching-Kriterien im 21. Feld. Während dieses Szenario mit Channel File 291 nicht erneut auftreten kann, bietet es auch Anhaltspunkte für Verbesserungen an den Prozessen und Wiederherstellungsmaßnahmen, die CrowdStrike einführt, um die Resilienz weiter zu verbessern.

Erkenntnisse und Maßnahmen

1. Die Anzahl der Felder im IPC-Template-Typ wurde beim Kompilieren des Sensors nicht validiert

Erkenntnisse: Zum Zeitpunkt des Vorfalls beschrieb der Sensorcode für den IPC-Template-Typ 20 verschiedene Eingabequellen für die Verwendung durch die Template-Instanz. Das heißt, wenn der Sensor eine Erkennungsentscheidung auf der Grundlage des IPC-Template-Typs treffen wollte, lieferte der Sensorcode 20 verschiedene Eingabequellen an den Content Interpreter. In der Definition des IPC-Template-Typs in der Template-Typ-Definitionsdatei wurde jedoch angegeben, dass 21 Eingabefelder erwartet werden. Diese Definition führte zu Template-Instanzen in Channel File 291, die mit 21 Eingangswerten arbeiten sollten. Diese Unstimmigkeit wurde bei der Entwicklung des IPC-Template-Typs nicht erkannt. Die Testfälle und der Rapid Response Content, die zum Testen des IPC-Template-Typs verwendet wurden, lösten weder bei der Entwicklung der Funktion noch beim Testen der Sensorversion 7.11 einen Fehler aus.

Maßnahme: Validierung der Anzahl der Eingabefelder im Template-Typ beim Kompilieren des Sensors.

Ein Patch für den Sensor Content Compiler, der die Anzahl der von einem Template-Typ bereitgestellten Eingabewerte überprüft, wurde am 19. Juli 2024 entwickelt und ging am 27. Juli 2024 als Teil des internen Build Toolings von CrowdStrike in Produktion. Der Patch für den Sensor Content Compiler stellt außerdem sicher, dass keine anderen Template-Typen auf jeglicher Plattform eine falsche Anzahl von Eingabewerten liefern.

2. Eine zur Laufzeit stattfindende Überprüfung von Array-Grenzen für Eingabewerte im Content Interpreter fehlte im Channel File 291.

Erkenntnisse: Der Rapid Response Content für das Channel File 291 wies den Content Interpreter an, den 21. Eintrag des Eingabe-Pointer-Arrays auszulesen. Der IPC-Template-

Typ generiert jedoch nur 20 Eingabewerte. Infolgedessen führte der Content Interpreter nach der Bereitstellung von Rapid Response Content, die ein Nicht-Wildcard-Matching-Kriterium für den 21. Eingabewert verwendeten, einen unzulässigen Lesevorgang des Eingabearrays durch. Dies ist kein willkürlicherSchreibzugriff auf den Speicher.Dies wurde unabhängig überprüft.

Maßnahme: Hinzufügen einer während der Laufzeit stattfindenden Überprüfung von Eingabe-Array-Grenzen zum Content Interpreter für Rapid Response Content im Channel File 291.

Die Überprüfung von Array-Grenzen wurde der Content Interpreter-Funktion, die Eingabe-Strings abruft, am 25. Juli 2024 hinzugefügt. Eine zusätzliche Überprüfung, ob die Größe des Eingabe-Arrays mit der Anzahl der vom Rapid Response Content erwarteten Eingabewerte übereinstimmt, wurde zeitgleich hinzugefügt. Diese Korrekturen werden über ein Hotfix-Release der Sensorsoftware auf alle Windows Sensor Versionen ab Version 7.11 und höher angewandt. Diese Version wird bis 9. August 2024 allgemein verfügbar gemacht.

Die zusätzliche Überprüfung der Array-Grenzen verhindert, dass der Content Interpreter einen unzulässigen Zugriff auf das Eingabe-Array durchführt und einen Systemabsturz verursacht. Die zusätzliche Überprüfung fügt eine weitereValidierungsebene während der Laufzeit hinzu, die prüft, ob die Größe des Eingabearrays der Anzahl der Eingabewerte entspricht, die vom Rapid Response Content erwartet werden.

Wir haben das Fuzz-Testing des Channel 291 Template-Typs abgeschlossen und erweitern es auf zusätzliche Rapid Response Content-Handler im Sensor.

Maßnahme: Korrektur der Anzahl der Eingabewerte, die vom IPC-Template-Typ bereitgestellt werden.

Der Sensorcode, der den IPC-Template-Typ definiert, wurde aktualisiert, um die richtige Anzahl an Eingabewerten (21) bereitzustellen. Dieser Fix wird über ein Hotfix-Release der Sensorsoftware für alle Windows Sensor Versionen ab Version 7.11 und höher angewandt. Diese Version wird bis 9. August 2024 allgemein verfügbar gemacht.

3. Die Template-Typ-Prüfung soll eine größere Vielfalt an Matching-Kriterien abdecken

Erkenntnisse: Bei der Entwicklung des IPC-Template-Typs wurden sowohl manuelle als auch automatisierte Tests durchgeführt. Diese Tests konzentrierten sich auf die funktionale Validierung des Template-Typs, einschließlich des korrekten Durchlaufs sicherheitsrelevanter Daten durch diesen sowie die Auswertung dieser Daten, um geeignete Erkennungs-Alerts auf der Grundlage von Kriterien zu generieren, die in Entwicklungstestfällen erstellt wurden.

Beim automatisierten Testen wurden interne und externe Tools genutzt, um die erforderlichen sicherheitsrelevanten Daten zu erstellen, die zur Nutzungdes IPC-Template-Typs unter allen unterstützten Windows-Versionen innerhalb einer breiten Teilmenge der erwarteten betrieblichen Anwendungsszenarien erforderlich sind. Für die automatisierten Tests wurde ein statischer Satz von 12 Testfällen ausgewählt, der die allgemeinen Erwartungen an den Betrieb repräsentiert und die Erstellung von Telemetrie- und Erkennungs-Alerts validiert. Teil dieser Tests war das Definieren eines Channel Files zur Verwendung in den Testfällen. Die Auswahl der Daten in dem Channel File erfolgte manuell und umfasste für alle Template-Instanzen ein Regex-Wildcard-Matching-Kriterium im 21. Feld. Das bedeutet, dass bei der Ausführung dieser Tests während der Entwicklung und der Release-Builds das latente unzulässige Auslesen im Content Interpreter nicht aufgedeckt wurde, wenn 20 statt 21 Eingabewerte bereitgestellt wurden.

Maßnahme: Erhöhte Testabdeckung während der Entwicklung von Template-Typen

Um sicherzustellen, dass wir alle Felder in jedem Template-Typ überprüfen, wurden automatisierte Tests erstellt, die jedes Feld mit Non-Wildcard-Matching-Kriterien testen. Dieser Schritt wurde für alle vorhandenen Template-Typen durchgeführt und ist für alle zukünftigen Template-Typen erforderlich. Darüber hinaus enthalten alle zukünftigen Template-Typen Testfälle mit zusätzlichen Szenarien, die die Nutzung in Produktionsumgebungen besser widerspiegeln.

4. Der Content Validator enthielt einen Logikfehler

Erkenntnisse: Der Content Validator hat die neuen Template-Instanzen evaluiert. Er stützte seine Bewertung jedoch auf die Erwartung, dass der IPC-Template-Typ mit 21 Eingabewerten bereitgestellt werden würde. Dies führte dazu, dass die problematische Template-Instanz an den Content Interpreter gesendet wurde.

Maßnahme: Erstellung zusätzlicher Prüfungen im Content Validator

Der Content Validator wird geändert, um neue Prüfungen hinzuzufügen und so sicherzustellen, dass der Inhalt in Template-Instanzen keine Matching-Kriterien enthält, die mit mehr Feldern übereinstimmen, als dem Content Interpreter als Eingabe bereitgestellt werden. Diese Fehlerbehebung wird bis zum 19. August 2024 veröffentlicht.

Maßnahme: Die Erstellung problematischer Channel 291 Files verhindern

Der Content Validator wurde geändert, sodass Wildcard-Matching-Kriterien nur im 21. Feld zulässig sind. Dadurch wird unzulässiger Zugriff in den Sensoren verhindert, die nur 20 Eingabewerte bereitstellen.

5. Die Validierung von Template-Instanzen soll auf die Prüfung innerhalb des Content Interpreters ausgeweitet werden

Erkenntnisse: Neu veröffentlichte Template-Typen werden in Bezug auf viele Aspekte wie Ressourcenauslastung, Auswirkungen auf die Systemleistung und Erkennungsvolumen einem Stresstest unterzogen. Für viele Template-Typen, einschließlich des IPC-Template-Typs, wird eine spezifische Template-Instanz für den Stresstest des Template-Typs verwendet, indem alle möglichen Werte der zugehörigen Datenfelder abgeglichen werden, um schädlicheSysteminteraktionen zu identifizieren.

Ein Stresstest des IPC-Template-Typs mit einer Test-Template-Instanz wurde in unserer Testumgebung durchgeführt, die aus einer Vielzahl von Betriebssystemen und Workloads besteht. Der IPC-Template-Typ hat den Stresstest bestanden und wurde für den Einsatz validiert. Die Template-Instanz wurde im Rahmen eines Rapid Response Content Updates veröffentlicht.

Die vom Content Validator getestete Template-Instanz beachtete jedoch nicht, dass die nicht übereinstimmende Anzahl von Eingabewerten zu einem Systemabsturz führen würde, wenn sie dem Content Interpreter vom IPC-Template-Typ zur Verfügung gestellt wird.

Maßnahme: Testverfahren für das Content-Konfigurationssystem aktualisieren

Das Content-Konfigurationssystem wurde mit neuen Testverfahren aktualisiert, um sicherzustellen, dass jede neue Template-Instanz getestet wird, unabhängig davon, dass die ursprüngliche Template-Instanz bei der Erstellung mit dem Template-Typ getestet wird. Dadurch werden Template-Instanzen vor der Bereitstellung in Produktionsumgebungen zusätzlich getestet.

6. Template-Instanzen sollen stufenweise bereitgestellt werden

Erkenntnisse: Jede Template-Instanz soll in einem gestaffelten Rollout bereitgestellt werden.

Maßnahme: Das Content-Konfigurationssystem wurde mit zusätzlichen Bereitstellungsebenen und Akzeptanzprüfungen aktualisiert

Durch die stufenweise Bereitstellung werden die Auswirkungen geschwächt, wenn eine neue Template-Instanz Fehler wie Systemabstürze, sprunghaften Volumenanstieg bei False Positivesoder Leistungsprobleme verursacht. Neue Template-Instanzen, die die Canary-Tests bestanden haben, müssen stufenweise auf breitere Bereitstellungsringe erweitert werden, oder rückgängig gemacht werden, falls Probleme festgestellt werden. Jeder Ring ist so konzipiert, dass potenzielle Probleme vor einer breiteren Bereitstellung identifiziert und

behoben werden. Auf die Erweiterung einer Template-Instanz auf den nächsten Ring folgt eine zusätzliche Konsolidierungsperiode, in der Telemetrie erfasst wird, um die Gesamtauswirkung der Template-Instanz auf dem Endgerät zu ermitteln.

Maßnahme: Kunden-Kontrolle über die Bereitstellung von Rapid Response Content Updates

Die Falcon-Plattform wurde aktualisiert, um den Kunden eine bessere Kontrolle über die Bereitstellung von Rapid Response Content zu ermöglichen. Kunden können wählen, wo und wann die Rapid Response Content Updates bereitgestellt werden. Wir arbeiten weiter an der Verbesserung dieser Funktion, um eine granularere Kontrolle über die Bereitstellung von Rapid Response Content zu ermöglichen. Außerdem werden Details zu Content Updates in den Versionshinweisen veröffentlicht, die Kunden abonnieren können.

Überprüfung durch unabhängige Dritte

CrowdStrike hat zwei unabhängige Drittanbieter von Softwaresicherheitslösungen damit beauftragt, den Falcon-Sensorcode hinsichtlich Sicherheit und Qualitätssicherung weiter zu prüfen. Darüber hinaus führen wir eine unabhängige Überprüfung des gesamten Qualitätsprozesses von der Entwicklung bis zur Bereitstellung durch. Beide Anbieter haben die Überprüfungen begonnen, mit dem Fokus auf den vom 19. Juli betroffenen Code und Prozess.

Technische Details

Hintergrund und Terminologie

CrowdStrike stellt Konfigurationsupdates von Sicherheitsinhalten für unsere Sensoren auf zweierlei Weise bereit: Sensor-Content, der direkt mit unserem Sensor ausgeliefert wird, und Rapid Response Content, der entwickelt wird, um möglichst schnell auf die sich verändernde Bedrohungssituation zu reagieren.

Die Verarbeitung von Regex-basiertem Rapid Response Content auf dem Sensor umfasst mehrere Komponenten:

- **Content Interpreter:** Teil des C++-Codes des Sensors, mit dem Eingabe-Strings anhand von regulären Ausdrücken (Regex) getestet werden können.
- **Template-Typen:** Enthalten vordefinierte Felder, die Threat-Detection-Ingenieure in Rapid Response Content nutzen können. Template-Typen werden in Code ausgedrückt und während der Build-Time in den Sensor kompiliert.

- **Template-Typ-Definitionsdatei:** Definiert die Parameter der einzelnen Template-Typen. Die Definitionen in dieser Datei enthalten Informationen darüber, welches Channel File den Rapid Response Content für jeden Template-Typ liefert, wie viele Eingabewerte der Template-Typ verwenden soll und welche Art von Daten für jedes Eingabefeld erforderlich sind.
- **Sensor-Content:** Legt fest, wie sicherheitsrelevante Daten mit Rapid Response Content kombiniert werden, um bestimmte Erkennungsentscheidungen zu treffen. Sensor-Content umfasst sensorinterne KI- und Machine-Learning-Modelle sowie Template-Typen. Er wird im Rahmen des Sensor-Releases kompiliert.
- **Template-Instanzen:** Matching-Kriterien, die von Detection-Ingenieuren entwickelt werden. Template-Instanzen bestehen aus Regex-Inhalten, die für die Verwendung mit einem bestimmten Template-Typ vorgesehen sind. Template-Instanzen identifizieren bestimmte Daten für die Verwendung in Sicherheitsuntersuchungsvorgängen. Template-Instanzen werden mithilfe einer Benutzeroberfläche definiert, die von der Template-Typ-Definitionsdatei gesteuert wird.
- **Rapid Response Content:** Besteht aus mehreren Template-Instanzen, die miteinander gebündelt sind. Rapid Response Content wird über Channel Files bereitgestellt.
- **Content Validator**: Überprüft die Gültigkeit von Channel Files anhand ihrer Definition in der Template-Typ-Definitionsdatei.
- **Content-Konfigurationssystem**: Wird verwendet, um Template-Instanzen zu erstellen, die über einen Mechanismus namens **Channel Files** validiert und auf dem Sensor bereitgestellt werden.

Verwendung von Kernel-Treibern in einem Sicherheitsprodukt

Wie David Weston im Microsoft Security Blog beschreibt, nutzen die Sicherheitsprodukte im Windows-Ökosystem, darunter auch der Falcon-Sensor, in der Regel Kernel-Treiber als Kernkomponenten eines robusten Sicherheitsprodukts.

Die Präsenz im Kernel bietet einen umfassenden Einblick in systemweite sicherheitsrelevante Aktivitäten, wie das Erstellen von Prozessen und Threads oder das Schreiben, Löschen und Ändern von Dateien auf Festplatte. Die vom Kernel offengelegten Schnittstellen ermöglichen es den CrowdStrike-Treibern, kritische Kontrollen für ein Sicherheitsprodukt durchzusetzen, z. B. Inline-Prävention von bösartigen Prozessen oder dass Malware-Dateien auf die Festplatte geschrieben werden.

Der Kernel-Treiber von CrowdStrike wird schon in einer frühen Phase des Systemstarts geladen, damit der Sensor Malware erkennen und abwehren kann, die vor dem Start von Prozessen im User-Modus gestartet wird.

Die Bereitstellung aktueller Sicherheitsinhalte (z. B. Rapid Response Content von CrowdStrike) für diese Kernelfunktionen ermöglicht es dem Sensor, Systeme vor einer sich schnell entwickelnden Bedrohungslandschaft zu schützen, ohne Änderungen am Kernelcode vornehmen zu müssen. Rapid Response Content besteht aus Konfigurationsdaten; es ist weder Code noch ein Kernel-Treiber.

CrowdStrike zertifiziert jeden neuen Windows-Sensor-Release durch das Windows Hardware Quality Labs (WHQL) Programm, das umfangreiche Überprüfungenmit allen erforderlichen Tests im Windows Hardware Lab Kit (HLK) und im Windows Hardware Certification Kit (HCK) von Microsoft umfasst. Der WHQL-Zertifizierungsprozess markiert das Ende eines umfassenden internen Testvorgangs, der Funktionstests, Langlebigkeitstests, Stresstests mit Fehlerinjektion, Fuzzing und Leistungstests. Während der Tests, die für das WHQL-Programm erforderlich sind, verwenden die Sensoren die zum Zeitpunkt der Zertifizierung neuesten Versionen der Channel Files.

Neue Windows-Versionen führen nach und nach Unterstützung für die Ausführung von zusätzlichenSicherheitsfunktionen im Userspace ein CrowdStrike aktualisiert seinen Agent stetig, um diese Unterstützung zu nutzen. Das Windows-Ökosystem muss sich noch stark weiterentwickeln, um robusteSicherheitsprodukte zu ermöglichen, diezumindest für einen Teil seiner Funktionen nicht auf einen Kernel-Treiber angewiesensind. Wir sind bestrebt, kontinuierlich direkt mit Microsoft zusammenzuarbeiten, um die sich ständig weiterentwickelnde Unterstützung für Sicherheitsprodukte im Userspace im Windows-Ökosystem zu nutzen.

Analyse des Absturzberichtes

Um zu veranschaulichen, wie die neuen Template-Instanzen im Channel File 291 zu einem Systemabsturz führten, untersuchen wir kurz einen Kernel-Absturzbericht von einem System, das von dem problematischen Inhalt betroffen war. Dies erweitert die Absturz-Analyse, die von David Weston im Microsoft Security-Blog geteilt wurde.

Wenn wir den Absturzbericht im Windows-Kernel-Debugger öffnen und den Standardbefehl „!analyze -v“ für eine schnelle Zusammenfassung verwenden, sehen wir, dass ein Speicherfehler (auch als „Zugriffsverletzung“ (access violation) bezeichnet) aufgetreten ist. *(Hinweis: Nicht damit zusammenhängende Debug-Details werden der Kürze halber weggelassen. Es wird ein repräsentativer Absturzbericht analysiert. Abhängig von den Details des Maschinenzustands gibt es Variationen des Berichtes.)*

```
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
```

Dieser automatisierte Triage-Befehl identifiziert `csagent.sys` als den Treiber, der den unzulässigen Speicherzugriff ausführt. `csagent.sys` ist der Dateisystemfilter-Treiber von CrowdStrike, eine Art Kernel-Treiber, der sich bei Komponenten des Windows-Betriebssystems registriert, um Benachrichtigungen über sicherheitsrelevante Systemaktivitäten in Echtzeit zu erhalten.

Zu den Benachrichtigungen, für die der CrowdStrike-Treiber registriert ist, gehört auch eine Benachrichtigung über die Erstellung von Named Pipes. Wenn der Treiber eine Named Pipe-Benachrichtigung erhält, werden diese Daten mit anderen Kontextinformationen über das System kombiniert. Diese kombinierten Daten werden zur Bewertung anhand der im Channel File 291 übermittelten Template-Instanzen vorgelegt.

Um diesen Prozess genauer zu betrachten,untersuchen wir den Registerzustand an der Stelle, an der der unzulässige Zugriff stattfindet, indem wir den Trap-Frame wiederherstellen und die vorherigen Anweisungen disassemblieren, um uns zu orientieren. *(Hinweis: Diese Disassemblierungsliste wurde gegenüber der Standardausgabe des Debuggers geändert, um den Code mit illustrativen Symbolnamen zu versehen.)*

```
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]
```

Vor diesem Codeabschnitt wurden die Kontextdaten aus der Named-Pipe-Benachrichtigung für den IPC-Template-Typ als Array von 20 Eingabezeigern vorbereitet, die jeweils zu einer String-Struktur weisen, die eine Buffer-Addresse und einen Größenwert enthält. Dieser Abschnitt wählt eine der Eingaben aus, um seine Buffer-Addresse und -größe, gemäß einem von Channel File 291 angegebenen Index, zurückzuerhalten.

Wenn wir diesen Code eingeben, wird die Adresse des Pointer-Arraysmit 20 Eingabewerten im Register rax gehalten, und das Register r11 gibt an, dass der abzurufende Eingabewert im Index 0x14 liegt, d. h. das 21. Element.

Wenn wir das Eingabe-Array untersuchen, finden wir tatsächlich ein Array von 20 Zeigern auf Eingabestring-Strukturen, gefolgt von einem 21. Wert, der *nicht* auf gültigen Speicher verweist:

```
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
```

Nach dem Einlesen dieses ungültigen Pointers in das Register r8 springt der Kontrollfluss im obigen Codeausschnitt zuerst zur Adresse `fffff802`0ebc14e8`, führt eine NULL-Pointer-Prüfung durch und versucht dann, den ungültigen Pointer zu lesen, was zu einem unzulässigen Lesevorgang und einem anschließenden Bugcheck führt.

ZUSÄTZLICHE RESSOURCEN

Verweise und Links zu zusätzlichen technischen Ressourcen
Remediation and Guidance Hub: Falcon Content Update für Windows-Hosts
Blog: Technische Details: Falcon Content Update für Windows-Hosts
Remediation Hub – Begriffsverzeichnis

Dieses Dokument ist eine Übersetzung der folgenden englischen Version https://www.crowdstrike.com/wp-content/uploads/2024/08/Channel-File-291-Incident-Root-Cause-Analysis-08.06.2024.pdf. Diese übersetzte Version dient ausschließlich der einfacheren Bezugnahme und Zweckmäßigkeit. Im Falle von Widersprüchen oder einer Unklarheit hat die englische Version stets Vorrang.