外部向けテクニカル根本原因分析—チャネルファイル 291

はじめに

本レポートでは、インシデント事後のプレレビュー（PIR）にて既に共有された情報をさらに詳しく説明し、インシデントの発見、緩和策、技術的な詳細、根本原因の分析について詳しく説明します。7 月 29 日午後 5 時（米国太平洋標準時）の時点で、前週比で、コンテンツ更新前と比較して、Windows センサーの約 99%がオンラインになっています。通常、センサーの接続には、前週比で約 1%のばらつきがあります。

この RCA 全体を通し、読みやすさを向上させるため、CrowdStrike Falcon プラットフォームを説明するために一般化された用語を使用しています。尚、本報告の日付はすべて米国時間における日付となります。本文書は、以下の英語（https://www.crowdstrike.com/wp-content/uploads/2024/08/Channel-File-291-Incident-Root-Cause-Analysis-08.06.2024.pdf）の翻訳です。本翻訳版は、参照を容易にし、便宜上の目的でのみ提供されています。矛盾や曖昧さが生じた場合は、常に英語版が優先されるものとします。

発生事象について

CrowdStrike Falcon センサーは、強力なオンセンサーAI と機械学習モデルを提供し、最新の高度な脅威を識別して修復することで顧客のシステムを保護します。これらのモデルは、センサーからの最新の脅威テレメトリと、Falcon Adversary OverWatch、Falcon Complete、クラウドストライク の脅威検知エンジニアによるヒューマンインテリジェンスからの学習によって最新の状態に保たれ、強化されます。この豊富なセキュリティテレメトリは、まずデータをフィルタリングし、各センサーでローカルのグラフストアに集約することから始まります。

各センサーは、継続的な洗練プロセスの中で、ローカルグラフストアからのコンテキストとライブのシステムアクティビティを関連付け、振る舞いと攻撃の痕跡（IOA）に変換します。この洗練プロセスには、組み込みのセンサーコンテンツとクラウドから配信されるラピッドレスポンスコンテンツを組み合わせたセンサー検知エンジンが含まれます。ラピッドレスポンスコンテンツは、センサーコードの変更を必要とすることなく、テレメトリの収集、攻撃者の振る舞いの兆候の特定、センサーの新たな検知と防御の強化に使用されます。ラピッドレスポンスコンテンツは、行動ヒューリスティックであり、クラウドストライクのオンセンサーAI の防御と検知の機能とは別個で区別されるものです。

ラピッドレスポンスコンテンツはチャネルファイルを通じて配信され、正規表現ベースのエンジンを使用してセンサーのコンテンツインタープリターによって解釈されます。各ラピッドレスポンスコンテンツチャネルファイルは、センサーリリースに組み込まれた特定のテンプレートタイプに関連付けられています。テンプレートタイプは、ラピッドレスポンスコンテンツと照合するためのアクティビティデータとグラフコンテキストをコンテンツインタープリターに提供します。

2024 年 2 月のセンサーバージョン 7.11 のリリースにより、クラウドストライクは新しいテンプレートタイプを導入しました。これにより、名前付きパイプやその他の Windows プロセス間通信（「IPC」）メカニズムを悪用する新しい攻撃手法を可視化して検知することが可能になりました。PIR で概説していますように、新しい IPC テンプレートタイプは、弊社の標準的なセンサーコンテンツ開発プロセスに従って開発・テストされ、フィールドでの使用に備えてセンサーに統合されました。IPC テンプレートインスタンスは、対応するチャネルファイル 291 番を介して、ラピッドレスポンスコンテンツとしてセンサーに配信されます。

新しい IPC テンプレートタイプは 21 の入力パラメーターフィールドを定義しましたが、チャネルファイル 291 のテンプレートインスタンスでコンテンツインタープリターを呼び出すインテグレーションコードでは、照合する入力値が 20 個しかありませんでした。このパラメーター数の不一致は、センサーのリリーステストプロセス、テンプレートタイプ（テストテンプレートインスタンスを使用）のストレステスト、また、フィールドでの IPC テンプレートインスタンスの最初の数回の正常な展開中には検出されなかったため、複数のレイヤーのビルド検証およびテストをすり抜けました。これは、テスト中および最初の IPC テンプレートインスタンスで、21 番目の入力にワイルドカードの一致基準を使用したことが一因です。

2024 年 7 月 19 日、2 つの IPC テンプレートインスタンスが追加で展開されました。そのうちの 1 つは、21 番目の入力パラメーターにワイルドカードでない一致基準を導入したものです。これらの新しいテンプレートインスタンスにより、チャネルファイル 291 の新しいバージョンは、センサーに 21 番目の入力パラメーターを検査することを要求するようになりました。このチャネルファイルがセンサーに配信されるまで、以前のチャネルバージョンの IPC テンプレートインスタンスでは 21 番目の入力パラメーターフィールドが使用されていませんでした。コンテンツバリデーターは新しいテンプレートインスタンスを評価しましたが、IPC テンプレートタイプには 21 の入力が提供されることを想定して評価しました。

問題のあるコンテンツを含むチャネルファイル 291 の新バージョンを受信したセンサーは、コンテンツインタープリターで潜在的な境界外読み取りの問題にさらされました。オペレー

ティングシステムからの次の IPC 通知で、新しい IPC テンプレートインスタンスが評価され、21 番目の入力値との比較が指定されました。コンテンツインタープリターは 20 個の値しか期待していませんでした。そのため、21 番目の値にアクセスしようとすると、入力データ配列の終端を超えて境界外のメモリが読み込まれ、システムがクラッシュしました。

要約すると、コンテンツバリデーターが検証する 21 個の入力と、コンテンツインタープリターに提供される 20 個の入力との間の不一致、コンテンツインタープリターに潜在していた境界外読み取りの問題、そして 21 番目のフィールドにおける非ワイルドカードマッチング基準に対する特定のテストの欠如という要因が複合的に作用し、結果としてシステムクラッシュが発生しました。このチャネルファイル 291 のシナリオは、現在再発することはありませんが、クラウドストライクがレジリエンスをさらに強化することを確実にするために展開しているプロセスの改善と緩和策にも反映されています。

調査結果と緩和策

1. IPC テンプレートタイプのフィールド数がセンサーのコンパイル時に検証されなかった

調査結果：インシデント当時、IPC テンプレートタイプのセンサーコードには、テンプレートインスタンスが使用する 20 種類の入力ソースが記載されていました。これは、センサーが IPC テンプレートタイプに基づいて検知判断を行う場合、センサーコードはコンテンツインタープリターに 20 種類の入力ソースを提供することを意味します。しかし、テンプレートタイプ定義ファイルの IPC テンプレートタイプの定義には、21 の入力フィールドが必要であると記載されていました。この定義の結果、チャネルファイル 291 のテンプレートインスタンスは、21 の入力で動作することを期待していました。この不一致は、IPC テンプレートタイプの開発中には検出されませんでした。IPC テンプレートタイプのテストに使用されたテストケースとラピッドレスポンスコンテンツは、機能の開発中やセンサー7.11 リリースのテスト中に障害を引き起こしませんでした。

緩和策：センサーのコンパイル時にテンプレートタイプの入力フィールドの数を検証します

テンプレートタイプが提供する入力の数を検証するセンサーコンテンツコンパイラーのパッチは、2024 年 7 月 19 日に開発され、2024 年 7 月 27 日にクラウドストライクの内部ビルドツールの一部として本番稼動しました。センサーコンテンツコンパイラーのパッチは、すべてのプラットフォームで、他のテンプレートタイプが不正確な入力数を提供していないことも確認しました。

2. チャネルファイル 291 のコンテンツインタープリター入力フィールドのランタイム配列境界チェックが欠落していた

調査結果：チャネルファイル 291 のラピッドレスポンスコンテンツは、コンテンツインタープリターに入力ポインター配列の 21 番目のエントリを読み取るように指示しました。しかし、IPC テンプレートタイプは 20 個の入力しか生成しません。その結果、21 番目の入力にワイルドカード以外の一致基準を使用したラピッドレスポンスコンテンツが配信されると、コンテンツインタープリターは入力配列の境界外の読み取りを実行しました。これは任意のメモリ書き込みの問題ではないことが、独立した調査にて確認されています。。

緩和策：チャネルファイル 291 のラピッドレスポンスコンテンツ用に、ランタイム入力配列境界チェックをコンテンツインタープリターに追加します

2024 年 7 月 25 日に、入力文字列を取得するコンテンツインタープリター機能に境界チェックが追加されました。入力配列のサイズがラピッドレスポンスコンテンツで想定される入力数と一致するか否かのチェックも同時に追加されました。これらの修正は、センサーソフトウェアのホットフィックスリリースを通じて、すべての Windows センサーバージョン 7.11 以降にバックポートされます。このリリースは、2024 年 8 月 9 日までに一般公開される予定です。

追加された境界チェックは、コンテンツインタープリターが入力配列の境界外アクセスを実行してシステムをクラッシュさせるのを防ぎます。追加のチェックにより、入力配列のサイズがラピッドレスポンスコンテンツで想定される入力数と一致するかどうかのランタイム検証のレイヤーが追加されます。

チャネル 291 テンプレートタイプのファズテストを完了し、センサー内の追加のラピッドレスポンスコンテンツハンドラーにも拡張しています。

緩和策：IPC テンプレートタイプによって提供される入力の数を修正します

IPC テンプレートタイプを定義するセンサーコードは、正しい入力数（21）を提供するように更新されました。この修正は、センサーソフトウェアのホットフィックスリリースを通じて、すべての Windows センサーバージョン 7.11 以降にバックポートされます。このリリースは、2024 年 8 月 9 日までに一般公開される予定です。

3. テンプレートタイプのテストでは、より幅広い一致基準をカバーする必要がある

調査結果：IPC テンプレートタイプの開発中に、手動テストと自動テストの両方が実行されました。このテストでは、テンプレートタイプを介したセキュリティ関連データの正しいフローを含むテンプレートタイプの機能検証と、開発テストケースで作成された基準に基づいて適切な検知アラートを生成するためのデータの評価に重点が置かれていました。

自動テストでは、社内外のツールを活用して、想定される運用ユースケースの広範なサブセットにおいて、サポートされているすべての Windows バージョンで IPC テンプレートタイプを実行するために必要なセキュリティ関連データを作成しました。自動テストでは、より広範な運用上の期待を代表して、テレメトリと検知アラートの作成を検証するために、12 の静的テストケースセットが選択されました。このテストの一部には、テストケース内で使用するチャネルファイルの定義が含まれていました。チャネルファイル内のデータの選択は手動で行われ、すべてのテンプレートインスタンスの 21 番目のフィールドに正規表現のワイルドカード一致基準が含まれていました。つまり、開発およびリリースビルド中にこれらのテストを実行しても、21 ではなく 20 の入力が提供された場合には、コンテンツインタープリターで読み取られた潜在的な境界外読み取りは発生しませんでした。

緩和策：テンプレートタイプの開発中にテスト範囲を増やします

各テンプレートタイプのすべてのフィールドが検証されていることを確認するために、各フィールドのワイルドカード以外の一致条件を使用してテストする自動テストが作成されました。このステップは既存のすべてのテンプレートタイプに対して実行されており、今後のすべてのテンプレートタイプでも要求されます。さらに、今後リリースされるすべてのテンプレートタイプには、本番環境での使用をより適切に反映する追加のシナリオによるテストケースが含まれます。

4. コンテンツバリデーターにロジックエラーが含まれていた

調査結果：コンテンツバリデーターは新しいテンプレートインスタンスを評価しました。しかし、IPC テンプレートタイプには 21 の入力が提供されることを想定して評価しました。その結果、問題のあるテンプレートインスタンスがコンテンツインタープリターに送信されました。

緩和策：コンテンツバリデーターで追加のチェックを作成します

コンテンツバリデーターは、テンプレートインスタンス内のコンテンツに対する新しいチェックを追加するように修正されています。これは、コンテンツインタープリターへの入力として提供されているフィールドよりも多くのフィールドに一致する一致条件が含まれないよ

うにするためのもので、この修正は、2024 年 8 月 19 日までに本番環境にリリースされる予定です。

緩和策：問題のあるチャネル 291 ファイルの作成を防ぎます

コンテンツバリデーターは、21 番目のフィールドでワイルドカード一致基準のみを許可するように修正されました。これにより、20 個の入力のみを提供するセンサーでの境界外のアクセスが防止されます。

5. テンプレートインスタンスの検証は、コンテンツインタープリター内でのテストを含むように拡張する必要がある

調査結果：新しくリリースされるテンプレートタイプでは、リソースの使用率、システムパフォーマンスへの影響、検知量など、さまざまな側面からストレステストが実施されます。IPC テンプレートタイプを含む多くのテンプレートタイプに対して、関連するデータフィールドのあらゆる値と照合してシステムに悪影響を与える相互作用を特定するために、特定のテンプレートインスタンスを使用してテンプレートタイプのストレステストが実施されます。

テストテンプレートインスタンスを使った IPC テンプレートタイプのストレステストが、さまざまなオペレーティングシステムとワークロードで構成されるテスト環境で実行されました。IPC テンプレートタイプはストレステストに合格し、使用の確認が取れ、テンプレートインスタンスは、ラピッドレスポンスコンテンツの更新の一環として本番環境にリリースされました。

しかし、コンテンツバリデーターでテストされたテンプレートインスタンスでは、IPC テンプレートタイプからコンテンツインタープリターに提供された際の入力数の不一致がシステムクラッシュを発生させることは観測されませんでした。

緩和策：コンテンツコンフィグレーションシステムのテスト手順を更新します

コンテンツコンフィグレーションシステムを新しいテスト手順で更新し、作成時に最初のテンプレートインスタンスがテンプレートタイプでテストされるか否かに関係なく、すべての新しいテンプレートインスタンスがテストされるようになりました。これにより、本番環境

への展開前にテンプレートインスタンスに対して追加のテストを実行できるようになります。

6. テンプレートインスタンスには段階的な展開が必要

調査結果：各テンプレートインスタンスは段階的に展開する必要があります。

緩和策：コンテンツコンフィグレーションシステムを更新し、展開レイヤーと受諾チェックを追加します

段階的な展開により、新しいテンプレートインスタンスがシステムクラッシュ、フォールスポジティブ検知の急増、パフォーマンスの問題などの障害を引き起こした場合の影響を軽減できます。カナリアテストに合格した新しいテンプレートインスタンスは、より広い展開サイクルに順次昇格されるか、問題が検知された場合はロールバックされます。各サイクルは、より広範囲に展開する前に潜在的な問題を特定して軽減するように設計されています。テンプレートインスタンスを次のサイクルに昇格させた後、追加の安定化時間が設けられ、その間にテレメトリが収集され、テンプレートインスタンスがエンドポイントに与える全体的な影響が評価されます。

緩和策：ラピッドレスポンスコンテンツの更新の展開を、お客様が管理できるようにします

Falcon プラットフォームが更新され、お客様はラピッドレスポンスコンテンツの配信をより制御できるようになりました。お客様は、ラピッドレスポンスコンテンツの更新を展開する場所とタイミングを選択できます。今後もこの機能を強化して、ラピッドレスポンスコンテンツの展開をより細かく制御できるようにしていきます。また、お客様が購読できるリリースノートを通してコンテンツ更新の詳細も提供します。

独立したサードパーティによるレビュー

クラウドストライクは、独立したサードパーティソフトウェアセキュリティベンダー二社に、Falcon センサーのコードをセキュリティと品質保証の両面からのさらなるレビューを依頼しました。さらに、開発から展開までのエンドツーエンドの品質プロセスについて、独立したレビューを実施しています。どちらのベンダーも、7 月 19 日の影響を受けたコードとプロセスを即座の焦点としたレビューを開始しました。

技術的な詳細

背景と用語

クラウドストライクは、2 つの方法でセンサーにセキュリティコンテンツコンフィグレーションの更新を提供しています。「センサーコンテンツ」は、弊社のセンサーに含まれる形で直接リリースされ、「ラピッドレスポンスコンテンツ」は、変化する脅威の状況に迅速に対応するために設計されています。

正規表現ベースのラピッドレスポンスコンテンツをセンサーで処理するためには、以下のようなコンポーネントが関係してきます。

- コンテンツインタープリター：センサーの C++コードの一部で、入力文字列を正規表現と照合してテストすることができます。
- テンプレートタイプ：脅威検知エンジニアがラピッドレスポンスコンテンツで活用できる事前定義されたフィールドが含まれています。テンプレートタイプはコードで記述され、ビルド時にセンサーにコンパイルされます。
- テンプレートタイプ定義ファイル：各テンプレートタイプのパラメーターを定義します。このファイルの定義には、各テンプレートタイプのラピッドレスポンスコンテンツを配信するチャネルファイル、テンプレートタイプが使用する入力の数、各入力に必要なデータの種類に関する情報が含まれます。
- センサーコンテンツ：特定の検知判断を行うために、セキュリティ関連データとラピッドレスポンスコンテンツを組み合わせる方法を決定します。センサーコンテンツには、オンセンサーの AI モデル、機械学習モデル、およびテンプレートタイプが含まれます。これは、センサーのリリースの一部としてコンパイルされます。
- テンプレートインスタンス：検知エンジニアが開発した一致基準。テンプレートインスタンスは、特定のテンプレートタイプで使用するための正規表現コンテンツで構成され、セキュリティ運用に使用する特定のデータを識別します。また、テンプレートタイプ定義ファイルによる UI ドリブンを使用して定義されます。
- ラピッドレスポンスコンテンツ：複数のテンプレートインスタンスのバンドルにより構成されています。ラピッドレスポンスコンテンツはチャネルファイルごとに提供されます。
- コンテンツバリデーター：チャネルファイルの有効性をテンプレートタイプ定義ファイルの定義と照合します。
- コンテンツコンフィグレーションシステム：テンプレートインスタンスを作成するために使用されます。テンプレートインスタンスは、チャネルファイルと呼ばれるメカニズムを通じて検証され、センサーに展開されます。

セキュリティ製品におけるカーネルドライバの使用

Microsoft セキュリティブログでデビッド・ウェストン（David Weston）が概説しているように、Falcon センサーを含む Windows のエコシステムにおけるセキュリティ製品は、一般的に、堅牢なセキュリティ製品のコアコンポーネントとしてカーネルドライバが活用されています。

カーネル内に存在することで、プロセスやスレッドの作成、ディスク上でのファイルの書き込み、削除、変更など、システム全体のセキュリティ関連アクティビティを詳細に把握できるようになります。カーネルが公開するインターフェースにより、クラウドストライクのドライバは、悪意のあるプロセスをインラインで防御したり、マルウェアのファイルがディスクに書き込まれるのをブロックするなど、セキュリティ製品にとって重要な制御を実施することができます。

クラウドストライクのカーネルドライバはシステム起動の初期段階から読み込まれ、ユーザーモードプロセスが開始する前に起動するマルウェアをセンサーが監視して防御できるようにします。

これらのカーネル機能に最新のセキュリティコンテンツ（クラウドストライクのラピッドレスポンスコンテンツなど）を提供することで、センサーはカーネルコードに変更を加えることなく、急速に進化する脅威の状況からシステムを防御することができます。ラピッドレスポンスコンテンツは構成データであり、コードやカーネルドライバではありません。

クラウドストライクは、Windows Hardware Quality Labs（WHQL）プログラムを通じて、新しい Windows センサーのリリースをそれぞれ認証します。これには、Microsoft の Windows Hardware Lab Kit（HLK）および Windows Hardware Certification Kit（HCK）で必要なすべてのテストによる広範なテストが含まれます。WHQL 認証プロセスは、機能テスト、耐久テスト、フォールトインジェクションを含むストレステスト、ファジング、パフォーマンステストを含む包括的な内部テストの終了を意味します。WHQL プログラムに必要なテスト中には、センサーは認証時の最新バージョンのチャネルファイルを使用します。

Windows の新しいバージョンでは、ユーザー空間でこれらのセキュリティ機能をさらに実行するためのサポートが導入されているため、クラウドストライクではこのサポートを利用できるようエージェントを更新します。

Windows エコシステムが、少なくとも一部の機能でカーネルドライバーに依存しない堅牢なセキュリティ製品をサポートするには、まだ多くの作業が必要です。私たちは、Windows がユーザー空間におけるセキュリティ製品のニーズに対するサポートをさらに追加していく中で、Microsoft と継続的に直接協力していくことを約束します。

クラッシュダンプ分析

チャネルファイル 291 の新しいテンプレートインスタンスがどのようにシステムクラッシュにつながったかを説明するために、問題のコンテンツの影響を受けたシステムのカーネルクラッシュダンプを簡単に検証します。これは、デビッド・ウェストン（David Weston）が Microsoft セキュリティブログで共有したクラッシュ分析を拡張させたものです。

Windows カーネルデバッガでクラッシュダンプを開き、標準の!analyze –v コマンドを使って簡単に要約すると、メモリ障害（「アクセス違反」とも呼ばれる）のバグチェックが発生していることが確認できます。（注：関係のないデバッグの詳細は省略しています。ここでは代表的なクラッシュダンプが分析されています。マシンの状態の詳細に応じて、ダンプにはさまざまなバリエーションが存在します。）

```
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
```

この自動トリアージコマンドは、境界外メモリアクセスを実行しているドライバとして `csagent.sys` を特定します。`csagent.sys` は、クラウドストライクのファイルシステムフィルタードライバで、Windows オペレーティングシステムのコンポーネントに登録し、セキュリティに関連するシステムアクティビティの通知をリアルタイムで受信するカーネルドライバの一種です。

クラウドストライクのドライバが登録する通知の中には、名前付きパイプの作成に関する通知があります。ドライバが名前付きパイプ通知を受信すると、このデータはシステムに関する他のコンテキスト情報と結合されます。この結合されたデータは、チャネルファイル 291 で伝達されるテンプレートインスタンスに対する評価のために提示されます。

このプロセスをより詳しく見るために、トラップフレームを復元し、直前の命令を逆アセンブルして、境界外メモリを読み込んだ時点のレジスタの状態を表示します。(注:この逆アセンブルリストは、コードにわかりやすいシンボル名を付けるために、標準のデバッガ出力から変更されています。)

```
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]
```

このコードスニペットの前に、名前付きパイプ通知からのコンテキストデータが、バッファアドレスとサイズ値を保持する文字列構造を指す 20 個の入力ポインターの配列として、IPC テンプレートタイプ用に用意されています。このスニペットは、チャネルファイル 291 で指定されたインデックスに従って、入力の 1 つを選択してバッファアドレスとサイズを返すことを目的としています。

このコードを入力すると、20 入力ポインタ配列のアドレスが rax レジスタに格納され、r11 レジスタは、取得される入力がインデックス 0x14、つまり 21 番目の要素にあることを示します。

入力配列を調べてみると、確かに入力文字列構造へのポインターが 20 個配列されており、その後に有効なメモリを指していない 21 番目の値が続いています。

```
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
```

この無効なポインタを r8 レジスタに読み込んだ後、上記のスニペットの制御フローは、アドレス `fffff802`0ebc14e8` への最初のジャンプを実行し、NULL ポインタのチェックを実行し、その後無効なポインタを介して読み取りを試行します。その結果、範囲外の読み取りとそれに続くバグチェックが発生します。

追加のリソース

修復とガイダンスハブ：Windows ホスト向け Falcon コンテンツ更新
ブログ：技術情報：Windows ホスト向け Falcon コンテンツ更新
修復ハブ – 用語集