Deloitte.
デロイト トーマツ

Deloitte Cyber Trends & Intelligence Report 2024

Feb. 2025

デロイト トーマツ サイバー合同会社
Cyber Intelligence Center

はじめに

2021 年に始まり 4 年目を迎える本レポートは、公開時期を年末から年初へと変更し、前年 1 年間の脅威動向や攻撃の傾向を振り返る二つの記事を用意しました。これらの記事で構成された第 1 章「2024 年の脅威動向」は比較的広い視野での脅威動向や攻撃傾向に興味のある皆様を対象としています。さらに今回は、Cyber Intelligence Center（CIC）で行っている技術検証の内容やそれを踏まえた脅威対策を解説する二つの深堀り記事で構成された第 2 章「CIC における技術検証」を加えました。こちらはセキュリティ監視・分析業務、CSIRT など直接技術的な業務に関わる皆様を対象としています。

「ランサムウェア脅威の動向」では 2019 年より CIC が独自に統計を取り続けているリークサイトでのデータ公開件数を引き続き紹介します。2023 年と比較して総数での大きな変化は見られませんが、二重恐喝ランサムウェアの一般化や攻撃グループの勢力図の変遷などの傾向がみられます。加えて、ランサムウェア攻撃の影響を大きく受ける初動対応に焦点を当てて、インシデント対応計画などで考慮すべき事項を解説します。

「VPN 製品の脆弱性と関連する攻撃の観測状況」では近年攻撃の侵入経路としてよく悪用される VPN 機器の脆弱性について、実際の脆弱性公開状況と CIC で観測しているそれらの脆弱性への攻撃試行を集計して紹介します。これらの集計とその分析結果は、VPN 機器そのものに何らかの理由があるというよりは、機器の運用管理が十分でなかった可能性と ASM（Attack Surface Management)による定期的な監視の有用性についても示唆しています。

「Windows Downdate の観察」では Blackhat USA 2024 で発表された Windows の脆弱性とその攻撃手法について CIC で検証した内容と対策を詳解します。当該攻撃手法は Windows の最新の防御機構をバイパスして認証情報を奪取するもので、2024 年 12 月時点でも根本的な対策が難しい状況です。CIC ではこのような影響度の大きい脆弱性やその攻撃手法について独自に検証を行い、対応方針を検討しています。

「EDR 回避手法の検証」では EDR 製品をインストールした環境で、それらに検知されずに C2 エージェントを実行する複数の手法を試行した結果と、これらの試行のような EDR を前提とした攻撃への対策の考え方を紹介します。CIC ではこのような洗練された攻撃に対処するため、検証結果に基づいた実際的な検知・防御手法を提供しています。

本レポートが日々高度化するセキュリティ対策の一助になれば幸いです。

第 1 章　2024 年の脅威動向

ランサムウェア脅威の動向

ランサムウェア被害は 2024 年も引き続き世界中で発生しており、脅威が収束する兆しは見えません。
ランサムウェア攻撃の手口では、システムの暗号化だけでなく、データを盗み出して金銭支払いに応じなければインターネット上に公開すると脅す「二重恐喝」がますます一般的になっています（図 1）。警察庁が公開している「令和 6 年上半期におけるサイバー空間をめぐる脅威の情勢等について」[1]によると、国内で報告されたランサムウェア被害のうち二重恐喝手口によるものが 8 割以上を占めています。攻撃グループがデータ公開のために運営しているリークサイトの数は、2024 年にデータ公開が行われたものだけで 90 個以上に及びます。

二重恐喝はデータ公開という形で被害が可視化されることから、リークサイトにおけるデータ公開件数はランサムウェアの脅威を定量的に評価するうえで有効です。本項では、当社 CIC が二重恐喝の始まった 2019 年後半から行っているリークサイトのモニタリング結果に基づき、ランサムウェア脅威の動向を解説します。
なお、リークサイトでのデータ公開は攻撃グループが自身のサイトで攻撃に成功したと主張しているものであり、虚偽や誇張が含まれる可能性があります。データが公開されたことと、実際にランサムウェア被害があったかはイコールではない点に留意が必要です。

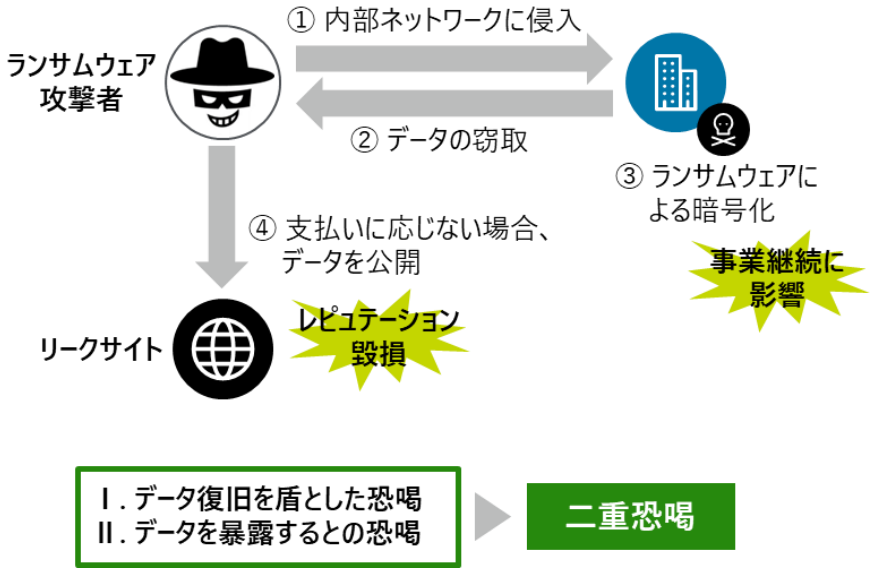

図 1　二重恐喝ランサムウェアの手口

1 https://www.npa.go.jp/publications/statistics/cybersecurity/data/R6kami/R06_kami_cyber_jousei.pdf

二重恐喝によるデータ公開被害の状況

2024 年にリークサイトでのデータ公開被害にあった組織の数は約 5,200 件と、前年の約 1.2 倍となりました。

図 2 は、リークサイトでのデータ公開件数を半年ごとにまとめたものです。（上期は 1 月～6 月、下期は 7 月～12 月）

データ公開件数は 2022 年下期までは半年に約 1,000 件のペースで推移していたものの、2023 年上期から増加傾向にあります。

月間の件数で比べると、2022 年頃までは 200 件前後だったのが、2024 年は平均 430 件以上と倍増しています。

次に日本企業のデータ公開被害を見ると図 3 の通りです。ここでは、国内拠点、海外現地法人の被害をあわせて日本企業の被害としています。

日本企業全体では世界全体ほどの顕著な件数の変化は見られないものの、国内拠点の被害件数は徐々に増加しています。

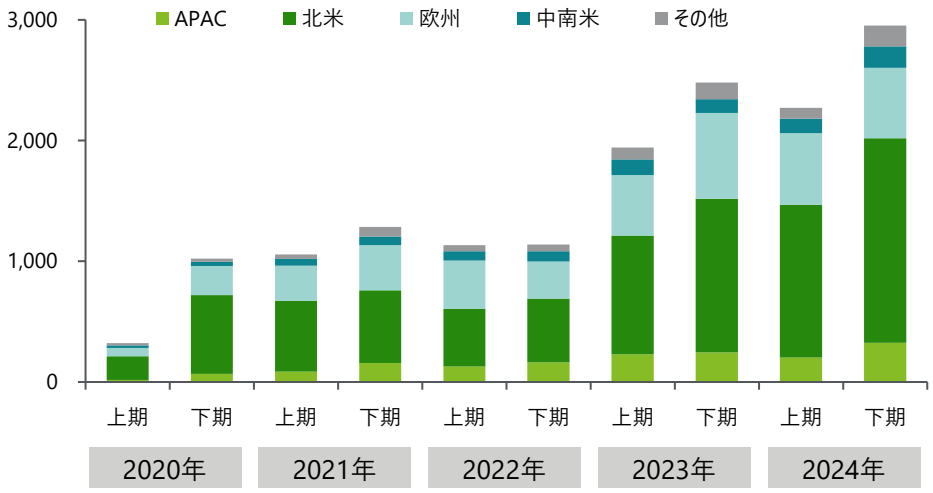

図 2　リークサイトでデータ公開被害にあった組織の件数の推移

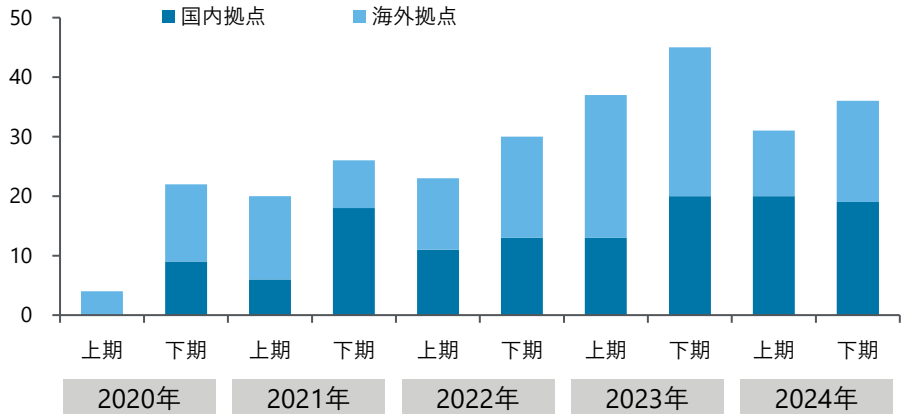

図 3　日本企業のデータ公開被害件数の推移

ランサムウェア攻撃グループ別のデータ公開件数

続いて、二重恐喝によるデータ公開についてランサムウェア攻撃グループごとの件数を見ていきます。

図 4 は、2024 年に 100 件以上のデータ公開を行ったグループについて、グループごとの件数を示したものです。

最も被害が多かったのは RansomHub で、2024 年のデータ公開件数の全体の約 10%を占めています。2024 年 2 月のリークサイト開設以降、驚異的なペースでデータ公開を行っています。

2023 年に首位だった LockBit は 2 位と、依然としてデータ公開件数は多いものの、2024 年下期は大幅に減少しています。

このほか、Hunters や BlackSuit、Meow などは 2024 年に入ってデータ公開件数が大幅に増加しています。

このようにランサムウェア攻撃グループの勢力図は目まぐるしく変化しています。

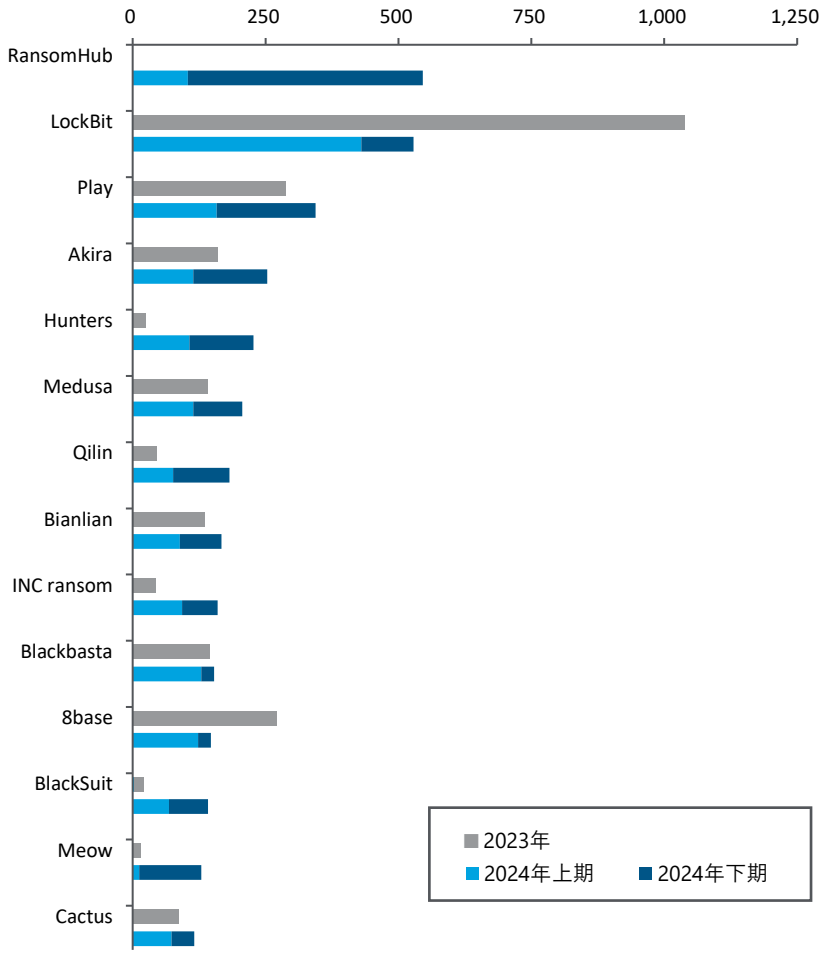

図 4　ランサムウェア攻撃グループごとのデータ公開件数

ランサムウェア被害と初動対応

前述のようにランサムウェア攻撃は世界中で猛威を振るっており、日本企業も例外ではありません。
毎年、一般紙で報じられるような大規模なランサムウェア被害が複数発生しています。
こうした大規模被害では、IT システムのほとんどが利用不能になり数か月にわたって事業に支障を来すことも珍しくありません。
このため、ランサムウェア攻撃への備えでは被害予防のための検知・防御態勢の整備だけでなく、万が一発生した場合の対応を検討しておくことも重要です。
大規模なランサムウェア被害が発生した場合、被害組織では大きく次の対応を行うことになります。

- 初動対応
  被害の拡大防止や対策本部の立ち上げなど
- 内部対応
  非常事態下での業務の継続・復旧
- 外部対応
  顧客、取引先、行政や攻撃者への対応など

通常のサイバー攻撃被害では、事前に策定したインシデント対応計画などに沿って対応するのが一般的です。
しかし、ランサムウェア攻撃ではその特性によって既存のインシデント対応計画がうまく機能しないことがあり得ます。

本レポートでは、特にランサムウェア攻撃の特性の影響を大きく受ける初動対応に焦点を当てて、インシデント対応計画などで考慮すべき事項を解説します。

休日・夜間を狙った攻撃

ランサムウェア攻撃では、ネットワーク内に侵入してドメインコントローラーの管理者権限を奪取し、それを悪用してランサムウェアを配布・実行するのが常套手段です。
攻撃者はファイルの窃取やドメインコントローラーの管理者権限の奪取などを行ったうえで、最後にランサムウェアを実行してデータを暗号化します。
データの暗号化にあたって、ランサムウェア攻撃者はインシデント対応を遅らせるために夜間や休日を狙ってランサムウェアを実行することが知られています。
図 5 は、2024 年にランサムウェア被害を公表した日本の組織のプレスリリースなどから、暗号化が行われた曜日を集計したものです。
図 5 で月曜日となっているものは「月曜日朝に暗号化されていることが発覚した」ケースが多く、実際には土日に暗号化が行われたと推測されます。
月曜日の分を土日に暗号化されたと仮定すると被害の約半数が土日に起こっていることになり、国内でもランサムウェア被害は土日に発生する可能性が高いといえます。
ランサムウェア被害が土日に発生することを考えると、休日・夜間の連絡体制や、被害拡大防止措置について事前にインシデント対応計画などに定めておくことが重要です。

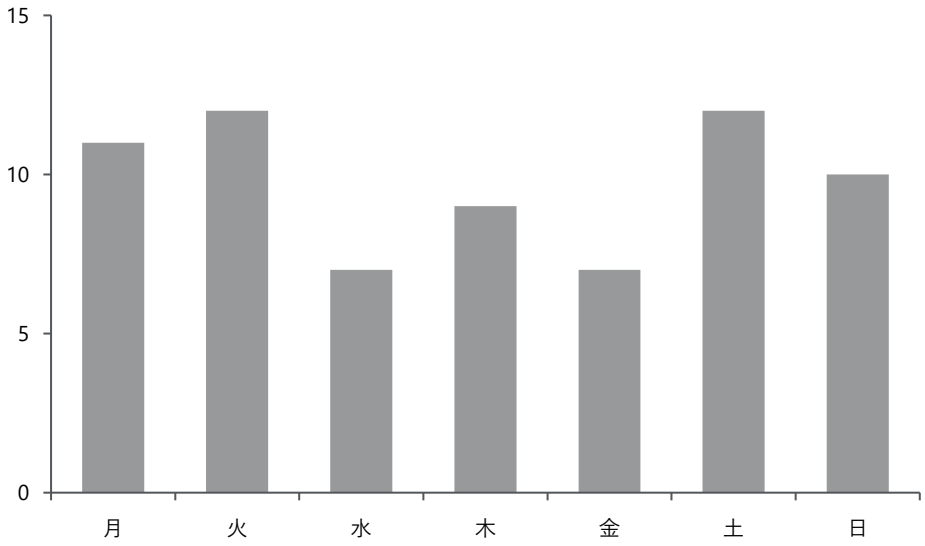

図 5　暗号化が行われた曜日

システムを利用できない状況に備えた準備

深刻なランサムウェア被害を受けた場合、インシデント対応にあたって問題となるのが社内システムのダウンによるコミュニケーションへの支障です。

メールサーバーのダウンによって、メールで緊急の連絡をとれなくなることが想定されます。

また、インシデント発生時に社内システム上の基盤に情報を集約する態勢をとっている場合、その基盤を利用できなくなることでインシデント対応の根幹が揺らぐことになります。

このほか、インシデント対応計画などをファイルサーバー上に置いている場合、暗号化によって閲覧できなくなる状況が生じる可能性があります。

このように深刻なランサムウェア被害では、インシデント対応の前提を覆される事態が生じることに留意する必要があります。

ランサムウェア攻撃への備えとして、次のような準備を行っておくことが重要です。

- 外部のコミュニケーションツールの確保
  インシデント発生時のコミュニケーションツールについて、社内システムによらない手段を準備しておく
- インシデント対応計画の印刷など
  インシデント対応計画などのインシデント発生時に閲覧する必要がある文書について、印刷して配布しておくか、社用携帯電話にデータ保存しておく

深刻なランサムウェア被害が発生すると、今まで当たり前だったことができなくなると認識しておく必要があります。

万が一、実際に被害が発生した際の混乱を最小限に抑えるためにも、平時から最悪のシナリオを想定して準備しておくことが重要です。

VPN 製品の脆弱性と関連する攻撃の観測状況

はじめに

2020 年以降、IPA への不正アクセス届出では、SSL-VPN 機器の脆弱性を悪用したとみられる被害事例が継続的に報告されています[2]。

他にも、SSL-VPN 機器の脆弱性に関する注意喚起や被害事例、対策などを様々な企業や機関がレポートやブログで取り上げています[3][4][5]。

本稿では、SSL-VPN 機器について、脆弱性報告状況とそれらを狙う攻撃の CIC における観測状況を紹介します。

大手 6 社の提供する機器に関する調査

SSL-VPN 機器の脆弱性報告件数を調査するにあたり、調査対象を次の条件に該当する脆弱性としました。

- SSL-VPN 機器を提供している大手 6 社（Ivanti、F5、Cisco、Citrix、Check Point、Fortinet）の製品の脆弱性
- CVE が採番されており、CVSS のスコアが 9.0 以上で影響度が高いと考えられるもの

2020 年以降に報告された脆弱性の件数を年次で集計した結果を図 6 に示します。総数は 85 件でした。

2020 年から 2021 年にかけて増加している一方、2021 年以降は減少傾向です。

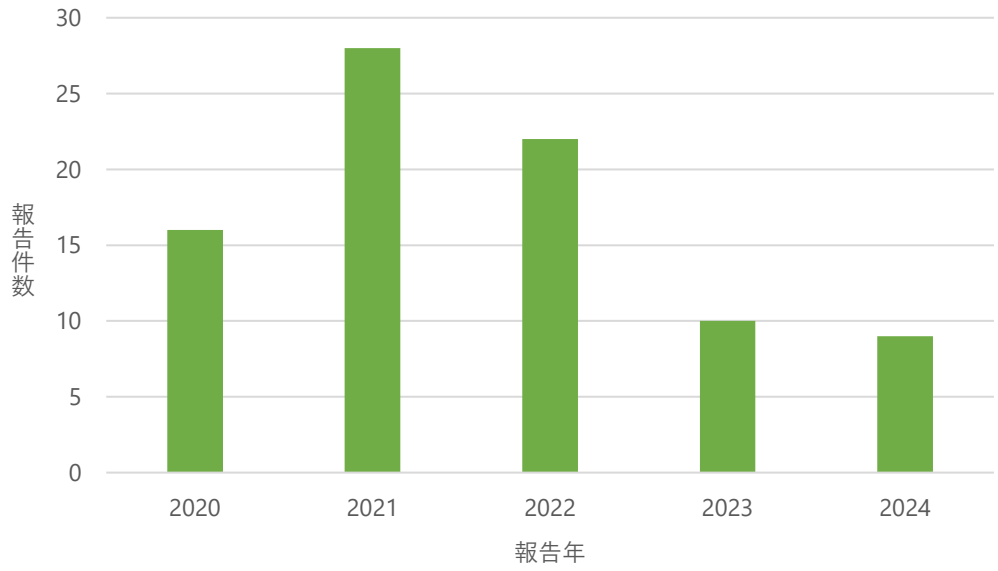

図 6　2020 年から 2024 年までの調査対象脆弱性報告件数の推移

[2] IPA,コンピュータウイルス・不正アクセスに関する届出について　https://www.ipa.go.jp/security/todokede/crack-virus/about.html

[3] 株式会社ラック,LAC Security Insight 第 9 号 2024 年夏　https://www.lac.co.jp/lacwatch/pdf/20240820_lsi_vol9.pdf

[4] Zscaler,最新の VPN リスク レポート：56%の組織が VPN の脆弱性を悪用した攻撃を経験
https://www.zscaler.com/jp/blogs/security-research/new-vpn-risk-report-56-enterprises-attacked-vpn-vulnerabilities

[5] CISA,MODERN APPROACHES TO NETWORK ACCESS SECURITY
https://www.cisa.gov/sites/default/files/2024-06/joint-guide-modern-approaches-to-secure-network-access-security-508c.pdf

脆弱性の種類ごとの割合を図 7 に示します。リモートコード実行（RCE）と認証バイパスが多数を占めています。
次に、大手 6 社のメーカー別報告件数の割合を図 8 に示します。Cisco 社、F5 社の割合が多い結果となりました。

Cisco 社については、中小企業向けの一部製品で複数の脆弱性が同時に報告されたことにより件数が多くなっています。
F5 社については、BIG-IP における影響度の高い脆弱性が調査期間全体を通して継続的に報告されています。

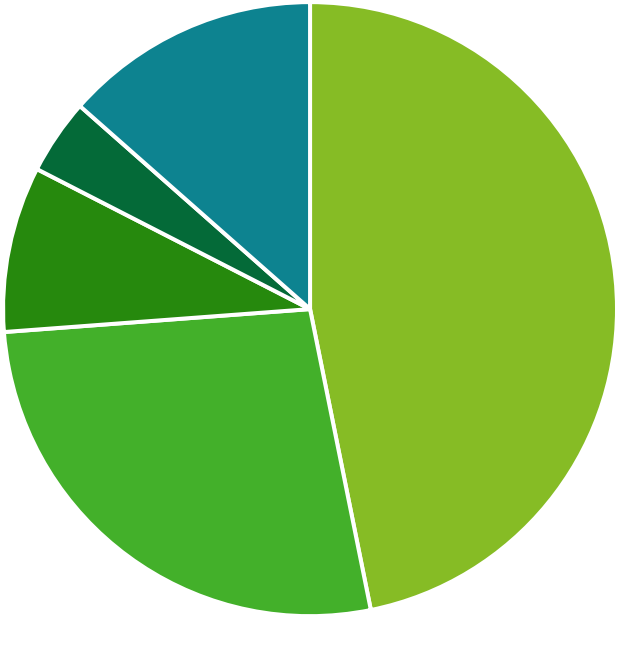

図 7　調査対象脆弱性の種別割合

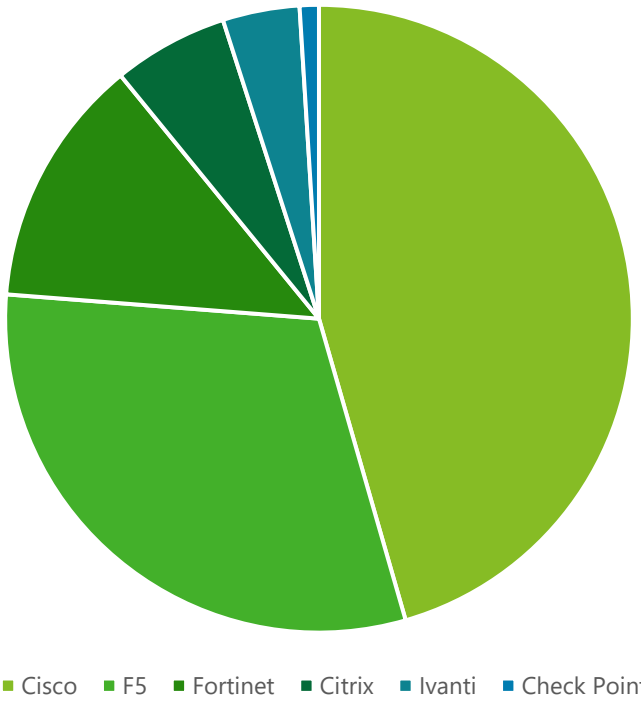

図 8　調査対象脆弱性のベンダー別報告件数割合

CIC での攻撃観測状況

前項で対象とした脆弱性について、それらを狙う攻撃の 2024 年の観測状況を集計しました。対象期間における検知数の推移を図 9 に示します。3 月と 11 月には一部の脆弱性を狙ったスキャンが大量に検知されたため、特に検知数が多くなっています。

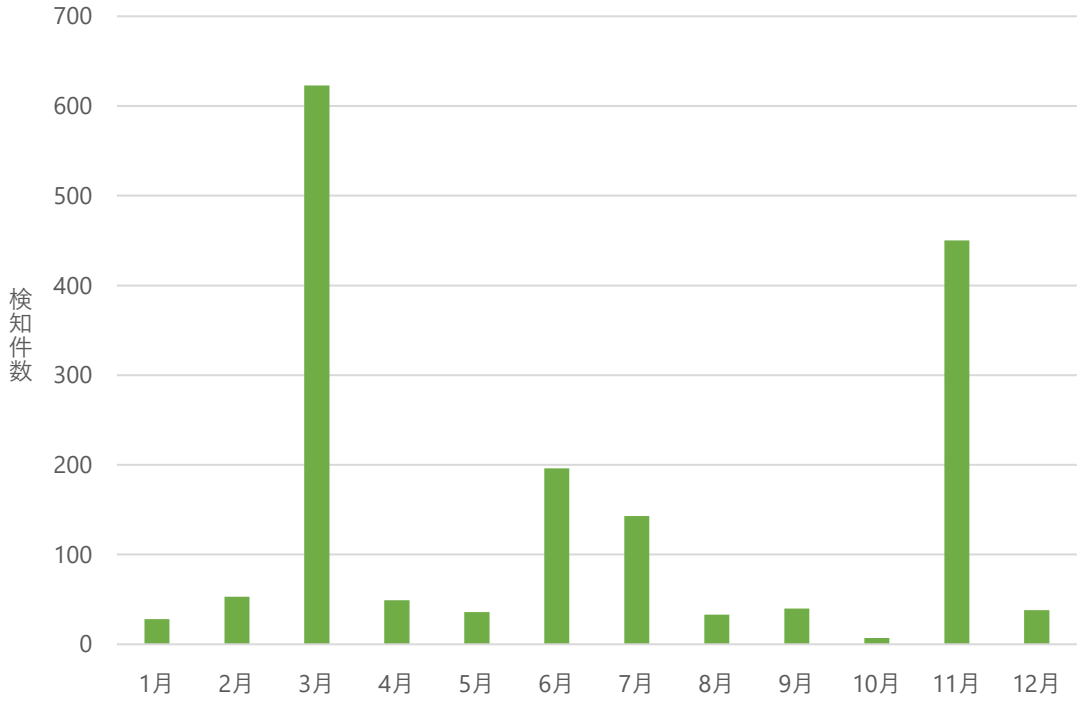

図 9　2024 年の CIC における対象脆弱性を狙った攻撃の検知数推移

次に、攻撃元 IP アドレスの ISP ごとの集計を図 10 に、国ごとの集計を図 11 に示します。ISP はホスティング会社が多くを占める結果となりました。また、国ごとの集計では、アメリカ合衆国、日本、オランダの 3 カ国で 50%を占める結果となりました。

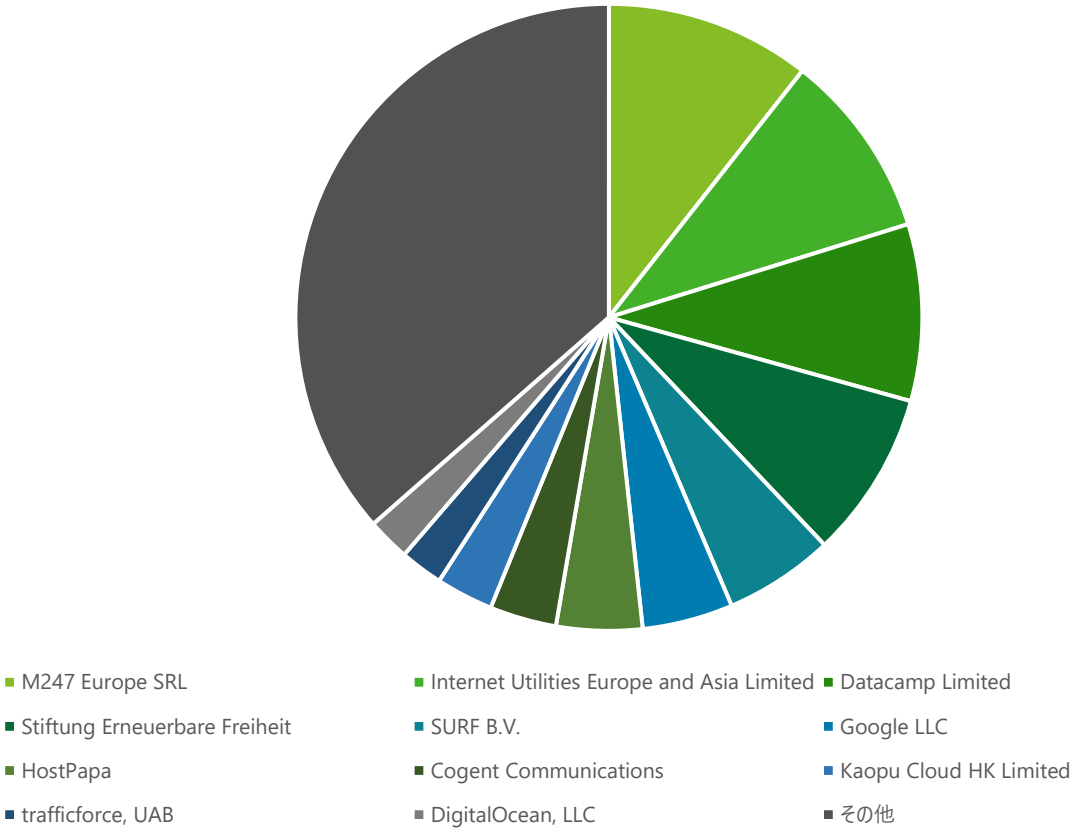

図 10　攻撃送信元 IP の ISP 別割合

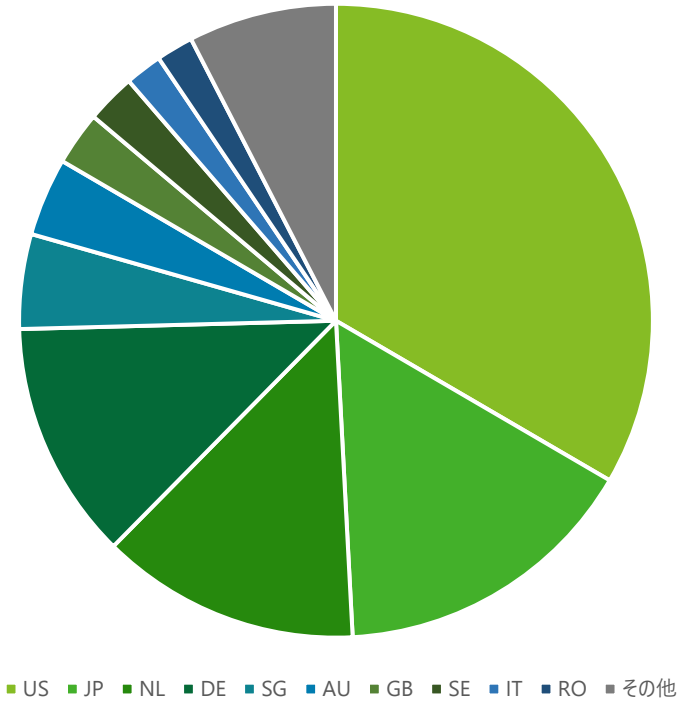

図 11　攻撃送信元 IP の国別割合

対象脆弱性のうち、CIC で観測した攻撃の件数が多い上位 3 件を表 1 に示します。新しく見つかった脆弱性だけでなく、古い脆弱性についても継続的に攻撃が行われていることを観測しています。
また、狙われた脆弱性のうち CVE の公開日が比較的新しい 2 件について、公開されてから攻撃が最初に観測されるまでの期間を表 2 に示します。影響度が高く新しい脆弱性についても、公開されてから攻撃が観測されるまでの期間は必ずしも短いわけではないことが分かります。
これらのことから、SSL-VPN 機器の脆弱性に対する攻撃は、未知の脆弱性だけでなく、既知の脆弱性を放置した場合にも発生する可能性が高いことが示唆されます。そのため、既知の脆弱性に対しても迅速な対策が必要です。

表 1　CIC で観測した攻撃の件数が多い脆弱性上位 3 件

順位	CVE	CVE 公開日	CVE の概要
1 位	CVE-2022-1388	2022-05-05	BIG-IP 製品における認証バイパスの脆弱性
2 位	CVE-2023-46747	2023-10-26	BIG-IP 製品における認証バイパスの脆弱性
3 位	CVE-2024-21887	2024-01-12	Ivanti 製品におけるコマンドインジェクションの脆弱性

表 2　CVE 公開日から攻撃観測開始日までの期間

CVE	CVE 公開日	攻撃観測開始日	公開日から攻撃観測開始日までの期間
CVE-2024-21887	2024-01-12	2024-01-19	7 日
CVE-2023-46747	2023-10-26	2024-03-01	128 日

脆弱性の公開から攻撃に至るタイムライン

前項の表 2 で示した 2 つの CVE について、CVE の公開日、PoC の公開日、CIC での攻撃観測開始日をそれぞれタイムラインにまとめたものを図 12 に示します。いずれのケースでも PoC は脆弱性公開からかなり短い期間で公開されていますが、PoC が公開されてから攻撃が観測されるまでの期間には顕著な差があります。このことからも攻撃者が既知の脆弱性をどういったタイミングで悪用するかという傾向が一様ではないということが言えるでしょう。

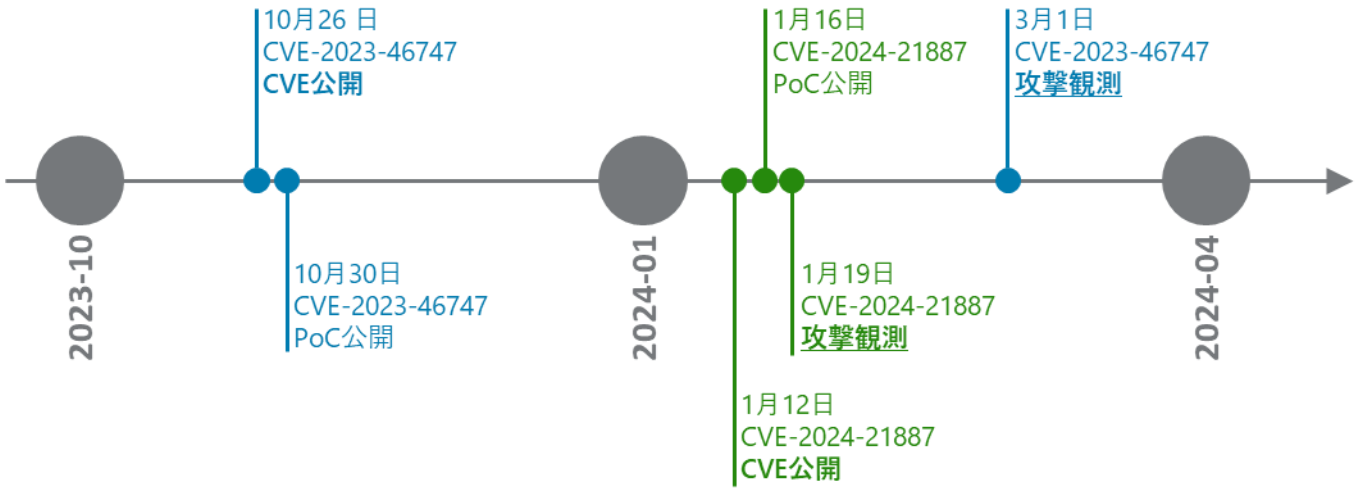

図 12　CVE 公開から PoC 公開および CIC での攻撃観測までのタイムライン

おわりに

SSL-VPN 機器について、脆弱性報告状況とそれらを狙う攻撃の CIC での観測状況を紹介しました。

調査の結果、影響度の高い脆弱性の報告件数は単純に増加しているわけではないことがわかりました。

また、CIC の観測範囲では SSL-VPN 機器の比較的古い脆弱性に対する攻撃が継続しており、必ずしも新しい脆弱性を狙う攻撃ばかりではないこともわかりました。

これらのことから、SSL-VPN 機器を狙った攻撃は、新たな脆弱性の公開直後だけでなく、恒常的に行われていると捉える必要があると考えられます。たとえこれまで被害がなかったとしても、脆弱なまま放置されている SSL-VPN 機器があればいつか被害にあう可能性があります。

このため、SSL-VPN 機器などの外部から侵害される可能性が高い機器は、確実に把握して管理することが重要です。

SSL-VPN 機器など外部に露出するノードのリスクを継続的に把握するためには、ASM（Attack Surface Management）の利用が有効です。一般的な ASM では、外部からアクセス可能な資産を定常的かつ定量的に調査する仕組みと、それらを管理するためのインターフェースが提供されるので、現状に不安がある場合はこの種の製品やサービスを検討してみることを推奨します。

なお、システムの構成や脆弱性の公開タイミングなどの事由により、どうしても緊急で対応しなければならない状況に陥る可能性も考えられます。月次や年次といった定例での対応だけでなく、緊急メンテナンスの要求に無理なく対応できる運用体制を構築しておくことが重要です。

第 2 章　CIC における技術検証

Windows Downdate の観察

はじめに

2024 年 8 月に開催された Black Hat USA 2024 カンファレンスにおいて「Windows Downdate: Downgrade Attacks Using Windows Updates」と題した発表[6]が行われました。
これは、Windows Update の仕組みを悪用して、Windows のシステム関連ファイルを任意のバージョンの正規ファイルへと「アップデート」させる（実際にはダウングレード）ことで、過去の脆弱性を再現可能な状態にするというものでした。
また、発表内容には、特定のファイルを配置することで本来必要な UEFI 上の設定変更を経ずに Windows の仮想化セキュリティ機能を無効化する手法も含まれていました。
さらに、発表時のデモでは、当時の Windows 環境において本来制限されているはずの Lsass（後述）のプロセスメモリへのアクセスが可能になり、前述の仮想化セキュリティ機能の無効化と併用することで、クレデンシャルダンプ攻撃が実現可能であることが示されました。
本件については CVE-2024-21302 および CVE-2024-38202 が割り当てられており、本稿執筆時（2024 年 12 月）においても根本原因の修正は行われておらず、緩和策のみが提供されている状況です。

本稿では、Windows Downdate のデモで公開されたクレデンシャルダンプ攻撃の再現を通して、当該脆弱性の悪用によりバイパスされた Lsass を保護する複数の仕組みを解説し、最後に攻撃の検知方法や現実的な緩和策を紹介します。

Windows Downdate とクレデンシャルダンプ攻撃

本節では、最近の Windows 環境で Lsass を保護している防御機構を紹介し、その後 Windows Downdate の PoC を再現しながらその仕組みを解説します。

資格情報（クレデンシャル）と LSASS

VBS が有効になっていない Windows 環境では、端末で認証を行うための資格情報（クレデンシャル）は Lsass というプロセスが保持しています。
そのため、Lsass のプロセスメモリにアクセスすることができれば、そこに保持されている資格情報を読み取ること（＝クレデンシャルダンプ攻撃）が可能です。
この資格情報とは、具体的には各アカウントの NT ハッシュ、Kerberos のキーやチケットの情報を指します。これらの資格情報を悪用することで、端末で実行されているサービスを利用したり、他の端末へ横展開したりすることが可能となるため、クレデンシャルダンプ攻撃は様々な局面で多用される傾向にあります。

VBS と PPL

次に、Lsass を保護する「VBS」と「PPL」を紹介します。
VBS[7]（Virtualization-Based Security）は、Windows のセキュアカーネル機能によって実現されており、仮想化機能である Hyper-V を用いて通常のユーザーモードやカーネルモードといった制約とは別に、VSM[8]（Virtual Secure Mode)という仕組みで分離された仮想空間を作り出します。

---

6 Windows Downdate: Downgrade Attacks Using Windows Updates https://www.blackhat.com/us-24/briefings/schedule/#windows-downdate-downgrade-attacks-using-windows-updates-38963
7 仮想化ベースのセキュリティ https://learn.microsoft.com/ja-jp/windows-hardware/design/device-experiences/oem-vbs
8 仮想保護モード https://learn.microsoft.com/ja-jp/virtualization/hyper-v-on-windows/tlfs/vsm

VSM には、分離ユーザーモードとセキュアカーネルモードが存在している空間があり、VTL（Virtual Trust Level）という仕組みで通常の OS 空間とのメモリアクセス制御がされています（図 13）。
現在は VTL0 と VTL1 という 2 つ VTL のみが実装されており、前者は通常の OS 空間、後者はセキュアカーネル等が存在するよりセキュアな空間です。

VBS が有効な環境では、従来 Lsass に保持されていた資格情報を VTL1 に存在する LsaIso という分離プロセスが保持しており、このプロセスでは Trustlet と呼ばれる特殊なデジタル署名やその検証が実行されます。
LsaIso は通常、Lsass へ暗号化した資格情報を連携しています。
この VBS で分離された空間に対しては、通常のカーネルモードからもメモリアクセスが行えないため、Lsass プロセスメモリから資格情報を奪取する試みを防ぐ役割があります。
この仕組みによって実現されている資格情報へのアクセス分離機能は Microsoft Defender Credential Guard と呼ばれています。
VBS は、その他にも侵害に悪用される可能性のあるカーネルメモリの割り当て領域を制限することで安全なランタイム環境内でコードを実行するメモリ整合性[9]などを備えています。

PPL（Protected Process Light）は、Windows によって提供される Protected Process 機能の拡張です。
Windows には、音楽や動画等を再生するメディア機能へのアクセスを著作権保護のために制限する目的で、管理者権限であっても特定のプロセスへはアクセスできないよう制御する Protected Process 機能が従来から実装されています。
この機能は、具体的には Windows Media 証明書によって署名されたプログラムでなければ Windows によって提供されるメディア関連の機能へのアクセスを制限する等の制約を実現します。
Lsass を保護している PPL は、この Protected Process 機能を拡張して実装されており、PPL 機能で保護されているプロセスは署名されている署名者のレベルによってアクセスを制限されます。
署名者のレベルが高いプロセスからは低いレベルの署名のあるプロセスへアクセスが可能ですが、その逆は許可されません。

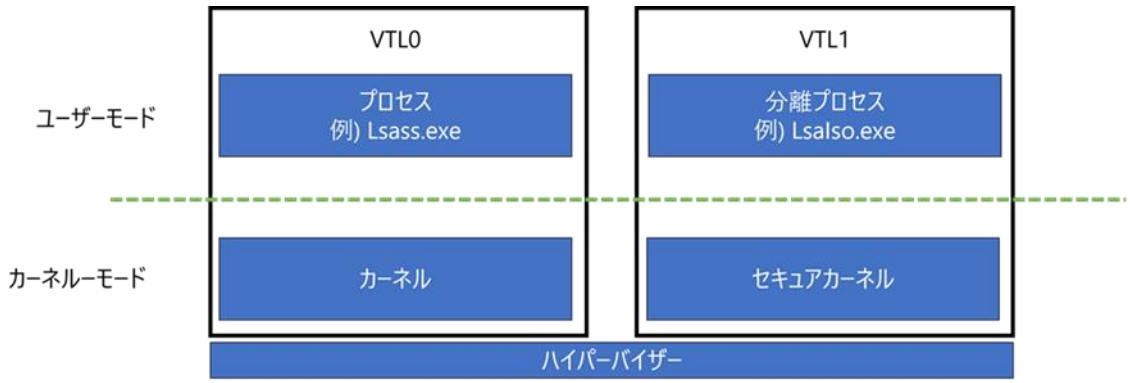

図 13　VSM のコンセプト

[9] メモリの整合性と仮想化ベースのセキュリティ　https://learn.microsoft.com/ja-jp/windows-hardware/drivers/bringup/device-guard-and-credential-guard

本機能により Lsass プロセスへのアクセスは特定権限以上の署名を付与されたプロセスのみに制限されており、攻撃者が作成したマルウェアなどのプロセスから資格情報が含まれる領域へのアクセス試行を防ぎます。

表 3 には本節で解説した VBS と PPL の概要をまとめたものです。

Windows Downdate で示された一連の攻撃ではこの 2 つの防御機構を無効化します。
次節以降では実際に PoC を動作させながら、対象となる Windows システムがどのように変更されているかも踏まえて解説します。

表 3　VBS および PPL のコンポーネントと機能

防御機構	コンポーネント	機能
VBS	Credential Guard	資格情報を保持するプロセスのメモリ空間へのアクセスを制限 （通常の OS 空間からは暗号化された資格情報のみしか見えない）
	メモリ整合性	カーネルメモリの割り当て領域を制限 カーネルドライバーの信頼性の担保
PPL		OS の重要なプロセスへは同じように信頼された署名を持つプロセスからのみアクセスを行うように制限

検証内容

検証環境

本稿で Windows Downdate で示された攻撃を再現するために使用した環境構成を次に示します。
なお、現行の Windows クライアント OS は、初期状態では VBS や PPL が有効化されておらず、一般的にはドメイン参加時に適用されるドメインポリシーによって有効化されます。そのため、攻撃対象となる Windows クライアントシステムを、別途構築した Active Directory サーバーで運用するドメインに参加させました。

- Windows クライアント PC（攻撃対象）：
  - Windows11 23H2
- Active Directory サーバー：
  - Windows Server 2022 21H2

Update ファイルの書き換えによるダウングレード

Windows Update は図 14 に示すようにクライアントプロセスとサーバープロセスに分かれて実行されます。

具体的な Windows Update の流れは次の通りです。

① クライアントプロセスが、アップデートフォルダーに含まれているファイルの設定値を用いてサーバープロセスにアップデートの実行を要求
② サーバープロセスはアップデートフォルダーおよび同フォルダー内のアップデートファイルの整合性を検証
③ サーバープロセスはアップデートフォルダーの内容に応じてアップデート対象のファイルを決定
④ サーバープロセスがアクションリスト(アップデート対象の作成や削除、移動などを定義しているファイル)を作成
⑤ 次回再起動時に OS がアクションリストの内容を実行

Windows Downdate の発表によれば、この一連のアップデートプロセスにユーザーが介入する余地があるかという観点で調査が行われました。アップデートファイル本体については、いずれのファイルも証明書を含む整合性チェックがあるため介入の余地がなく、アクションリスト改ざんの可否が焦点となりました。

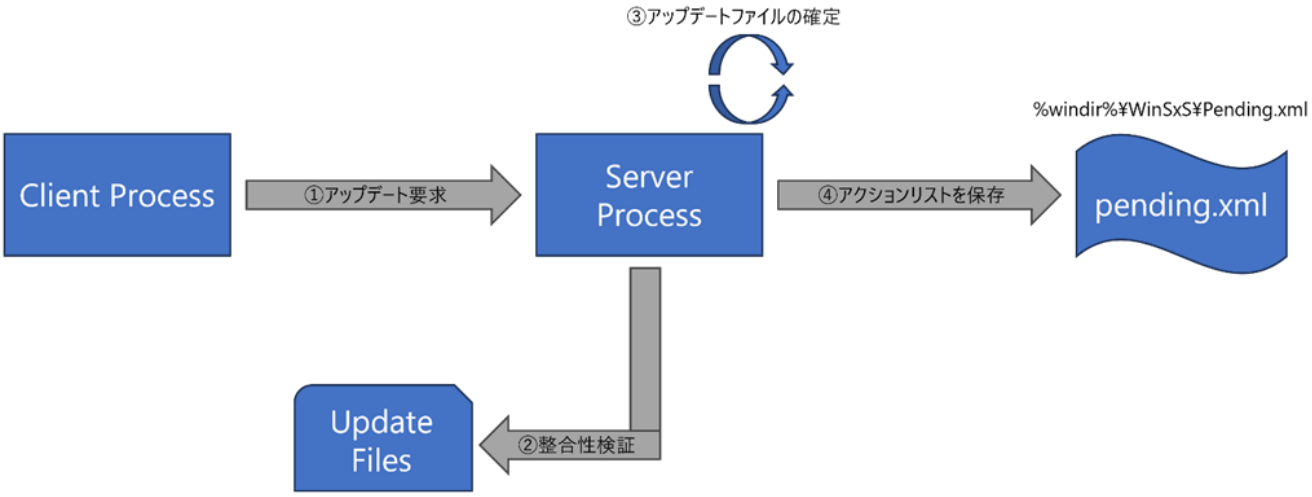

図 14　Windows Update の動作フロー

ファイルシステムに書き出されるアクションリスト（pending.xml）は Administrators グループの権限であっても編集できません。これは、Accesschk[10]を用いて実際に確認することができます。

図 15 のように、書き込み権限を付与されているのは TrustedInstaller と SYSTEM アカウントのみであることが分かります。しかし、実はレジストリにアクションリストを解釈するための設定が保存されており、該当のレジストリキー（HKLM¥software¥Microsoft¥Windows¥CurrentVersion¥SideBySide¥Configuration）は Administrators グループの権限で編集することが可能です（図 16）。

このレジストリキー配下の値「PoqexecCmdline」に設定されているデータを、任意の XML ファイルへと書き換えることで、アクションリストの内容に介入し、任意のバージョンのシステムファイルを置換することが可能となります。なお、本レジストリの書き換えや自身で作成した XML ファイルの読み込みは、前述のアップデートフォルダーの検証後に実行されるため、デジタル署名等の検証には晒されません。

図 15 pending.xml のアクセス権

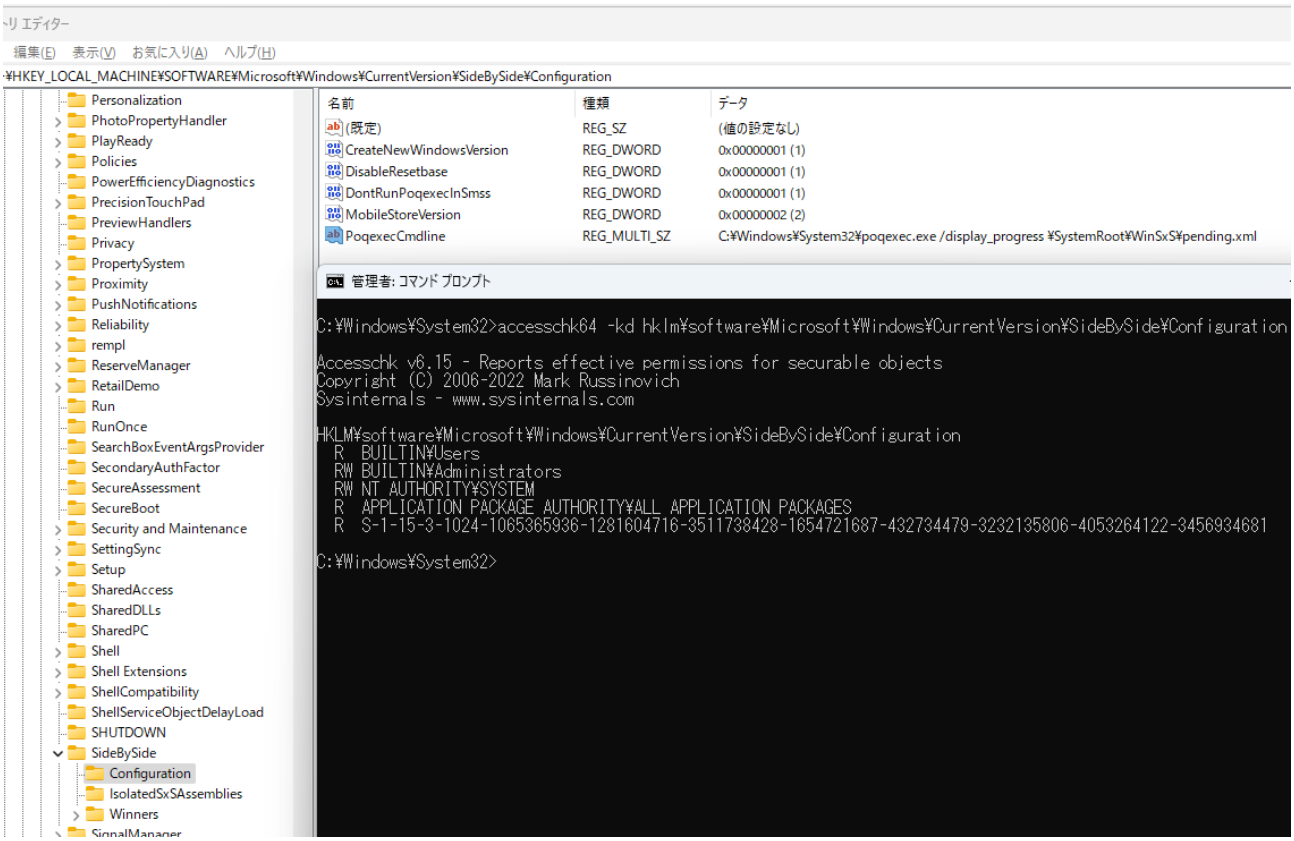

図 16 レジストリ値「PoqexecCmdline」のアクセス権

10 Sysinternals AccessChk https://learn.microsoft.com/ja-jp/sysinternals/downloads/accesschk

以降では PoC を実行しながら攻撃の流れを紹介します。

ここでは、Windows Downdate の PoC が公開されているリポジトリ[11]に example として収録されている「PPLFault Patch Downgrade」を実行することで、正規の Windows システムファイルを、脆弱性を持つバージョンへとダウングレードさせます。

最初に、PoC 実行後にバージョンが下がっていることを確認するため、事前に対象ファイルの１つである「ci.dll」のバージョンを確認しました。図 17 は Sigcheck[12]を用いてファイルのバージョン情報を確認した結果です。検証開始時における該当のドライバーファイルのバージョンは「10.0.22621.4391」でした。

次に、PoC 実行中の詳細な挙動を把握するために Process Monitor[13]を起動し、それから PoC を実行しました（図 18）。

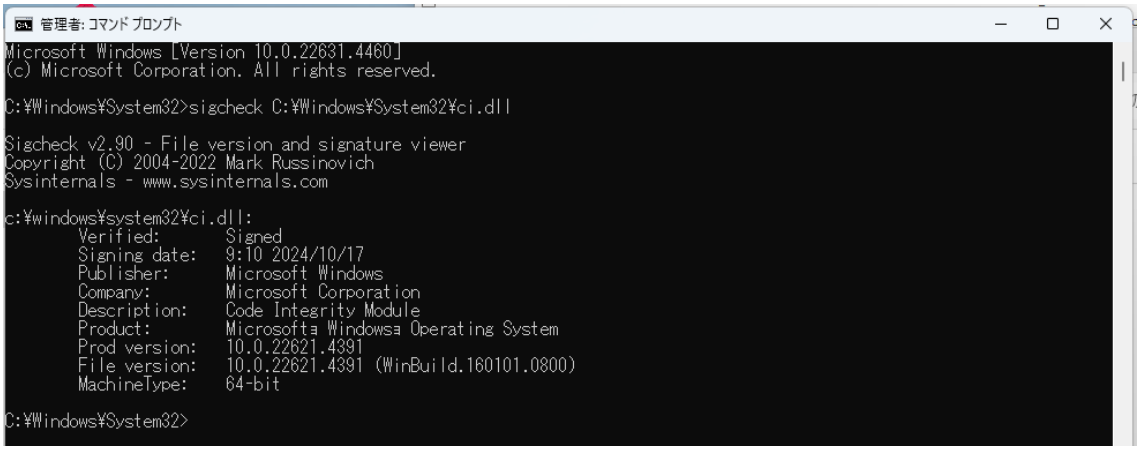

図 17　PoC 実行前の ci.dll のバージョン

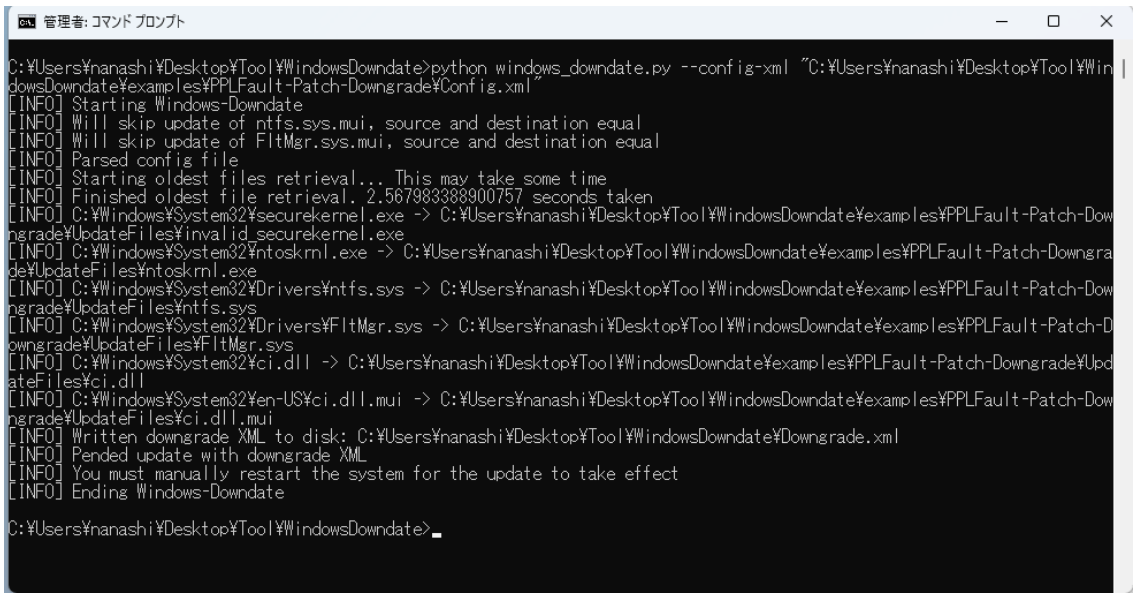

図 18　Windows Downdate の実行結果

[11] Windows Downdate（Github リポジトリ）　https://github.com/SafeBreach-Labs/WindowsDowndate

[12] Sysinternals Sigcheck　https://learn.microsoft.com/ja-jp/sysinternals/downloads/sigcheck

[13] Sysinternals Process Monitor　https://learn.microsoft.com/ja-jp/sysinternals/downloads/procmon

PoC 実行後、アクションリストを解釈するためにレジストリに設定されていた値を確認しました。図 19 の通り、「HKLM¥software¥Microsoft¥Windows¥CurrentVersion¥SideBySide¥Configuration」配下の値「PoqexecCmdline」の内容が、自身で用意した xml を参照する内容へ書き換わっています。

次に、Process Monitor のプロセスツリーを表示して PoC を実行した Python スクリプトの実行時刻と PID を確認して該当時間帯のイベントに注目します（図 20）。

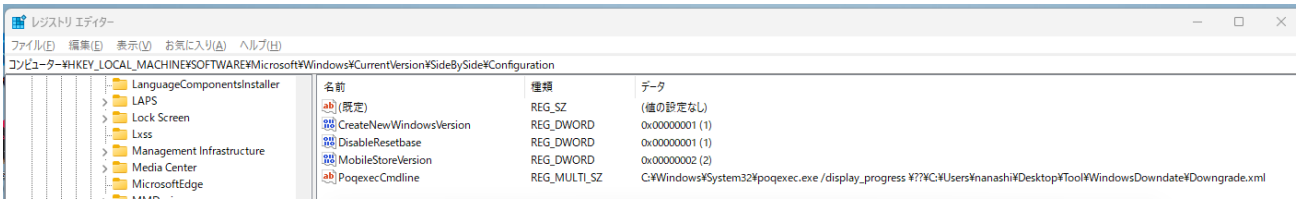

図 19 Windows Downdate 実行後の PoqexecCmdline

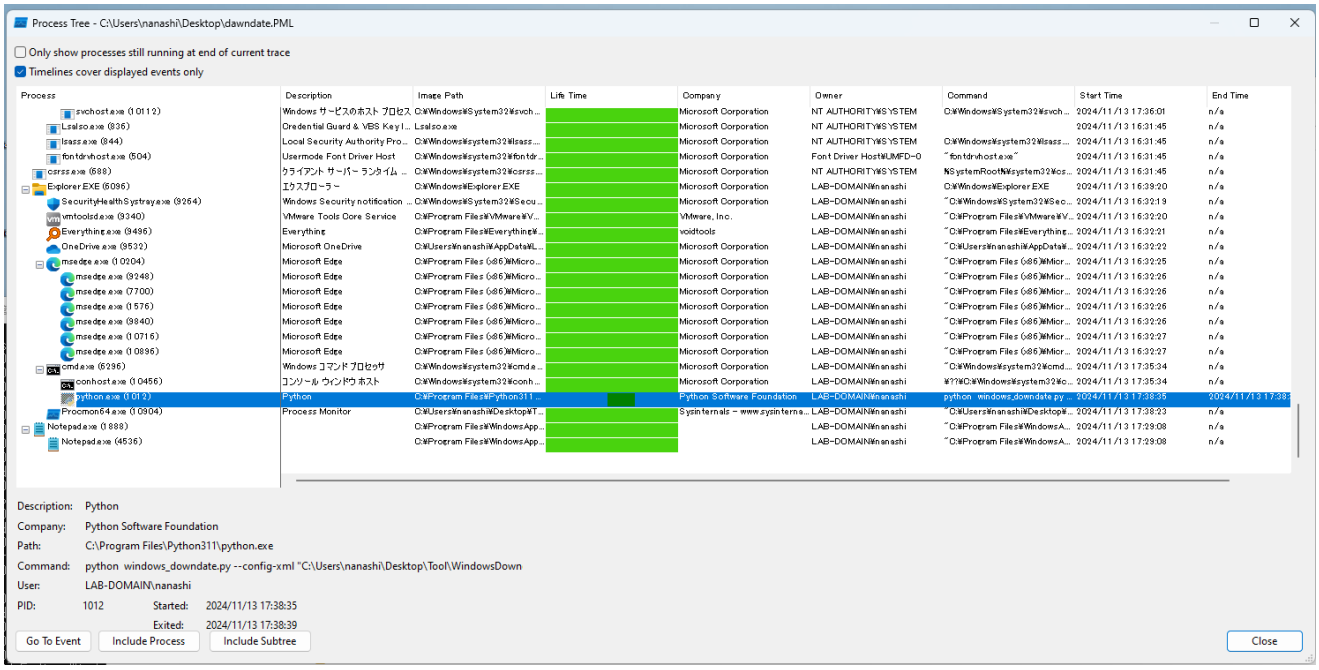

図 20 Windows Downdate 実行中のプロセスツリー

また、図 21 の通りレジストリ値を設定する所作が記録されているイ　　とが確認できました（図 22）。
ベントの詳細から、用意した XML ファイルのパスを書き込んでいるこ

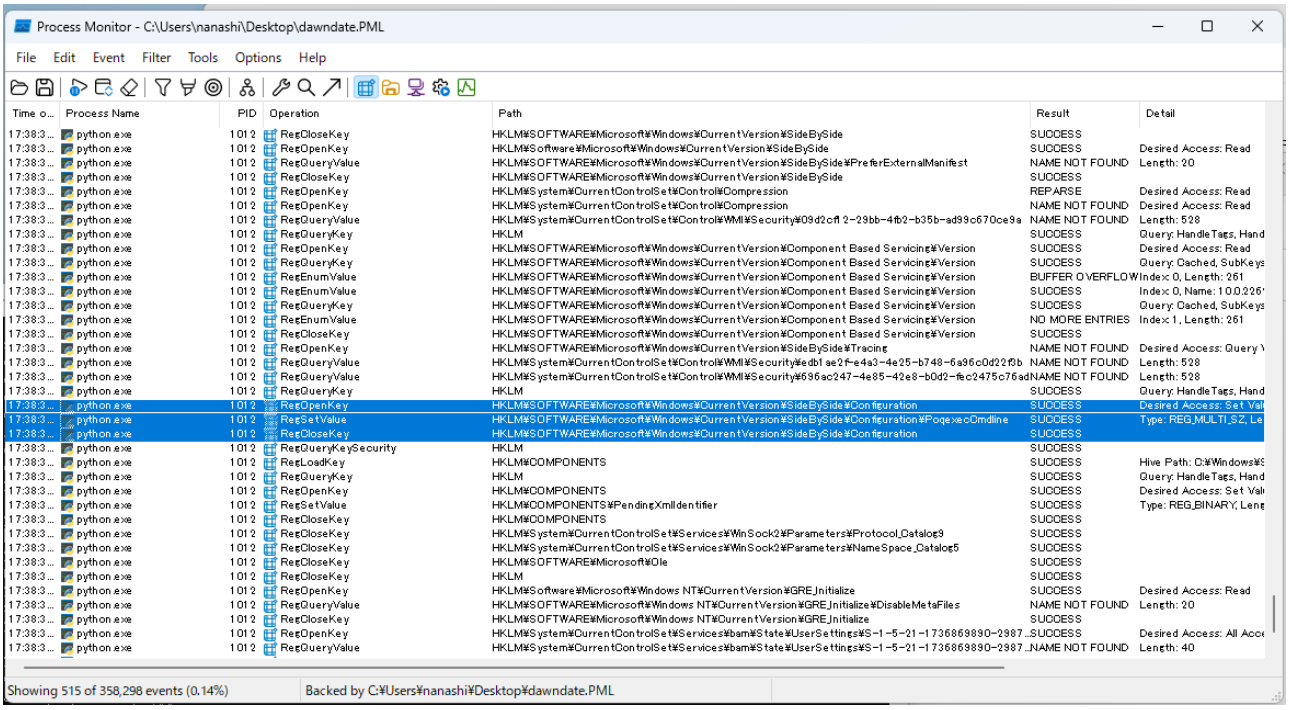

図 21　Windows Downdate 実行中のレジストリ関連イベント

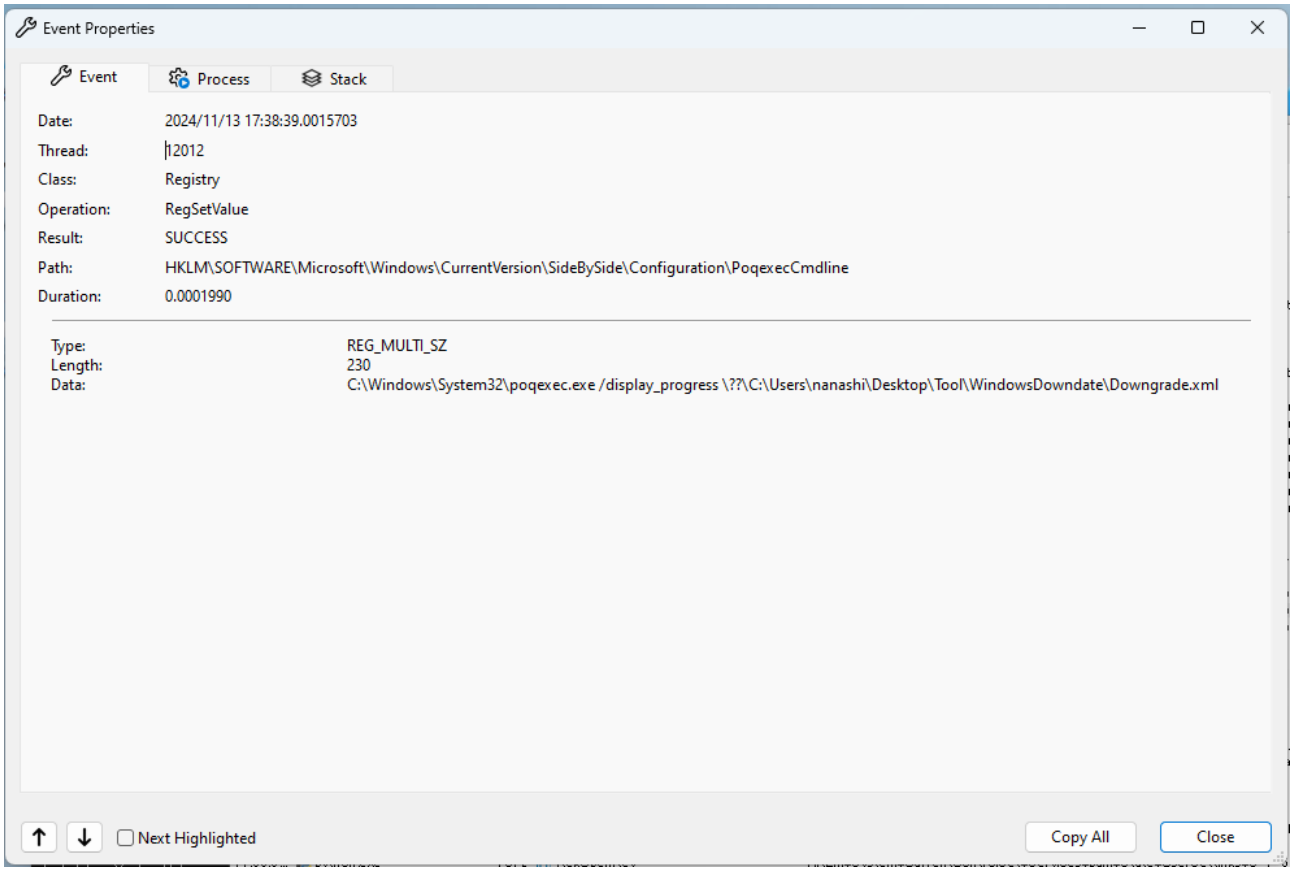

図 22　レジストリの設定を記録したイベント詳細

さらに Windows イベントログには、次回再起動時に Windows Update プロセスを開始させるために Windows Modules Installer（サービス名:TrustedInstaller）サービスを自動起動するよう設定変更したことが記録されていました（図 23）。

レジストリに保持されているアクションリストの参照先が書き換わっていることが確認できたので、最後に再起動を実施します。
再起動後、図 24 に示す通り「ci.dll」が「10.0.22621.1」までダウングレードしていることが確認できました。

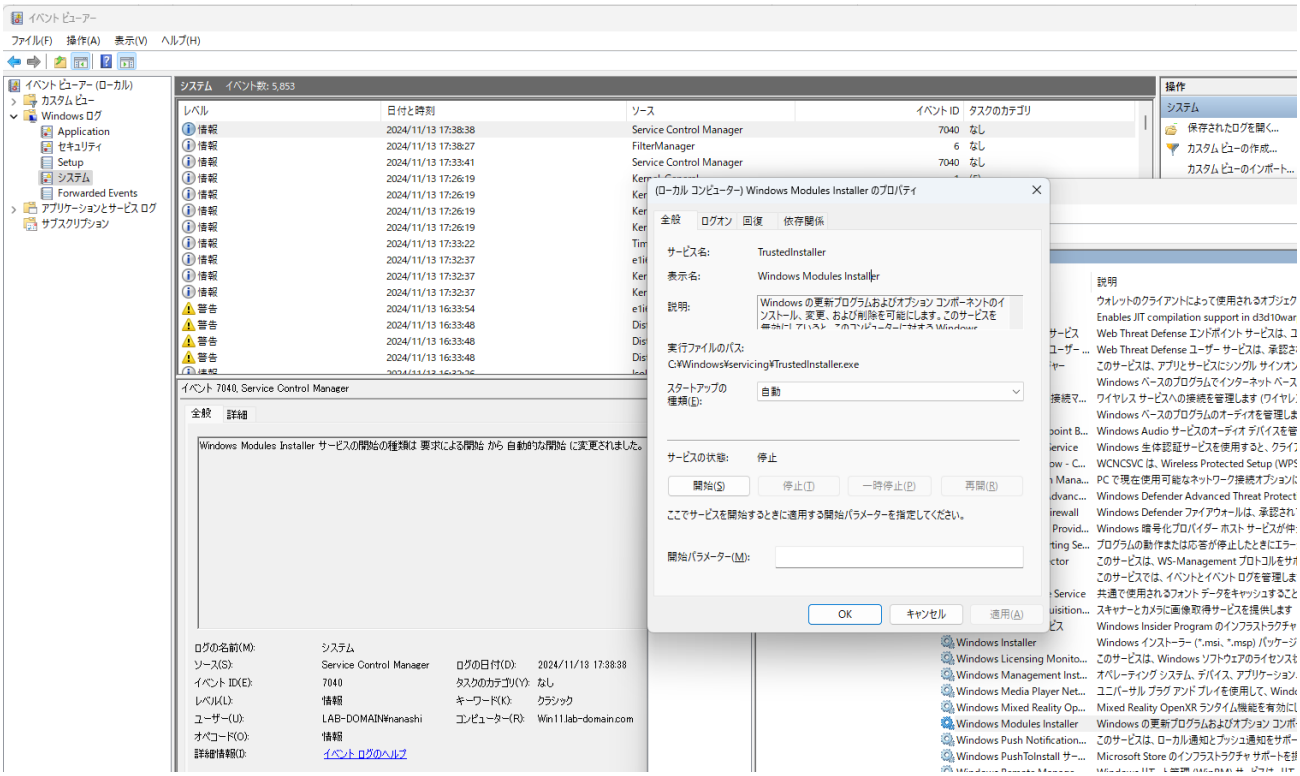

図 23　Windows Downdate 実行後に記録された Windows イベントログ

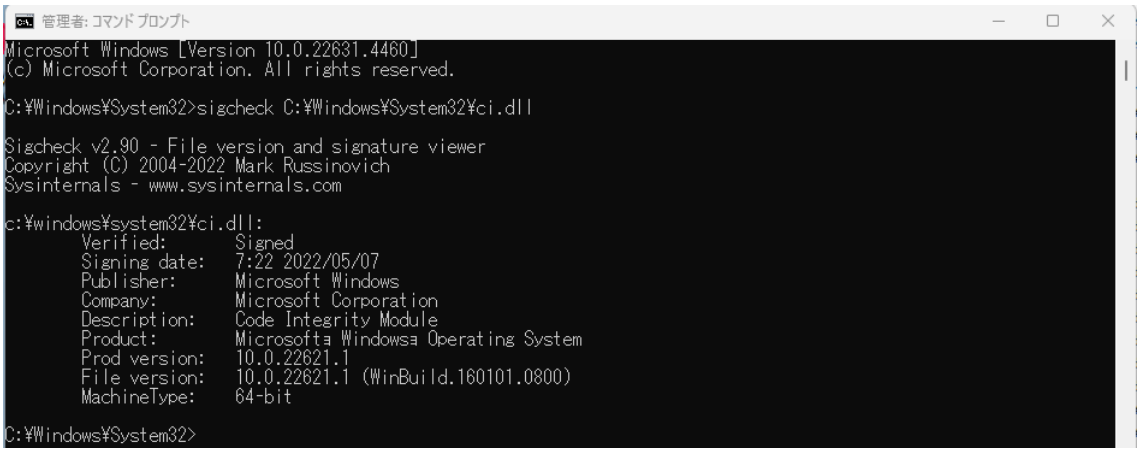

図 24　Windows Downdate 実行後に再起動した後の ci.dll のバージョン

また、Windows Update の「更新プログラムの確認」を実行すると、最新の状態である旨のステータスが表示されます（図 25）。
ここまでの検証により、Windows Update の一連のプロセスにおいて、対象とした Windows のシステムファイルを任意のバージョンへダウングレード可能であることが確認できました。

次節では、Lsass を守る防御機構である VBS と PPL の無効化について解説します。

VBS(Credential Guard) の無効化

Windows Downdate の発表では、VBS のコアコンポーネントである securekernel.exe を、前項と同様のダウングレード手法を用いて署名が付いてない無効なファイルへ置き換えた場合に、VBS 機能が無効化されると紹介されています。
この無効化についても実際に PoC を動作させて確認しました。
まず開始前に、msinfo32.exe を起動して VBS が有効化されているか確認しました。
図 26 のように「仮想化ベースのセキュリティ」が実行中であれば VBS が有効な状態であることを示しています。

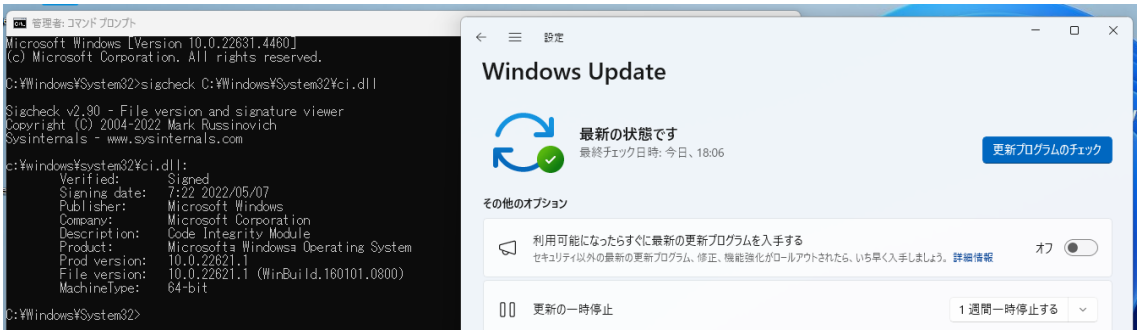

図 25　Windows Downdate 実行後に再起動した後の Windows Update のステータス

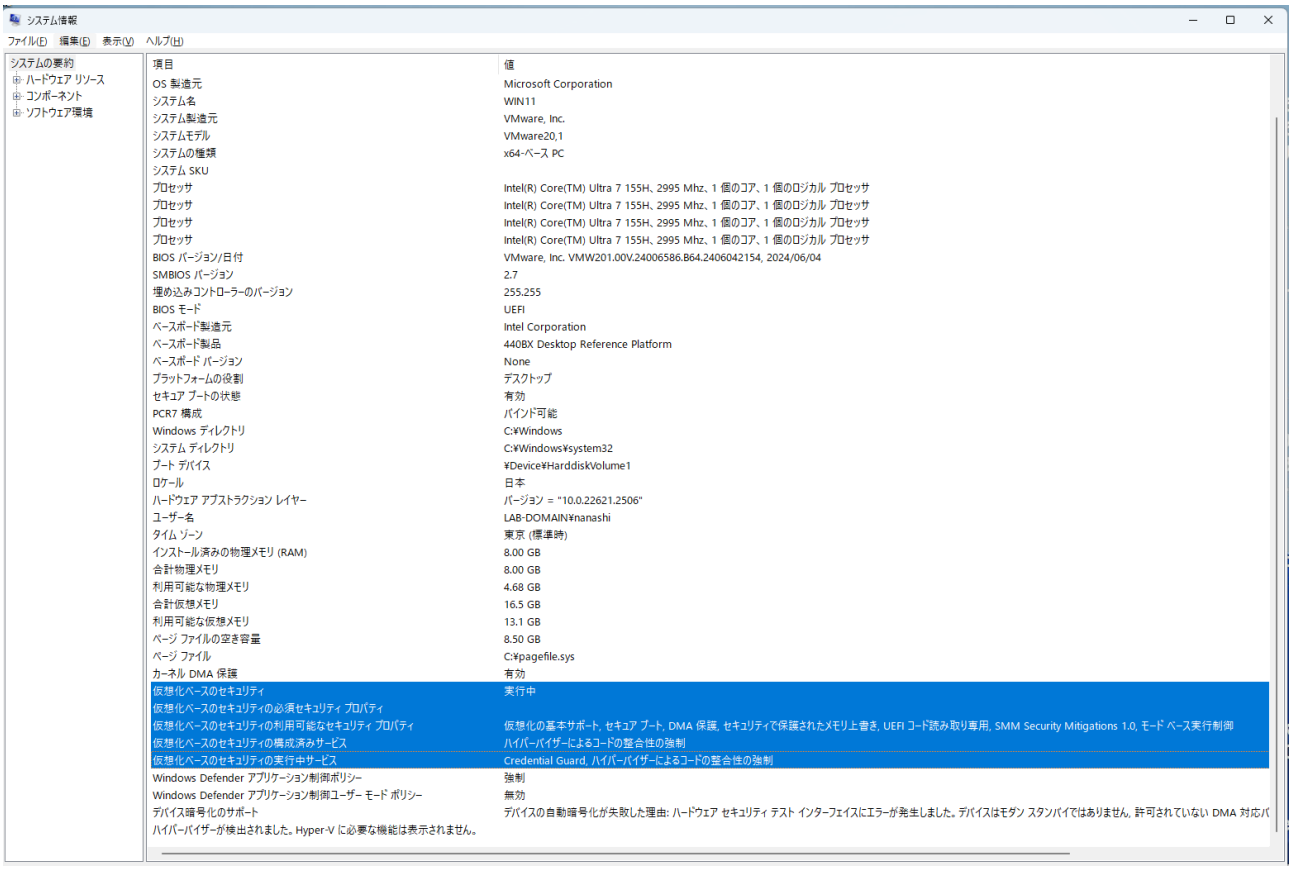

図 26　msinfo32.exe で表示される VBS のステータス

VBS が有効化されている場合は、既述の通り、VTL1 のセキュアなメモリ空間に Trustlet で資格情報を保持している Lsalso.exe というプロセスが動作しています。
このプロセスの存在は Process Explorer[14]などで確認することができますが、通常の OS 空間からアクセスできないため、詳細な情報は表示されません。

また、Lsass.exe プロセスは通常のメモリ空間に存在するものの、こちらは PPL で保護されているため、Administrators グループ権限を有していてもアクセスすることはできません。図 27 は Process Explorer で表示される Lsalso.exe と Lsass.exe のプロパティ情報を並べて表示したものです。

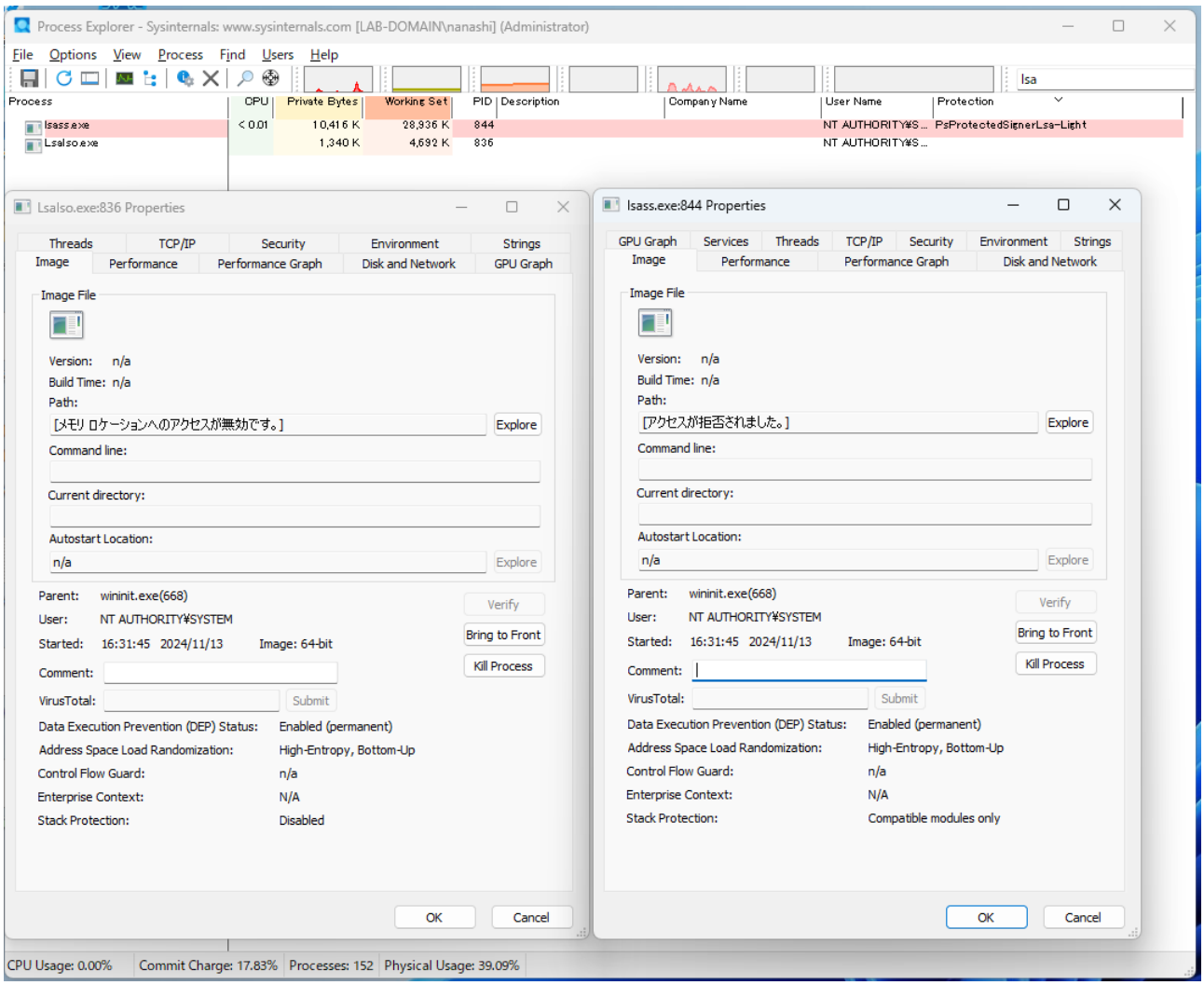

図 27　Lsalso の（左）と Lsass（右）のプロパティ情報

[14] Sysinternals Process Explorer　https://learn.microsoft.com/ja-jp/sysinternals/downloads/process-explorer

PoC（前節で動作させたものと同じ PPLFault の example）を動作させて確認する前に、置き換え対象のシステムファイル securekernel.exe を確認しました。

図 28 の左側が置き換えに用いるデジタル署名が無効な invalid_securekernel.exe、右側が置き換え前の正規の securekernel.exe です。

両者ともデジタル署名を保持しているものの、左側の置き換え用ファイルは署名が破損または変更されて無効な状態になっていることが分かります。

ここまでの確認を行った後、前項と同様に PoC を実行し、アクションリストのパスを保持するレジストリキーの値が書き換わっていることを確認したうえでシステムを再起動しました（図 29）。

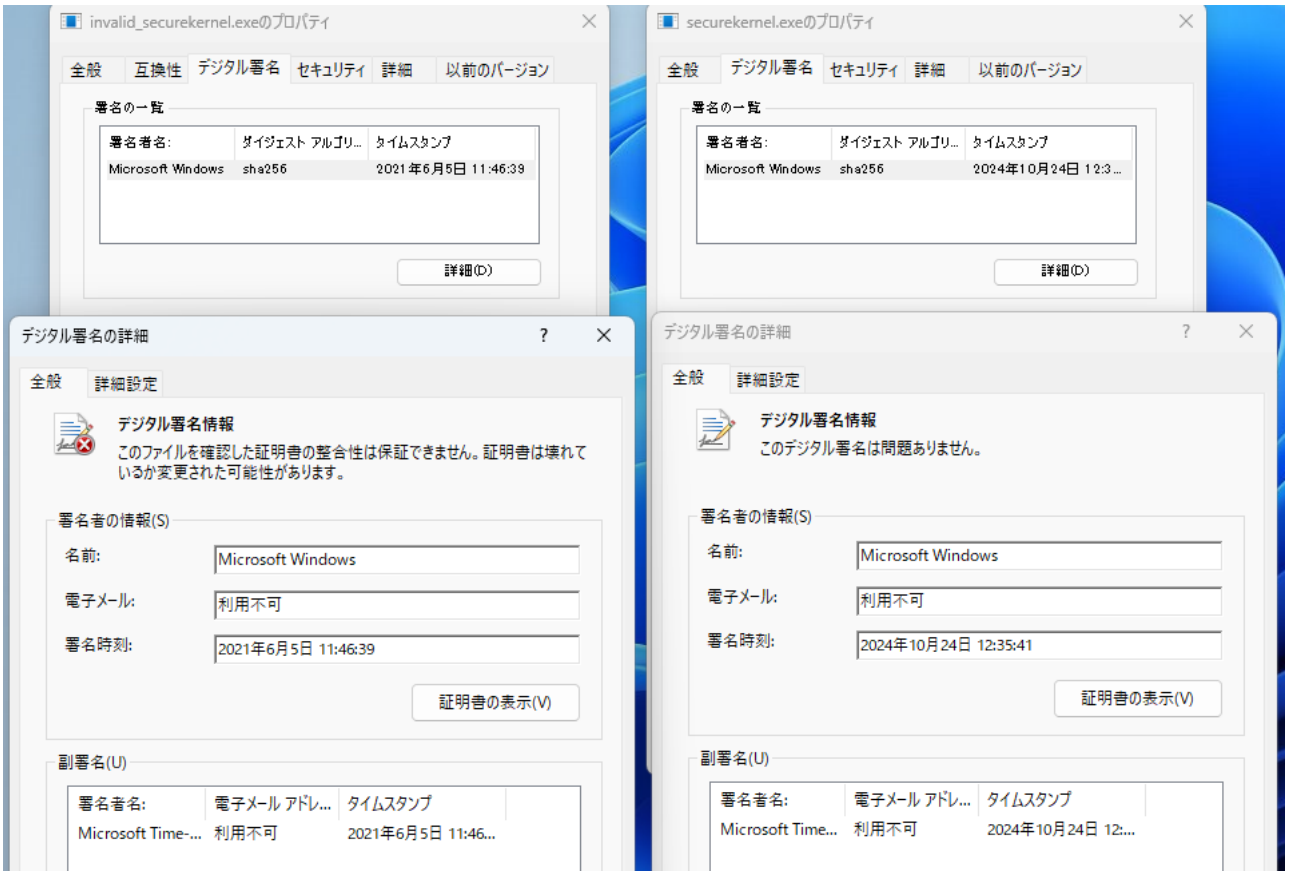

図 28　置き換え用のファイル（左）と置き換え前の正規ファイル（右）

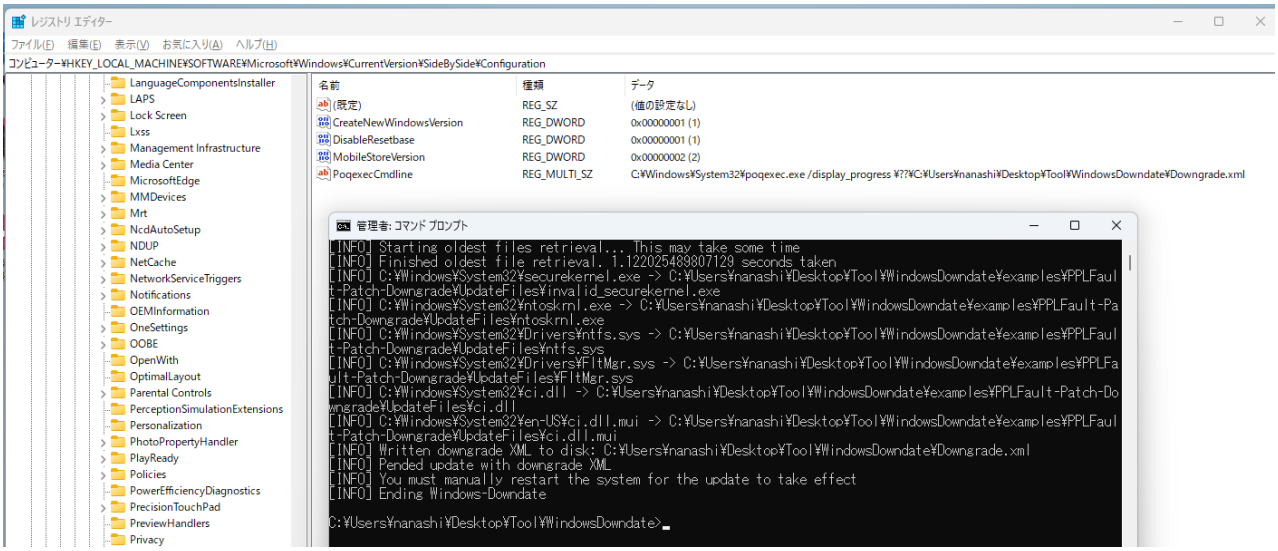

図 29　Windows Downdate 実行後の PoqexecCmdline

図 30 は、再起動後に C:¥Windows¥System32 配下の securekernel.exe が、証明書の無効な実行ファイルへ置き換わっていることを確認したものです。

また、図 31 は msinfo32.exe の表示から VBS が無効化されていることを示すものです。

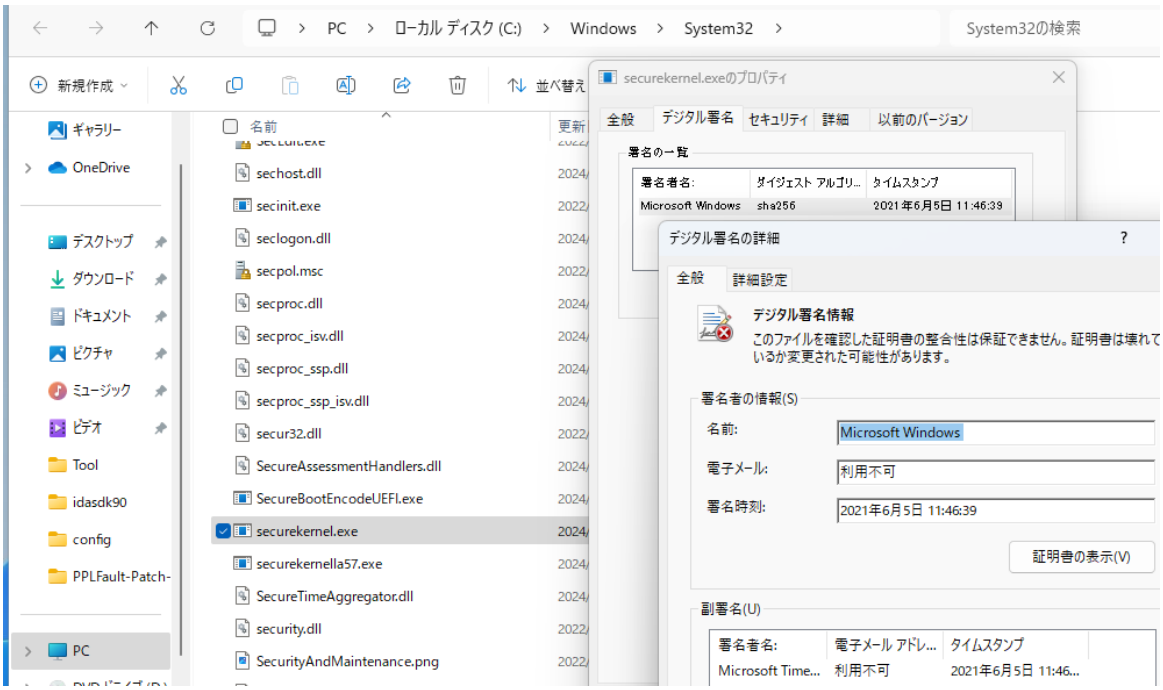

図 30 Windows Downdate 実行後の再起動後に置き換わった securekernel.exe

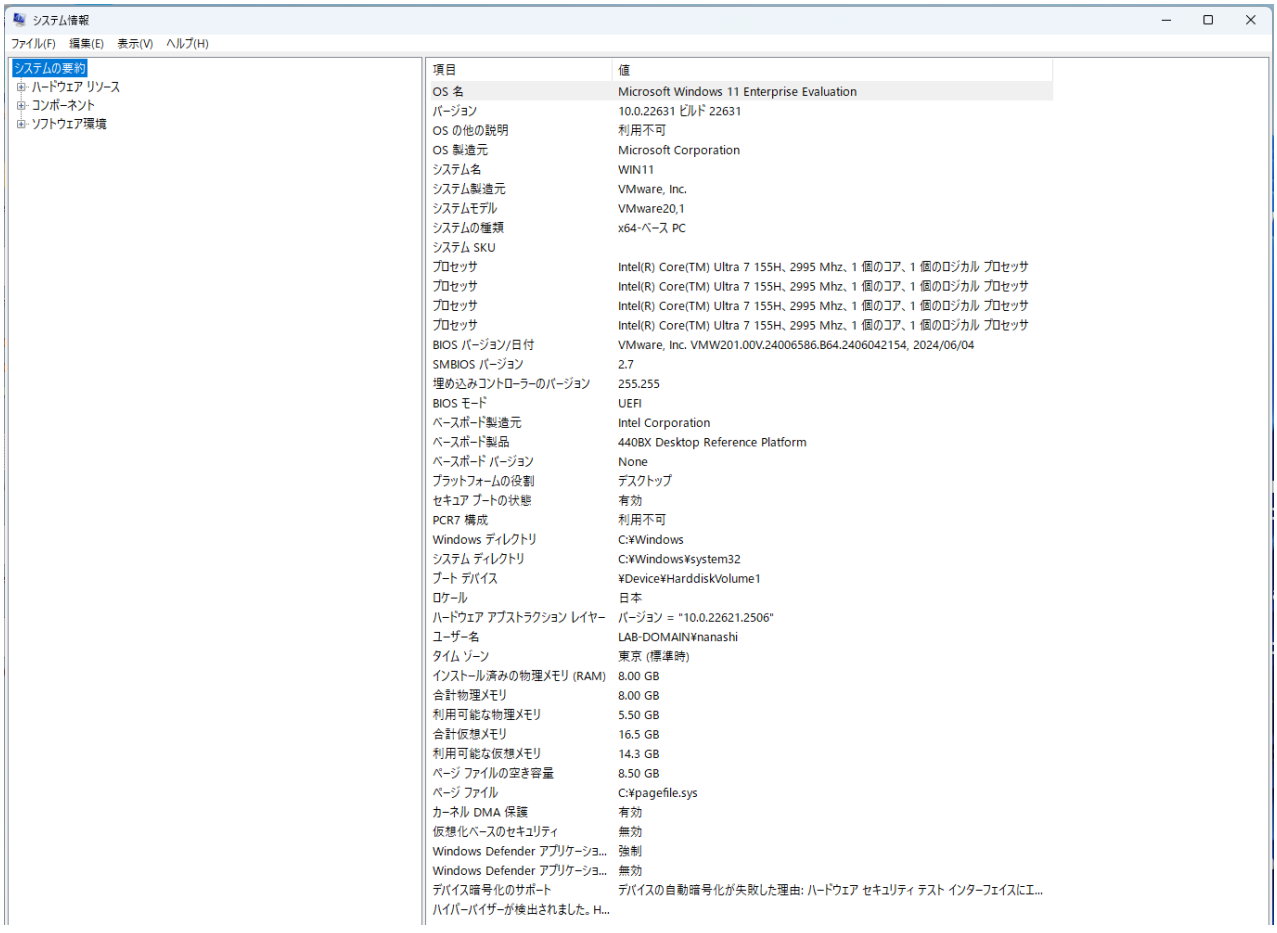

図 31 Windows Downdate 実行後の再起動で無効化された VBS のステータス

ここまでの手順により VBS が無効化されたため、VTL によって管理されるセキュアなメモリ空間は構成されず、LsaIso.exe プロセスも実行されません（図 32）。
この結果から、当該 PoC によって securekernel.exe を証明書が無効なバージョンへ置き換えることで VBS が無効化されることが確認できました。
なお、この PoC によって悪用される脆弱性が CVE-2024-21302 と CVE-2024-38202 に該当します。

次節では、ここまでの手順によりダウングレードされ、既知の脆弱性が再現可能になった検証環境で、実際に PPL をバイパスして Lsass プロセスへのアクセスを試みます。

PPL のバイパス（PPLFault）

PoC に採用されている PPL のバイパス手法は 2023 年に Black Hat Asia で発表された PPLFault という手法[15]の実装です。
本手法は PPL の中でも最も高いレベルの署名である WinTcb を持つ services.exe に対して任意のペイロードを持つ DLL ファイルを読み込ませることで、Lsass プロセスのメモリダンプを実行させるものです。
PPL プロセスの実行時にはプロセス本体だけでなく、読み込む DLL ファイルについても特別な署名を持つファイルのみが許可されます。しかし本手法では services.exe が呼び出す特定の DLL ファイル（EventAggregation.dll）がメモリ上にマッピングされた後のページング動作を悪用することで、任意のペイロードを読み込ませます。
通常、プログラム署名の検証はファイルを読み込む際に実行され、ページング処理に伴う再読み込み時には行われません。本手法のように検証後に検証済みのデータを改ざんするなどして実行させる手法は TOCTOU(Time of check to time of use)と呼ばれます。

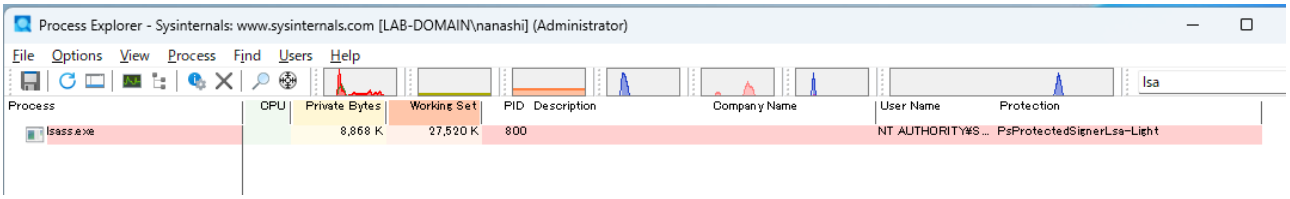

図 32　VBS が無効化され LsaIso.exe が実行されず Lsass のみが起動しているステータス

[15] PPLFault（Github リポジトリ）　https://github.com/gabriellandau/PPLFault

図 33 に示した PPLFault の実行手順は次の通りです。

① CloudFilter という Windows API を用いて、コールバック関数を設定した空のプレースホルダーファイルを SMB アクセス可能なパスに設定します。

② 前項のプレースホルダーから EventAggregation.dll に SMB 経由のパスへシンボリックリンクを作成します。

③ EventAggregation.dll が呼び出された後に呼びだされる devobj.dll に oplock を設定し、services.exe が EventAggregation.dll を読み込んだ後の処理へ進めないようにします。

④ services.exe を起動します。

⑤ services.exe が EventAggregation.dll の読み込みを開始します。

⑥ シンボリックリンク参照により SMB 経由でコールバックが呼び出されます。ここでは正規の EventAggregation.dll を返します。

- これにより読み込まれる DLL ファイルの署名検証が完了します。
- その後、Page Fault を発生させるために services.exe プロセスの working set[16]と standby list[17]を空にします。
- oplock の解除を行い services.exe が後続の処理へ進めるようにします。

⑦ services.exe では working set が空なので、EventAggregation.dll のページファイル参照時にも Page Fault が発生してページングが実行されます。

⑧ ページングにより再度コールバックが呼び出され、EventAggregation.dll の読み込みが要求されます。

- この時、特定のプロセスをダンプするようパッチを適用した DLL ファイルを返します。既に署名の検証は完了しているため、この DLL ファイルに関する検証は行われません。

⑨ 最終的に WinTcb 署名を受けている services.exe のプロセスでカスタム DLL のペイロードが実行され、任意のプロセスのメモリダンプを行います。

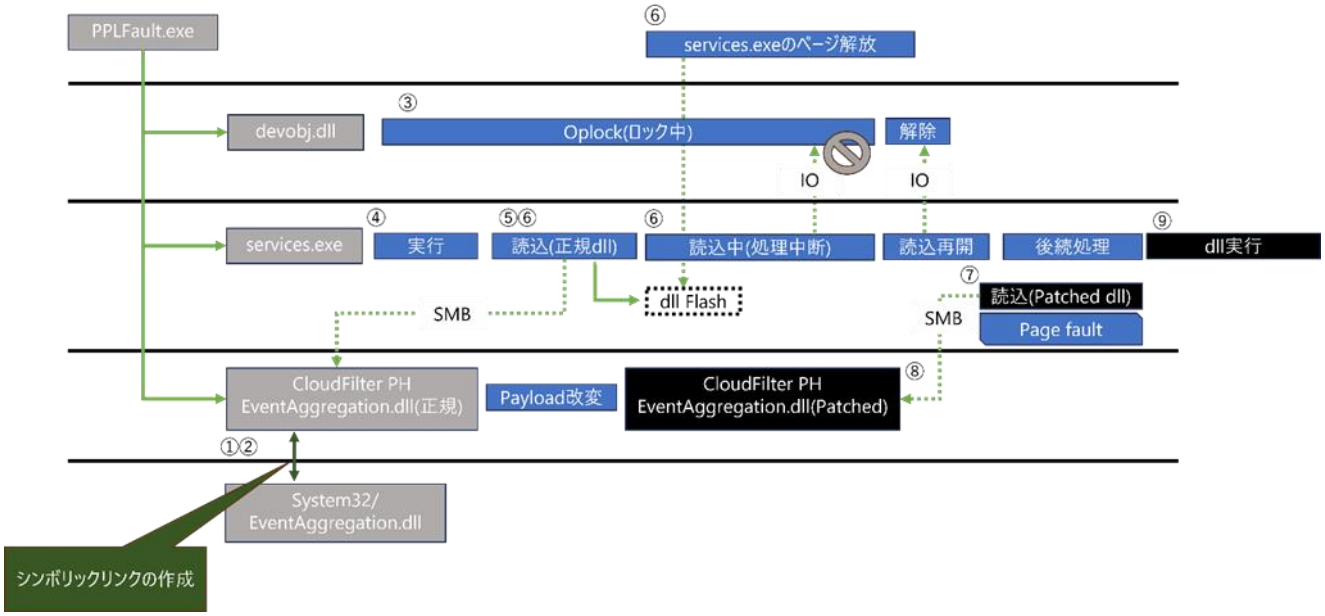

図 33　PPLFault の概略図

[16] プロセスメモリにおいて、物理メモリ上マッピングされているメモリページの集合を Working set と呼びます。

[17] アクセス頻度が低いメモリページを一時的に退避して保持しておくページの集合を Standby list と呼びます。

前述の手順を踏まえて PPLFault の PoC を動作させます。
まず動作前に Process Monitor を起動させ、プロセスイベントのキャプチャを有効にしておきます。
実行後、図 34 のように Lsass プロセスでは PPL が有効であるにもかかわらず、プロセスメモリのダンプファイル（.dmp）が作成されていることを確認できました。
また、Process Monitor を見ると、PPLfault.exe から services.exe が生成されていたことが分かります（図 35）。

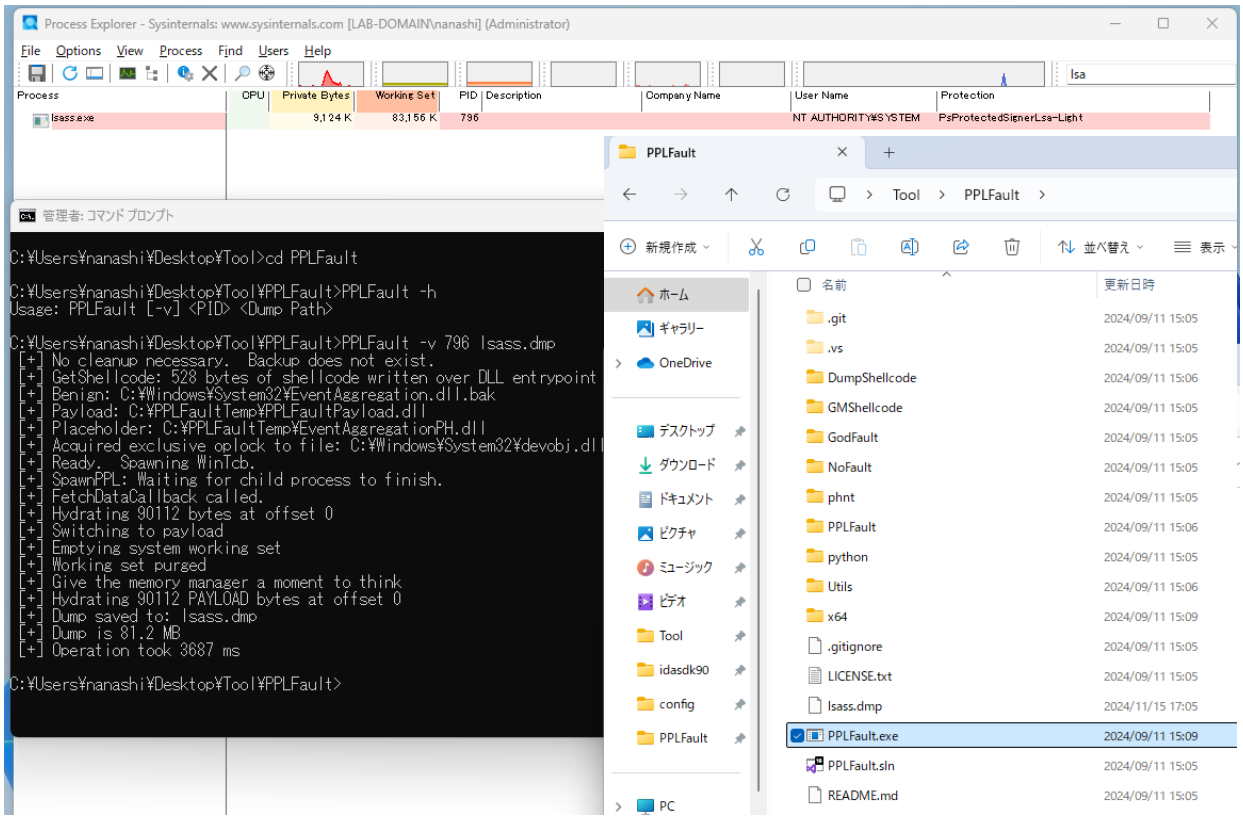

図 34　PPLFault の実行結果

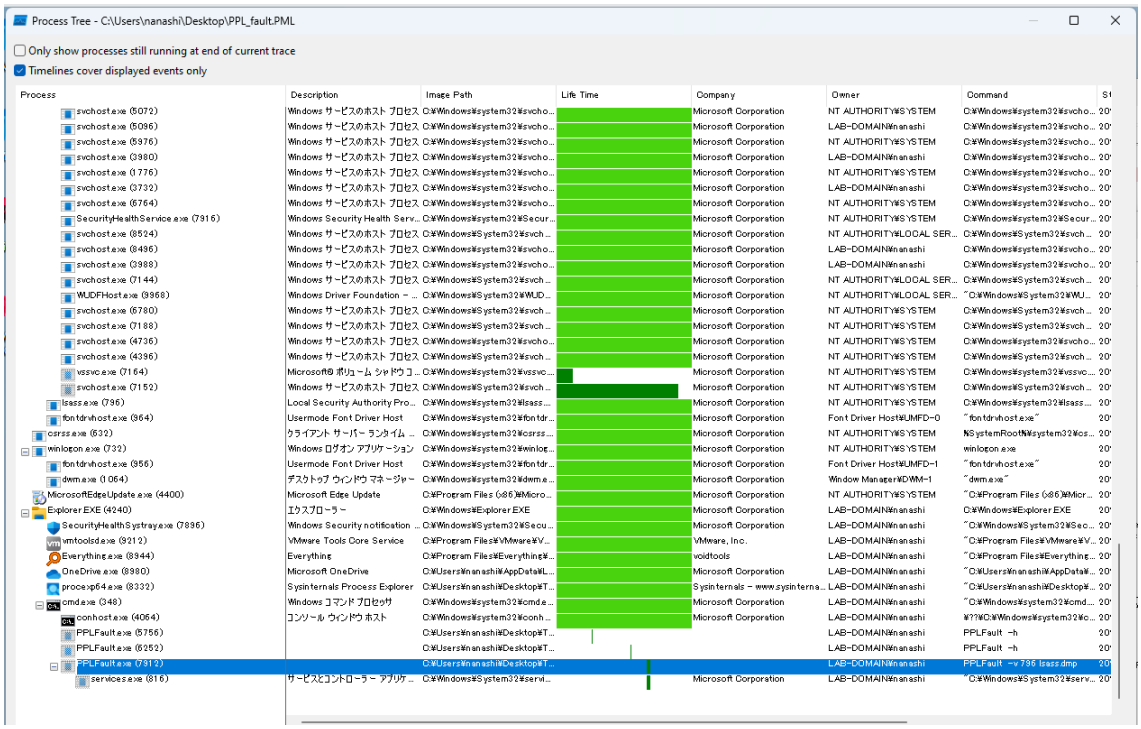

図 35　PPLFault 実行中のプロセスツリー

Process Monitor では、DLL ファイルに対して SMB 経由でアクセスをさせている様子やページングを行う時間を稼ぐために oplock を設定してプロセスを遅延させている様子なども確認することができます（図 36）。
ここまでの手順により、PPLFault を用いて Lsass プロセスのメモリダンプを取得しました。

クレデンシャルダンプ（Mimikatz）

実際に出力した Lsass プロセスのメモリダンプを、Mimikatz を用いて解析することで、資格情報を取得できることを確認しました（図 37）。

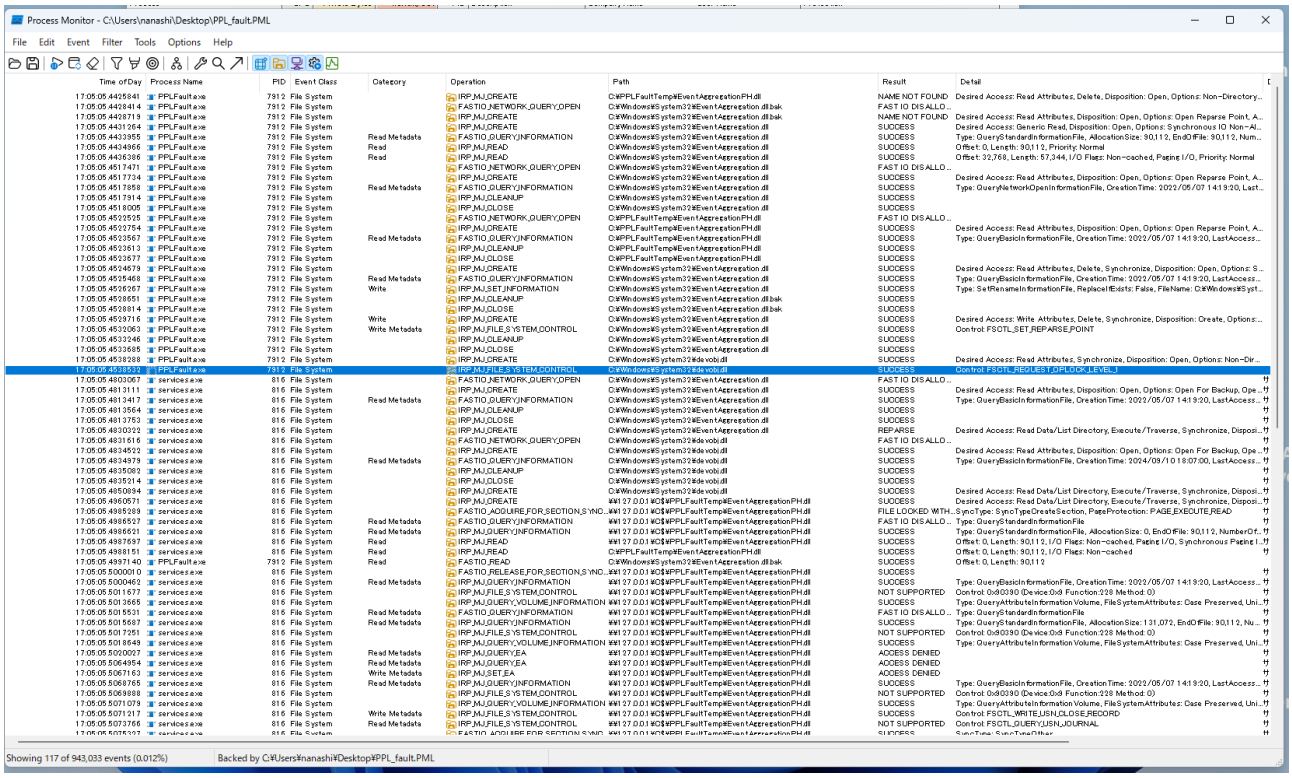

図 36　PoC 実行中に Process Monitor で記録したイベント

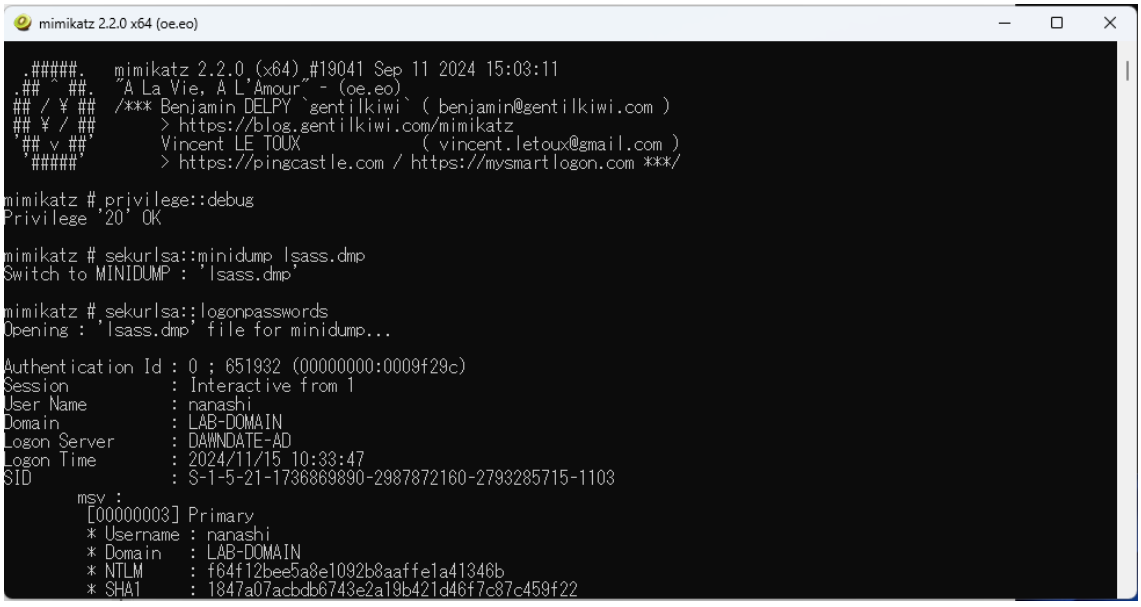

図 37　メモリダンプファイルを Mimikatz で解析した結果

また、Windows Downdate の公開内容にも含まれていた通り、VBS と PPL どちらかのみが有効化されていた場合には、いずれかの防御機構に阻まれ、資格情報奪取の試みが成功しないことも別途確認しました。

VBS のみが有効化されている場合

VBS のみが有効化されている場合には、Credential Guard に妨げられ、生の資格情報にアクセスすることはできません。Mimikatz を実行すると、資格情報は暗号化されたまま出力されます（図 38）。

PPL のみが有効化されている場合

PPL のみが有効化されている場合には、管理者権限であっても適切な署名がないプロセスから Lsass へアクセスすることはできません（図 39）。

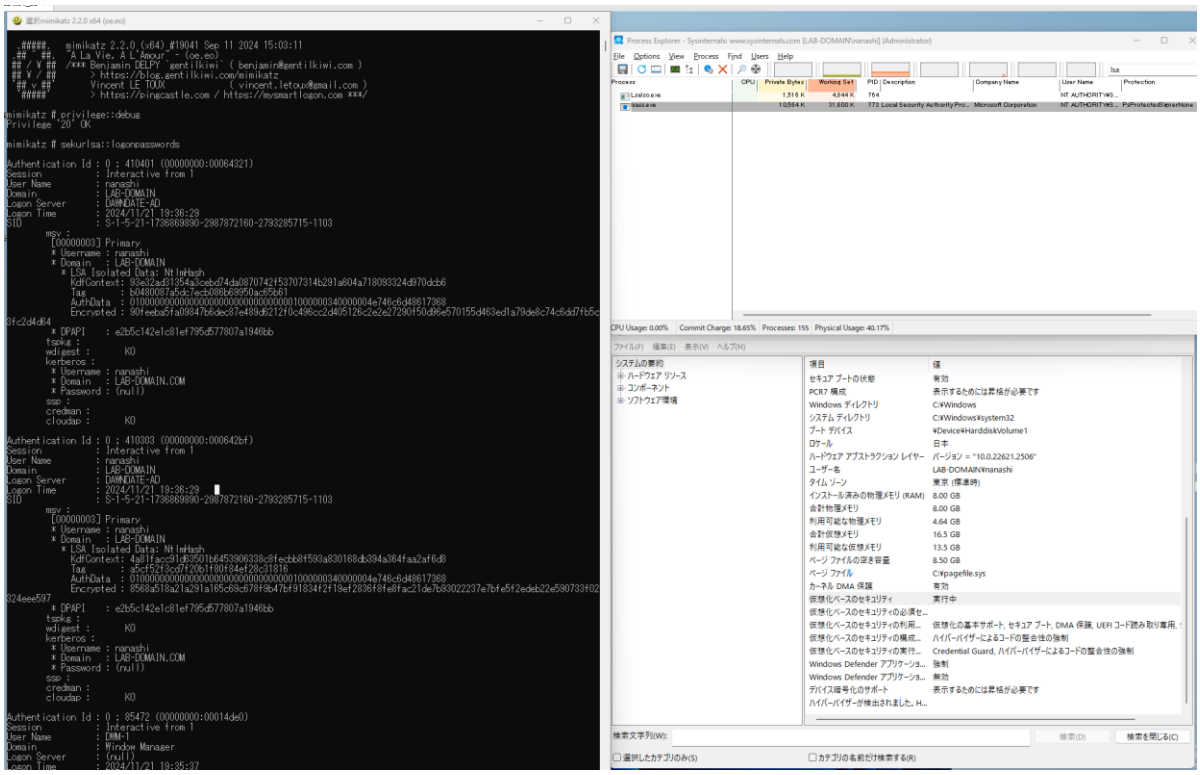

図 38　VBS のみが有効な状態で Mimikatz を実行した結果

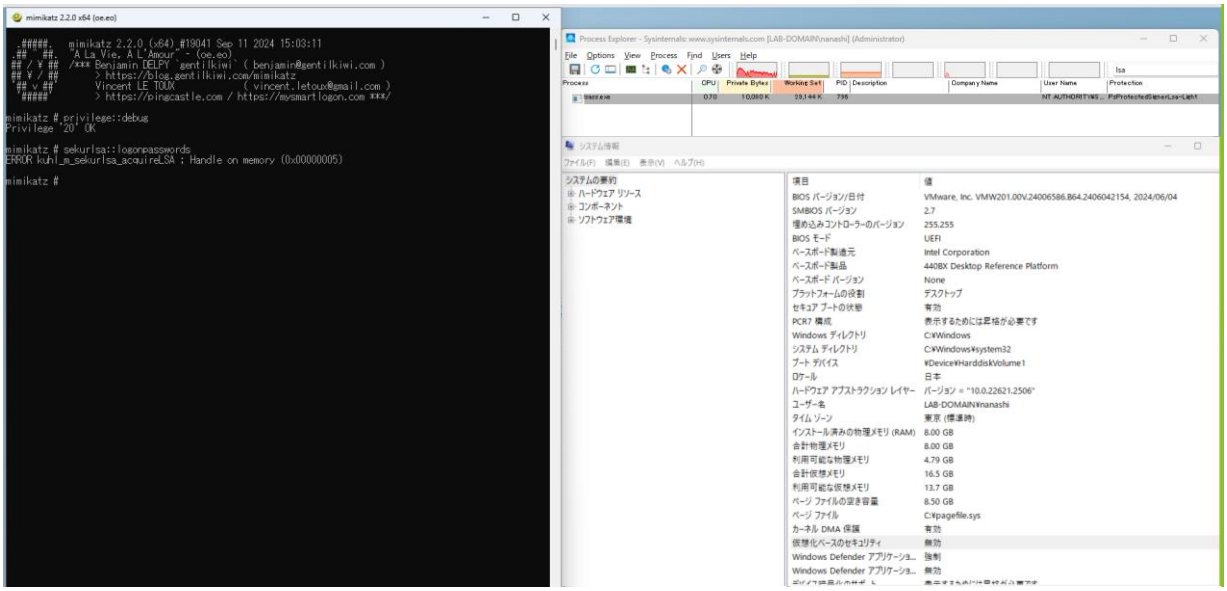

図 39　PPL のみ有効な状態で Mimikatz を実行した結果

Windows Downdate で残る痕跡

前節まで紹介してきた PoC の実行に伴い、対象となった Windows システムに残される痕跡の情報を表 4 にまとめました。本件の攻撃手法が悪用された場合、アクションリストのレジストリエントリには明確な証拠が残ります。さらにレジストリアクセスやファイルアクセスの監査を有効にしてイベントログを外部に転送するようにすることで、攻撃者が資格情報取得後に行う他の所作に関する情報を収集したり、記録の消去や改ざんを回避したりすることが可能です。

表 4　一連の PoC を実行して確認した痕跡

痕跡の残る場所	残るタイミング	残る痕跡	
レジストリ	Window Downdate のアップデートプロセス書き換え	Registry key	HKLM¥software¥Microsoft¥Windows¥CurrentVersion¥SideBySide¥Configuration
		Value Name	PoqexecCmdline
		Value Data	C:¥Windows¥System32¥poqexec.exe /display_progress<任意の XML path>
Windows Event	Window Downdate のアップデートプロセス書き換え	Event ID	7040
		説明	Windows Modules Installer サービスの開始の種類は 要求による開始 から 自動的な開始 に変更されました。
プロセスツリー	PPLFault の PoC 実行	Wininit.exe 以外の親プロセスからの service.exe の実行	

Windows の緩和策

現在 Microsoft から提供されている緩和策[18]では、脆弱な VBS システムファイルを取り消す「Microsoft が署名した失効ポリシー」の適用が挙げられています。これを適用することで、更新されていない VBS システムファイルが OS に読み込まれなくなることが案内されています。

本緩和策のポリシーは新しいバージョンの Windows では 2024/8/13 の更新プログラムで導入されていると記載されています。確認のため Windows11 24H2 で PoC を試したところ、VBS の無効化については再現できましたが、その後のシステムファイルのダウングレードを試みるとブルースクリーンとなり、ブートループが発生しました。これは Windows の緩和策が有効化されている影響によるものと考えられます。

Windows のアップデートプロセスへの介入そのものについては未修正である点や、VBS の無効化そのものは可能な状況であるため、本攻撃手法に関連した派生の侵害手法が悪用されるようになる可能性が考えられます。2024 年 12 月までに本脆弱性が実際の攻撃に用いられたという報告は確認されていませんが、本手法が根本的に無効化されたわけではありません。

技術検証の重要性

本件のような根本的に対策の難しい脆弱性やその攻撃手法が、セキュリティベンダーや研究者によって公開されることは少なくありません。もちろん、通常はサポートベンダーによって何らかの緩和策が提供されます。しかし、そういった緩和策の有効性や攻撃の成立条件、また、攻撃が成立した場合の影響や後続の挙動の追跡可否など、具体的な情報が公開されることはほとんどありません。

このような脅威に適切に対応するためには、本稿で示したような検証、そして検証を踏まえた対策を検討する体制を維持することが必要です。セキュリティ専業ではない事業会社などの組織でそういった体制の維持が現実的ではない場合は、その役割を補完する相談先を確保しておくことを推奨します。

参考文献 :

Pavel Yosifovic 他著「インサイド Windows 第 7 版 上 / 下」日経 BP 社　2018 年

[18]仮想化ベースのセキュリティ（VBS）関連のセキュリティ更新プログラムのロールバックをブロックするためのガイダンス　https://support.microsoft.com/kb/5042562/ja-jp

EDR 回避手法の検証

はじめに

企業環境への EDR（Endpoint Detection & Response）製品の導入が進むにつれ、EDR が導入された環境を前提とするような攻撃手法も報告されるようになりました。
本稿では、実際に EDR を導入した検証環境を用いて EDR 回避を企図した攻撃手法の有効性を検証し、その結果を踏まえて実効的な対応策を紹介します。
なお、EDR は稼働環境や運用方針などにより検知精度が大きく異なる場合があります。本稿の検証結果はあくまで前述の検証環境における内容であることをご了承ください。また、安易な悪用を防ぐために攻撃手法の詳細は省略しています。

EDR が分析対象とするイベント

EDR は OS 上で発生する様々なイベントを監視することで脅威を検知します。具体的には、次のようなイベントが EDR の監視対象として一般的です。

■プロセス生成/終了
プロセスにまつわる情報を収集し、不審なプロセスの生成や、本来発生しえないような子プロセスの生成などのイベントを中心に監視します。

■レジストリアクセス
レジストリには OS やプログラムにおける設定情報が格納されています。プログラムの自動起動に関するレジストリキーなど、攻撃者に悪用されやすいエントリを中心に監視します。

■ファイルアクセス
ファイルの書き込みや削除などのイベントを監視します。アラートの対象としてだけでなく、インシデント発生時の分析にもファイルアクセスイベントが利用されます。

■ネットワークアクセス
ホストのネットワークアクセスを監視しています。プロセスごとの接続先が記録されたイベントは、ファイルアクセス同様インシデント発生時の分析にも有用です。

■API コール状況
API がどのように利用されているかを監視しています。API を監視することで、そのプログラムが OS のどのような機能を使ったかを分析することができます。

■認証イベント
システムへのサインイン/サインアウトなどの認証イベントを監視しています。

EDR は、これらの OS のイベントを監視することで、従来のアンチウイルスのようなファイルへのパターンマッチをベースとする検知から大きく監視対象を広げました。

次節では、実際の EDR 製品が一般的な攻撃手法をどの程度検知可能か検証した結果を紹介します。

基本的な EDR 回避手法の検証

検証にあたって

本稿では、一般的な攻撃の初期侵入フェーズを想定したマルウェア実行手法を検証しました。具体的には、攻撃者が何らかの方法で攻撃対象のユーザーにスクリプトファイル等を配布し、当該ユーザーが誤認等によってそのファイルを実行することで、結果的に攻撃者の意図したマルウェアが業務端末上で実行させられてしまうような局面を対象としています。図 40 は実行イメージを示しています。

なお、本検証では実行対象の「マルウェア」として、Havoc という C2 フレームワーク[19]のエージェントを選択しました。

スクリプトファイル等を経由することで EDR の検知を回避してエージェントを実行し、Havoc サーバーとの C2 通信を確立することができれば回避成功とします。

検証結果

検証結果は表 5 の通りです。製品 A および製品 B それぞれに検知できなかった攻撃はあるものの、大部分の攻撃を検知することができました。

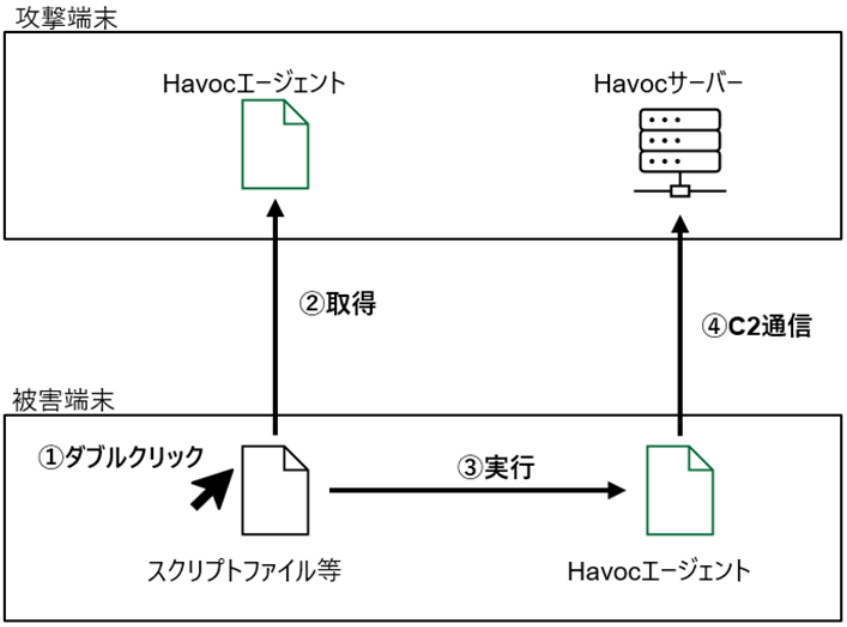

図 40　検証イメージ

表 5　検証結果概要

エージェント実行手法	製品 A	製品 B
PowerShell	△	△
VBScript	×	○
Windows Command Shell (cmd.exe)	○	○
MSHTA	○	×

[19] Havoc（Github リポジトリ）　https://github.com/HavocFramework/Havoc

PowerShell

PowerShell は Windows に標準インストールされており、多様な機能を有しているために攻撃者によって悪用されやすいツールです。
今回の検証では、Havoc エージェントをダウンロードし、特定のフォルダーに保存したうえで実行する PowerShell スクリプトを.ps1 ファイルとして保存し、ダブルクリックで実行しました。実行時、画面上は青いウィンドウが表示されただけで何も起こりませんでした（図 41）。
一方、この時に Havoc のコンソールを確認すると、問題なく C2 接続が行えていることが確認でき（図 42）、EDR に検知・防御されることなく Havoc エージェントの実行に成功しました。

次に、実際の攻撃事例などで報告されることの多い「-EncodedCommand」オプションを利用します。
このオプションは UTF-16LE でエンコードされた文字列を Base64 デコードしてから PowerShell スクリプトとして実行します。
前述の PowerShell スクリプトを Base64 エンコードした文字列（図 43）を使って実行したところ、EDR で不審なイベントとして検知されました。

図 41　PowerShell スクリプト実行直後の様子

図 42　Havoc サーバーに対し C2 接続が成功した様子

図 43　エンコードした PowerShell スクリプト抜粋

VBScript

VBScript は Microsoft が開発した、Visual Basic に似たスクリプト言語です。今後の Windows リリースで廃止が予定されていますが、現在でも頻繁に攻撃者に悪用されているため検証対象としました。VBScript で通信やプロセスの起動を行うために、オートメーションが可能なソフトウェアを CreateObject 関数でオブジェクトを生成してから利用します。今回の要件としては HTTP 通信が利用できること、ファイルに保存できること、実行ファイルを実行できることが必要なので、それぞれ対応するオブジェクトを生成します。

上記要件を満たすスクリプトを作成し、.vbs ファイル（図 44）として保存し実行したところ、製品 A では検知しませんでしたが（図 45）、製品 B では検知しました。製品 B では単なる「不審なイベント」や、「スクリプトインタープリターがリモートのファイルを取得した」といったアラートを検知していました。

Windows Command Shell（cmd.exe）

「コマンドプロンプト」も検証の対象としました。コマンドプロンプト単体では外部からのファイル取得が行えないため、certutil.exe を利用してファイルをダウンロードし、その後、ダウンロードしたファイルを実行します。これも以前から使われ続けている攻撃手法です。

certutil.exe でファイルのダウンロードを試みた段階で製品 A、製品 B ともに EDR により検知しました。

また、certutil.exe に設定した引数を修正して偽装を試みた場合や、製品 A において、存在しない URL や存在しないパスなどを試行した際も、いずれも検知しました。

このような傾向からは、EDR 製品が certutil.exe の実行にかなり注目して脅威判定を行っている可能性が考えられます。

```
Option Explicit

Dim objHTTP, strURL, strFilePath, objFSO, objShell, bStrm

Set objHTTP = createobject("Microsoft.XMLHTTP")
Set bStrm = createobject("Adodb.Stream")
Set objShell = CreateObject("WScript.Shell")
```

図 44　VBScript ファイル一部抜粋

ID	External	Internal	User	Computer	OS	Process	PID	Last	Health
12ff88f0	192.168.1.1…	192.168.30.…	user	DESKTOP-RC2OUSG	Windows 10	safe_execut…	5400	2s	healthy

図 45　Havoc サーバーに C2 接続が成功した様子

MSHTA

MSHTA とは Windows で HTML アプリケーションを実行するためのホストプログラムです。MSHTA.exe はインターネット上に公開されている HTA ファイルを実行することができますが、MSHTA のみで追加ファイルのダウンロードや実行はできません。その場合、VBScript や JScript などと組み合わせて実行する必要があります。

今回の検証では JScript から cmd.exe を呼び出し、cmd.exe から PowerShell.exe を呼び出すスクリプトを作成しました。最終的に PowerShell.exe で Havoc エージェントをダウンロードし、ディスク上に保存した後に実行します。また、実行の様子を Process Monitor を使って様子を確かめました。

図 46 のように Process Monitor の記録からは、mshta.exe → cmd.exe → PowerShell.exe → ローダーと順に実行された流れが確認できました。

製品 A はこの挙動を的確に「MSHTA プロセスの不審なイベント」として検知しました。

検証結果の考察

製品 A、B 共に概ね攻撃を検知することができましたが、最初に検知できなかった部分について考察します。

両製品ともに、PowerShell で作成した.ps1 ファイルの実行を検知することができませんでした。ファイルの中身は特定の URL からファイルをダウンロードし実行するだけのシンプルなコードであるため、それだけで脅威と考えるべきなのか判定は難しいと考えられます。

一方で、EncodedCommand 引数で実行した場合は、両製品とも検知することができました。本件で検証した EDR は検知ロジックを確認することができないため、検知した理由を特定することは難しいですが、スクリプトが Base64 エンコードされているという事実のみをもって脅威の対象と判定された可能性も考えられます。

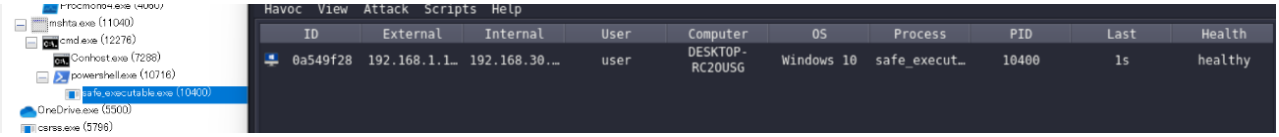

図 46　Process Monitor で観測した結果

複数の EDR 回避手法を組み合わせた検証

複数の検知回避手法を組み合わせて EDR に検知されにくいと考えられる攻撃を試行しました。

攻撃フロー

図 47 はこの攻撃のフローを示しています。VBScript のスクリプトファイルから侵害を開始し、PowerShell を経由して、最終的にメモリ上でコードを実行できるローダーを実行します。

なお、本節の検証では、攻撃フローをローダーが実行されるまでと、実行されたあとの 2 つのフェーズに分けて検討します。

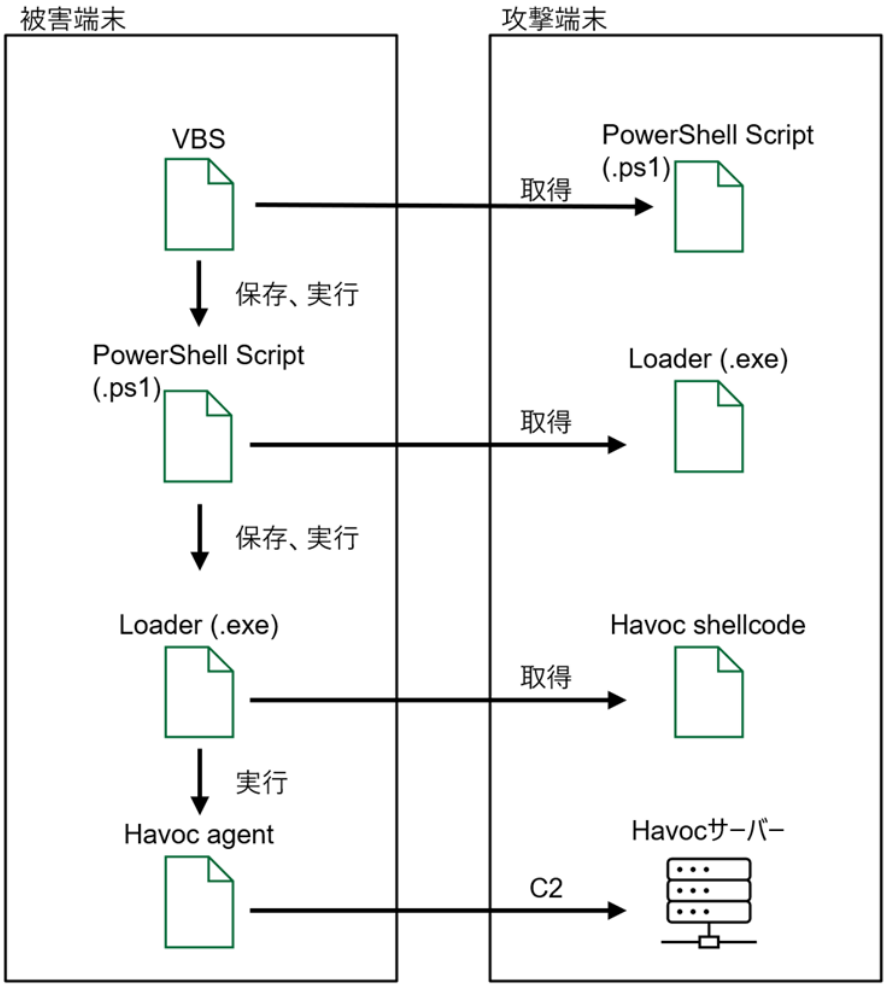

図 47　攻撃フロー図

ローダーが実行されるまでのフェーズ

まず VBScript を取り上げます（図 48）。今回の検証では VBScript ファイルに細工は行いませんでした。これは、通常、EDR はプロセスのイベントを中心に分析するため、スクリプトファイルの内容は検知精度に大きな影響を与えないと考えたためです。

この VBScript ファイルが実行されると、まず HTTP 通信で指定された URL からペイロードデータを取得し、Temp フォルダーに保存します。ここで取得したデータは PowerShell のスクリプトファイルなので、そのまま実行します。

この PowerShell スクリプトも特段細工をしていません。ローダーとなる.exe ファイルを取得して実行するだけの 4 行しかない小さなスクリプトファイルです。

VBScript から PowerShell スクリプトを実行し、そこから.exe ファイルが実行される流れは EDR で検知すべき疑わしいイベント（群）と考えられますが、この段階では製品 A、B 共に検知しませんでした。

ローダーが実行された後のフェーズ

本検証ではマルウェアのローダーを自作し、アンチウイルスで検知されないよう若干の難読化を施しました。このローダーの一番のポイントは、取得した Shellcode をメモリ上で実行することです。一般的なプロセスインジェクションの手法を用いることで実現しました。

この手法は広く利用されており、特定の Windows API を使用するため、容易に検知するかと思われましたが、製品 A では検知しませんでした。一方、製品 B では「メモリ上で脅威を検知した」というアラートが発報されました。

製品 A では図 49 に示す通り検知することなく最終的に Havoc のエージェントを実行することに成功しました。

図 48　攻撃に利用した VBScript ファイル

図 49　EDR では検知されず、Agent は動作している様子

検証結果の考察

本稿冒頭の基本的な検証結果をもとに、EDR による検知回避を企図して手法を複数組み合わせたところ、一部の製品ではその回避に成功しました。

なお、今回の検証ではイベントトレースやカーネルコールバックといった EDR の監視の仕組みに干渉するような複雑な技術を用いるのではなく、脅威判定の対象となり難いと考えられるようなシンプルな手法を組み合わせたことが、奏功したものと考えています。

Havoc のエージェントそのものは、ほぼ初期設定まま使用しているため、単体で実行を試みても容易に検知されてしまいます。しかし、本件のようにシンプルな手法を組み合わせることで検知されずに実行することが可能です。つまり、既知の攻撃ツールに対しても同様の注意が必要です。

例えば Mimikatz のように古くから利用されているツールであっても、本件のように実行方法を工夫したり、実行ファイルをカスタマイズしたりして検知を回避することが可能です。

CIC における取り組み事例

CIC では様々な EDR 製品を監視しており、各製品が発報するアラートだけでなく、定期的にハンティング作業を行うことで、複雑な脅威や高度な脅威の兆候などを広く発見する試みを継続しています。このハンティング作業は、独自に作成した複数の脅威シナリオに基づいて、各シナリオに対応するイベント(群)を発見することを目的として実施しています。

今後の展望として、自動化および各種アラートを組み合わせた検知の実装を進めています。

各種アラートを組み合わせた検知とは、ハンティングの発展形として、アラートの重大度や検知した機器に関わらず不審と判断できるイベントをルールとして定義し検知させることです。図 50 は旧来の個別検知と各種アラートを組み合わせた検知のコンセプトを並べて示したものです。

本稿でここまで試行してきた EDR 回避手法に関連した「アラートを組み合わせた検知」のシナリオとして、2 つの具体例を示します。

[1] EDR で取得したデータと脅威情報を組み合わせるルール

VBScript から PowerShell スクリプトが生成され、PowerShell スクリプトが C2 サーバーへ通信を行うシナリオを想定します。このケースでは、それぞれのイベントを個別に検知すると「不審な VBScript が実行された」、「不審な PowerShell が実行された」、「既知の C2 サーバーへの通信は発生した」という 3 種類のアラートが検知されると考えられます。

CIC が取り組んでいる「アラートを組み合わせた検知」は、これらの一連の不審なイベントをパターン化することで、より高い精度で脅威を検知するための試みです。

これにより、さらに高度な脅威の検知や、大量に発報されるアラートの負荷軽減、分析作業の効率化などが見込まれます。EDR のデータのみならず、プロキシーログなどの他の情報と組み合わせた分析を行う必要があります。この分析は、データの特性や分析順序などを考慮した複雑な内容となります。

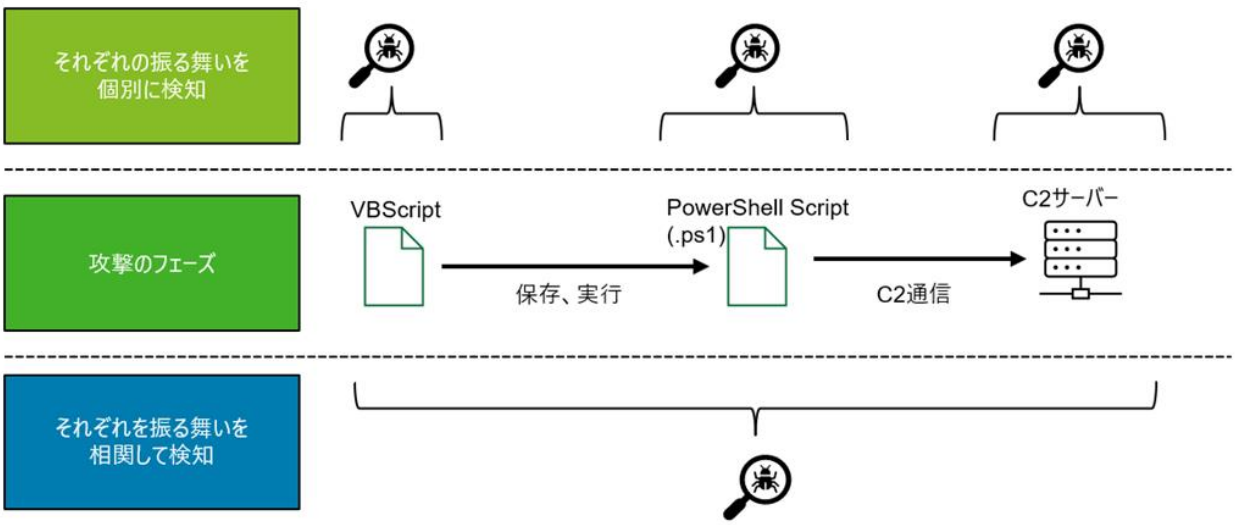

図 50　個別検知と「組み合わせ検知」のイメージ

[2] Shell 系プロセスの異常なイベントを検知するルール

cmd.exe や PowerShell.exe など Shell 系のプロセスがユーザー権限で起動された後に、子プロセスとして高権限のプロセスが起動されたことを検知するようなルールも有効と考えられます。ただし、これだけでは特殊な方法で管理者権限を取得するツールなどで誤検知が発生する可能性があります。

図 51 はこのルールの構成要素を示したものです。破線部のようなレジストリへのアクセスや、不審なファイルの設置など攻撃者の挙動に基づいたルールを設定することにより、精度の高い検知が可能になります。

最後に

本稿では、EDR 回避手法について検証を行った結果を紹介いたしました。一部の攻撃手法では EDR に検知されることなく「マルウェア」が実行されたことを示す結果となりましたが、このような状況は特段珍しいことではありません。

EDR に検知されないイベントがあること、EDR を回避されるリスクがあることを把握したうえで運用することが重要です。

CIC で開発を進めているような高度な組み合わせ検知だけでなく、監査の強化や最小権限の徹底など、有効な方策は他にも考えられます。情報セキュリティにかかわるポリシーや体制などの見直しの機会に、どのような対策を行えば必要とする効果が得られるのか、利便性やコストを踏まえて検討することを推奨します。

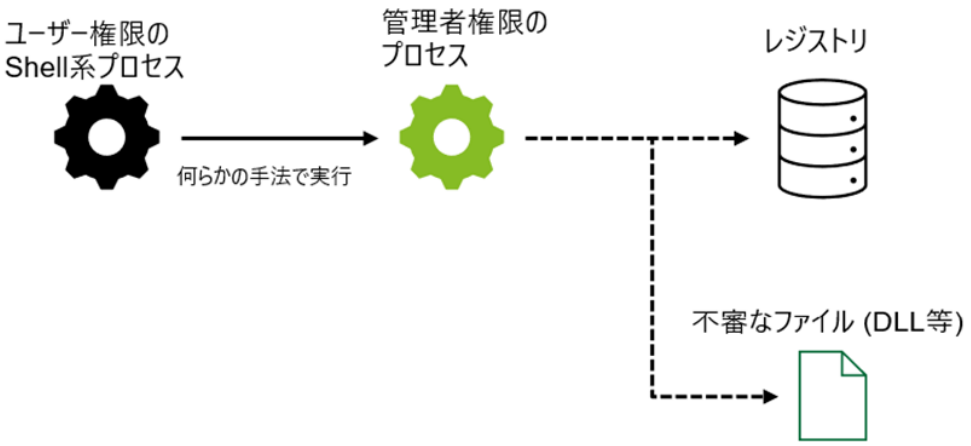

図 51　権限にフォーカスしたルールイメージ

おわりに

今回のレポートでは主として「脅威インテリジェンス」および「セキュリティ監視・分析」の観点から最新の脅威動向と CIC での観測および検証内容、そしてそれらへの対応方針の考え方などを紹介しました。DX の波及に伴い複雑化する一方の IT 環境ですが、脅威もまた衰えを見せることなく継続し、毎月のように重大な被害事例が報告されるなか、脅威インテリジェンスと監視・分析技術の両輪なしに実効的なセキュリティサービスを提供することは困難です。CIC では、これらの領域を組み合わせて補完し合うことでより質の高いサービスを提供することを目指しています。

後段の記事で紹介したような根本的な対処が難しい攻撃手法や、EDR などのセキュリティソリューションを回避する攻撃に対して、CIC で実装を進めている高度な「組み合わせ検知」は有効な対策となります。しかし、担保したいセキュリティレベルによっては、EDR だけでなくアカウント管理システムや資産管理サービスなどの監査機能をも駆使して様々なイベントを収集する必要があり、運用面で一定の負担となることは避けられません。
事業内容や IT 資産の構成に応じて「何をどこまでどうやって守るのか？」 目指すべきセキュリティレベルやその適用範囲を検討するとき、優先順位を量るためには脅威インテリジェンスの情報が、技術的な実現可能性や運用負荷を量るためには詳細な技術的背景が欠かせません。この領域を検討される際にはぜひご相談いただければと思います。

今後もセキュリティ対策や情報収集の参考にしていただくべく、CIC の監視およびインテリジェンスサービスで得られた知見や分析結果を発信していきます。

執筆者

佐藤 功陛	**パートナー**
鳥谷部 彰則	**マネージングディレクター**
吉村 修	
水越 尚平	
日平 祐介	
山中 理央	
前田 幸平	

Deloitte.
デロイト トーマツ

デロイト トーマツ サイバー合同会社
Cyber Intelligence Center (CIC)
Mail: ra_info@tohmatsu.co.jp
URL: www.deloitte.com/jp/dtcy
【国内ネットワーク】東京・名古屋・福岡

IS 669126 / ISO 27001