IIR

Internet
Infrastructure
Review

Mar.2020
Vol.46

定期観測レポート

SOCレポート

フォーカス・リサーチ(1)

Windowsのメモリイメージ取得の注意点

フォーカス・リサーチ(2)

解析対象への前提知識を必要としないバイナリプログラム解析技術

IIJ
Internet Initiative Japan

Internet Infrastructure Review

March 2020 Vol.46

エグゼクティブサマリ

東京で二度目のオリンピック・パラリンピック競技大会が開催される2020年。大会に向けた準備が最終段階に入ろうとするなかで飛び込んできたのは、新型コロナウイルスのニュースでした。地球上で人や物の往来がますます活発になった現在、感染症の急速な伝播を防ぐことが非常に困難になっていると、日々の報道を見ていて強く感じます。WHOや各国の政府機関が協力して伝播防止に取り組んでいるにもかかわらず、感染者の数は日に日に増えています。

急速に伝播するのはウイルスによる感染症だけではありません。インターネット上では膨大な情報が猛烈なスピードで流通しており、新型コロナウイルスに関しても様々な情報が私たちの目に入ってきます。それらの情報は必ずしも正確なものばかりではなく、特定の視点に立ったもの、特定の者の利益を代弁したもの、あるいは、故意に受け手を誤認させる悪意を持ったものなど、様々です。情報通信技術が高度に発達した今を生きる私たちは、多様な情報を入手する自由を持つと同時に、膨大な情報の真偽を自ら調べ、考え、行動することが求められていると、今回の出来事を通じてあらためて認識させられました。

「IIR」は、IIJで研究・開発している幅広い技術を紹介しており、日々のサービス運用から得られる各種データをまとめた「定期観測レポート」と、特定テーマを掘り下げた「フォーカス・リサーチ」から構成されています。

1章の定期観測レポートでは、SOCレポートを取り上げます。IIJのSOCは、サービス提供しているセキュリティ機器のログをはじめ、膨大なログを情報分析基盤で分析し、得られた脅威情報を情報発信サイト「wizSafe Security Signal」でタイムリーに発信しています。本レポートにおいては、2019年の主要なセキュリティトピックのなかからSOCが注目したものをリストアップすると共に、情報分析基盤の活用から明らかになった特筆すべき活動として、Elasticsearchサーバからの情報漏えい、DDoS攻撃、Emotetについて紹介します。いずれもIIJ独自の観測データを交えた解説となっており、興味深く読んでいただけると思います。

2章のフォーカス・リサーチでは、Vol.45に引き続きフォレンジック向けメモリイメージの取得について解説します。前号はLinuxのメモリイメージの取得に関してでしたが、今号はWindowsのメモリイメージの取得についてです。単にメモリイメージを取得するツールの紹介ではなく、メモリイメージの取得時や解析時に注意すべき点を解説します。また、個々のプロセスのダンプを確実に取得する方法も提案します。

3章のフォーカス・リサーチで取り上げたのは、IIJ-II技術研究所が行っている、解析対象への前提知識を必要としないバイナリプログラム解析技術です。プログラム解析技術は統合開発環境に組み込まれ、開発の効率化やバグの削減などに役立っています。一方、マルウェアの疑いのあるプログラムの振る舞いを調べる際など、既に配布された「得体の知れない」バイナリプログラムの解析が必要となる場合があります。本研究では前提知識のないバイナリプログラムでも静的解析が可能な技術を開発しており、ここではその内容を紹介します。

IIJは、このような活動を通してインターネットの安定性を維持しながら、日々、改善・発展させていく努力を続けています。今後も企業活動のインフラとして最大限に活用いただけるよう、様々なサービスやソリューションを提供し続けてまいります。

島上 純一（しまがみ じゅんいち）
IIJ 取締役 CTO。インターネットに魅かれて、1996年9月にIIJ入社。IIJが主導したアジア域内ネットワークA-BoneやIIJのバックボーンネットワークの設計、構築に従事した後、IIJのネットワークサービスを統括。2015年よりCTOとしてネットワーク、クラウド、セキュリティなど技術全般を統括。2017年4月にテレコムサービス協会MVNO委員会の委員長に就任。

SOCレポート

1.1 はじめに

IIJでは、2016年にセキュリティブランド「wizSafe（ウィズセーフ）」を立ち上げ、お客様が安全にインターネットを利用できる社会の実現に向けて日々活動しています。過去Vol.38[*1]の「SOCレポート」ではwizSafeの中核である情報分析基盤について、Vol.42[*2]では2018年に明らかとなった脅威と情報分析基盤を用いた新たな取り組みを紹介しました。今回は2019年におけるセキュリティの主要なトピックについて第1.2節で振り返り、取り上げたトピックに関連する脅威について情報分析基盤上で観測したものを第1.3節で紹介します。

1.2 2019年セキュリティトピックス

ここでは、2019年の主要なセキュリティトピックの中から、SOCが注目したものを表-1にピックアップしてまとめます。

＊1　Internet Infrastructure Review（IIR）Vol.38（https://www.iij.ad.jp/dev/report/iir/038/01.html）。
＊2　Internet Infrastructure Review（IIR）Vol.42（https://www.iij.ad.jp/dev/report/iir/042/01.html）。

表-1 2019年セキュリティトピックス

月	概要
1月	**国内のインターネットサービス企業が提供するファイル転送サービスにおいて、第三者の不正アクセスに起因する個人情報の漏えいが発生した。約480万件の会員情報が影響を受けたとされ、サービスを2020年3月31日に終了すると発表された。** "「宅ふぁいる便」サービス終了のお知らせ（2020年1月14日）" https://www.filesend.to/
2月	**総務省及び国立研究開発法人情報通信研究機構（NICT）は、容易に推測可能なパスワードを入力するなどの方法により、サイバー攻撃に悪用される恐れのあるIoT機器の調査及び当該機器の利用者への注意喚起を行う取組「NOTICE（National Operation Towards IoT Clean Environment）」を2月20日から開始した。** "IoT機器調査及び利用者への注意喚起の取組「NOTICE」の実施" http://www.soumu.go.jp/menu_news/s-news/01cyber01_02000001_00011.html "IoT機器調査及び利用者への注意喚起の取組「NOTICE」の実施" https://www.nict.go.jp/press/2019/02/01-1.html
3月	**3月8日をもって「Coinhive」のサービスが終了した。仮想通貨の度重なる仕様変更や市場価値下落などの要因により、経済的にサービスを継続することが難しくなったとのこと。**
3月	**海外のPCメーカが管理するサーバがAPT（Advanced Persistent Threat：持続的標的型）攻撃を受けた。この攻撃により、同社製ノートパソコンに付随するユーティリティを利用しアップデートを実行した一部のユーザへ、悪意のあるコードを含むファイルが配信されていた。** "ASUS response to the recent media reports regarding ASUS Live Update tool attack by Advanced Persistent Threat (APT) groups" https://www.asus.com/News/hqfgVUyZ6uyAyJe1
4月	**国内の高等教育機関などで利用される"ac.jp"ドメインが、取得資格を満たさない第三者により取得され、成人向けWebサイト開設に利用されていたことが発覚した。ドメイン登録時の資格要件確認に不備があったとのこと。公共性の高いドメインの信頼性確保は重要であることから、総務省は再発防止策などを要請した。** "株式会社日本レジストリサービスに対する「.jp」ドメイン名の管理・運用に係る措置（要請）" http://www.soumu.go.jp/menu_news/s-news/01kiban04_02000152.html
5月	**リモートデスクトップサービスに存在するリモートコード実行の脆弱性（CVE-2019-0708）、通称「BlueKeep」が公開された。マルウェアの感染拡大に重大な影響を与える脆弱性であるとの判断から、サポートが終了したOSに対してもセキュリティ更新プログラムが提供された。11月には実際にBlueKeepを悪用した攻撃も観測された。** "CVE-2019-0708 | リモート デスクトップ サービスのリモートでコードが実行される脆弱性" https://portal.msrc.microsoft.com/ja-JP/security-guidance/advisory/CVE-2019-0708
5月	**セキュリティ企業3社が不正アクセスを受け、セキュリティソフトウェアの開発ドキュメントやソースコードを含む機密情報が流出した可能性があることが発表された。後日、一部企業では、本事象による影響を受けていないことが確認された。** "Top-Tier Russian Hacking Collective Claims Breaches of Three Major Anti-Virus Companies" https://www.advanced-intel.com/blog/top-tier-russian-hacking-collective-claims-breaches-of-three-major-anti-virus-companies
6月	**FreeBSD及びLinuxカーネルに、細工されたSACKパケットを受信することでカーネルパニックを引き起こす可能性のあるTCPベースの脆弱性（CVE-2019-11477）、通称「SACK Panic」を含む複数の脆弱性が存在することが発表された。**
7月	**バーコード決済サービスにおいて、一部のアカウントが第三者による不正アクセス及び不正利用の被害を受けたと公表された。複数端末からのログインに対する対策や二要素認証を含む追加認証の検討が不十分であったことなどが原因として挙げられている。本事件を受け、同サービスは9月30日をもって廃止された。** "「7pay（セブンペイ）」サービス廃止のお知らせとこれまでの経緯、今後の対応に関する説明について" https://www.sej.co.jp/company/important/201908011502.html
7月	**国内の仮想通貨取引所において、約30億円分の仮想通貨が不正流出したことが公表された。流出した仮想通貨は、いずれもオンライン環境下で管理される「ホットウォレット」にて保管されており、その秘密鍵が窃取・不正利用されたと見られる。** "（開示事項の経過）当社子会社における仮想通貨の不正流出に関するお知らせとお詫び（第三報）" https://contents.xj-storage.jp/xcontents/AS08938/8a8b8ec7/f5b1/445e/a543/eade0775d325/140120190716472191.pdf
7月	**国内の自動車製造・販売会社の内部情報を格納した全文検索エンジンElasticsearchが外部から認証不要で参照できる状態であったことが公表された。格納されていた内部情報は、従業員の個人情報や内部ネットワーク、端末情報など約40GB分に上った。** "Honda Motor Company leaks database with 134 million rows of employee computer data" https://rainbowtabl.es/2019/07/31/honda-motor-company-leak/
8月	**2019年4月以降に報告された複数のSSL VPN製品の脆弱性を狙った攻撃が活発化した。8月にはBlack Hat USA 2019にて脆弱性に関する詳細が公開されたほか、PoCや当該脆弱性を悪用した攻撃の観測情報も報告されている。SOCにおいても、Pulse Secureの脆弱性（CVE-2019-11510）を悪用した攻撃通信を観測した。** "OVER 14,500 PULSE SECURE VPN ENDPOINTS VULNERABLE TO CVE-2019-11510" https://badpackets.net/over-14500-pulse-secure-vpn-endpoints-vulnerable-to-cve-2019-11510/
9月	**エクアドル国民2,000万人超の情報を格納した全文検索エンジンElasticsearchが外部から認証不要で参照できる状態であったことが発表された。** "Report: Ecuadorian Breach Reveals Sensitive Personal Data" https://www.vpnmentor.com/blog/report-ecuador-leak/
9月	**Wikipedia、Twitch、Blizzardの各サーバに対するDDoS攻撃が発生した。一連の攻撃はMirai亜種とみられるボットネットにより引き起こされていた。**
11月	**JPCERT/CCは、マルウェアEmotetに関する注意喚起を行った。2019年10月後半から、実在の組織や人物になりすましたメールに添付されたWordファイルによる感染被害の報告を多数受けているとのこと。SOCでは2019年9月末より活発な活動を観測している。** "マルウエア Emotet の感染に関する注意喚起" https://www.jpcert.or.jp/at/2019/at190044.html
12月	**複数の企業が、マルウェアEmotetに感染したことを公表した。感染した端末に保存されたメールアドレスやメール本文が漏えいした可能性があり、各企業を装った不審なメールの添付ファイルやURLリンクを開かないように注意喚起を行っている。**
12月	**自治体がリース契約満了に伴い返却したサーバから、データ削除前のハードディスクが盗難に遭っていたことが発覚した。当該ハードディスクはリース会社がデータ消去を委託している企業の社員により横領され、オークションサイト上で出品・落札されていた。** "リース契約満了により返却したハードディスクの盗難について" https://www.pref.kanagawa.jp/docs/fz7/prs/r0273317.html
12月	**海外のSNSに登録された2億6700万件超のユーザ情報が、認証不要で外部から閲覧可能なElasticsearchサーバ上に公開されていたことが報告された。** "Report: 267 million Facebook users IDs and phone numbers exposed online" https://www.comparitech.com/blog/information-security/267-million-phone-numbers-exposed-online/

1.3 観測情報

本節では、情報分析基盤を活用して明らかになった、2019年の特筆すべき活動について取り上げます。

1.3.1 外部公開されたElasticsearchサーバからの情報漏えい

■ Elasticsearchと情報漏えい事件

2019年は大規模な個人情報の漏えいが度々発生しました。中でも全文検索エンジンElasticsearchの設定不備による情報漏えいが目立った印象です。第1.2節で紹介したセキュリティトピックスでも、Elasticsearchに関連した情報漏えい事件を3件取り上げています。掲載した事件のほかにも、2019年1月には米国のクラウドデータ管理会社の顧客情報[*3]、2019年5月にはパナマ国民の約90%の情報[*4]が含まれていたElasticsearchサーバが外部から認証なしでアクセス可能な状態であったことが報告されています。いずれも流出した情報の量が多く、数百万件を超える情報や数十GB以上のデータが流出しています。

Elasticsearchは、Elastic社を中心に開発されているApache Luceneを基盤としたオープンソースの全文検索エンジンです[*5]。分散型システムで、並行して大量のデータを処理することで高速な検索を実現します。そのため、大規模な情報を扱う際に選択されることも少なくなく、セキュリティインシデントが発生すると流出する情報の量も多くなる傾向があると考えられます。また、ElasticsearchではRESTful APIが提供されており、HTTPプロトコルでデータの検索や操作が可能です。HTTPアクセスする際のポート番号はデフォルトで9200/TCPが用いられます。

SOCでは、アクセス可能なElasticsearchサーバを探索していると思われる9200/TCPへのスキャン通信が2019年に増加したことを観測しています。

■ 観測情報

図-1にIIJマネージドファイアウォールサービスにて観測された9200/TCPへのスキャン通信の件数及び送信元IPアドレス数の推移を示します。通信件数の値は年間に観測した9200/TCP宛てスキャン通信の合計を100%として正規化した当該通信の割合を記載しています。

図-1において、9月21日から10月31日にかけてのスキャン通信の増加が目立ちます。この41日間で発生した9200/TCPへのスキャン通信は、2019年の1年間に発生したスキャン通信の約23.37%を占めており、1日あたりの送信元IPアドレス数はピーク時で30,394件まで増加しています。これは1月1日のIPアドレス数(308件)と比較すると約98.68倍の数値です。この時期にElasticsearchに関する脆弱性は発表されておらず、スキャン通信が増加した明確な理由は分かっていません。

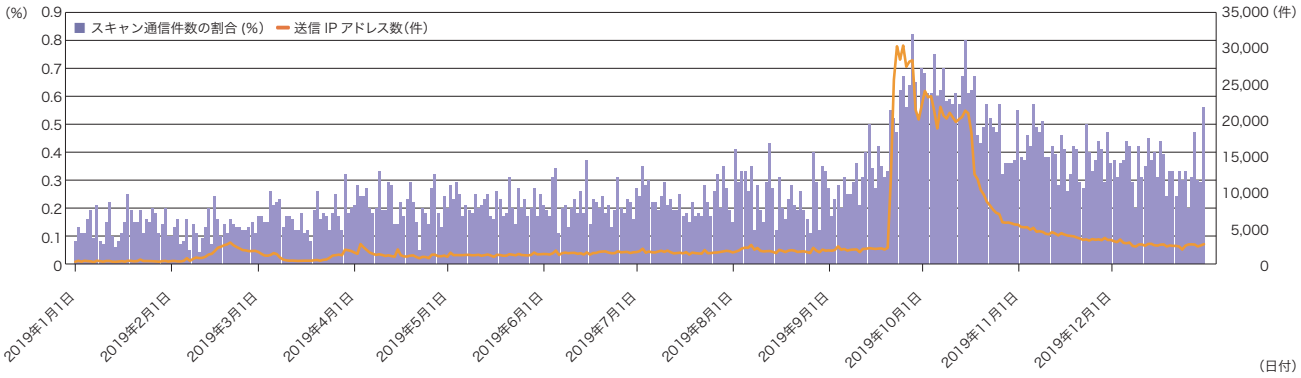

図-1 9200/TCPへのスキャン活動(2019年1月～12月)

＊3 techcrunch(https://techcrunch.com/2019/01/29/rubrik-data-leak/)。
＊4 securityaffairs、"Personally identifiable information belonging to roughly 90% of Panama citizens were exposed on a poorly configured Elasticsearch server."(https://securityaffairs.co/wordpress/85462/data-breach/panama-citizens-massive-data-leak.html)。
＊5 Elasticsearch(https://www.elastic.co/jp/elasticsearch)。

スキャン通信が増加する直前の9月16日にエクアドル国民2,000万人超の情報が閲覧可能であったとの報道がされており、この事件を契機とした増加である可能性も考えられますが、関連性を示す明確な証拠は見つかっていません。

2月中旬と4月上旬についても、一時的なスキャン活動の増加が見られます。送信元IPアドレス数は2月20日と4月3日に一時的に3,000件近くまで増加していました。2月19日に、Elastic社からElasticsearchの脆弱性(CVE-2019-7611)[*6]が公表されており、それを起因としたElasticsearch稼働サーバの探索と推測されます。CVE-2019-7611はアクセス許可に関する脆弱性で、悪用された場合、情報の取得や改ざんが行われる可能性があります。また、この時期に発生したElasticsearchへの攻撃についてCisco Systems社のセキュリティ部門であるTalosからレポートが公開されています[*7]。レポートによると過去に発表された脆弱性(CVE-2014-3120、CVE-2015-1427)を狙った攻撃が観測されていたようです。

年間を通したトレンドとしても、9200/TCPへのスキャン通信は全体的に増加傾向があります。12月における1日あたりの通信件数の平均は、1月平均の0.14%から0.35%と約2.46倍の増加、送信元IPアドレス数は1月平均の384.74件から2702.10件と約7.02倍も増加しています。同様の傾向は警察庁が公開している定点観測レポート[*8]でも見られ、Talosのレポートと同様にCVE-2015-1427を狙ったものと考えられる攻撃が紹介されています。

■ 対策

2019年に大きく報道されたElasticsearchに関連する情報漏えい事件のほとんどは、設定不備により認証不要でアクセス可能であったことが原因です。インターネット側からのアクセスが不要な場合はファイアウォールなどで9200/TCPを含む不要な通信を拒否したり、必要な場合も信頼できるIPアドレスからのみ接続を許可し適切な認証を設けるなど基本的な対策が重要です。また、Talosや警察庁のレポートにあるように、過去の脆弱性を狙う攻撃も依然として観測されています。システムに関連する脆弱性情報が公表された際には、システムへの影響を確認の上、修正プログラムの適用も併せて必要となります。

1.3.2 DDoS攻撃の観測

IIJでは様々な手法のDDoS攻撃を観測・対処しています。本項では、2019年のDDoS攻撃のトピックを取りまとめます。はじめに2019年にIIJが対処したDDoS攻撃のうち、IIJ DDoSプロテクションサービスで検出した攻撃についてまとめます。次に2019年に話題となった攻撃手法を取り上げ、最後に2019年に話題となった攻撃手法における国内の被害事例に関して、観測情報と併せて紹介します。

＊6 Elastic、"Security issues"(https://www.elastic.co/jp/community/security)。
＊7 Cisco Talos、"Cisco Talos Honeypot Analysis Reveals Rise in Attacks on Elasticsearch Clusters"(https://blog.talosintelligence.com/2019/02/cisco-talos-honeypot-analysis-reveals.html)。
＊8 警察庁、「Elasticsearch の脆弱性を標的としたアクセスの増加等について」(https://www.npa.go.jp/cyberpolice/important/2019/201910021.html)。

■ 2019年DDoS攻撃観測サマリ

2019年9月では、Wikipedia、Twitch、BlizzardへのDDoS攻撃が話題となりました。ここでは、2019年にIIJが対処したDDoS攻撃のうち、IIJ DDoSプロテクションサービスで検出した攻撃について取りまとめます。表-2にIIJ DDoSプロテクションサービスで検出した攻撃の件数や通信量を示します。

表-2の中で、TCPを用いた攻撃はSYN Flood攻撃やSYN/ACKリフレクション攻撃があり、UDPを用いた攻撃はUDP Amplification攻撃とUDP Flood攻撃があります。UDP Amplification攻撃では利用されるアプリケーションプロトコルにいくつか種類が存在し、DNS、NTP、LDAPなどがあります。

DDoS攻撃の発生件数は月単位で1日あたりの平均を算出していますが、2019年にはDDoS攻撃が顕著に活発な月はありませんでした。1秒あたりのパケット数が最も大きかったのは5月で、最も長い攻撃は1月に発生していました。また、最大パケット数が比較的大きかったのは5月、7月、12月でしたが、攻撃の最長時間は1時間未満に収まっていました。毎月の最大通信量及び最大攻撃時間で利用された攻撃種別の多くはLDAPやDNSを用いたUDP Amplificationが目立っています。

■ 2019年のDDoS攻撃トピック

表-2で紹介した攻撃手法のほかにも、DDoS攻撃として利用できる新手法がいくつか話題となりました。ここでは、2019年に話題となったDDoS攻撃の手法について、3つのキーワードを用いて紹介します。

- Web Services Dynamic Discovery（WSD）
- Apple Remote Management Service（ARMS）
- SYN/ACKリフレクション

1つ目のWSDは、Simple Object Access Protocol（SOAP）によって特定のネットワーク帯において対象のサービスを検出したり、データのやり取りを実現したりするプロトコルです。利用するポートは3702/UDPで、OSがWindows Vista以降のパソコンやプリンタなどで利用されていることが知られています。このプロトコルを利用したDDoS攻撃の可能性は、zeroBS GmbHにて示されています[*9]。なお、3702/UDPに対してインターネット上から応答を返すIPアドレス数は約63万個存在していたことが確認されています[*10]。SOCでは、2019年8月に3702/UDPに対するスキャン活動の増加を観測しています[*11]。図-2では、2019年にSOCで観測した当該ポートへのスキャン活動の推移を示します。なお、通信件数の値は年

表-2 2019年 DDoS観測情報サマリ

月	月間の件数（1日平均）	最大秒間パケット数（万）	最大通信量		最長攻撃時間	
			帯域	手法	時間	手法
1	13.58件	約179万	17.38Gbps	DNS Amplification	3時間20分	SYN Flood
2	15.75件	約284万	27.89Gbps	LDAP Amplification	1時間18分	LDAP Amplification
3	14.00件	約652万	19.30Gbps	SSDP Amplification	2時間32分	SSDP Amplification
4	22.96件	約97万	9.21Gbps	DNS Amplification	41分	DNS Amplification
5	16.16件	約886万	39.29Gbps	LDAP Amplification	41分	DNS Amplification
6	10.93件	約148万	8.11Gbps	SSDP Amplification及びSYN/ACKリフレクション	30分	SSDP Amplification及びSYN/ACKリフレクション
7	16.41件	約738万	75.67Gbps	DNS Amplification	38分	NTP Amplification
8	18.10件	約91万	8.77Gbps	LDAP及びDNS Amplification	1時間35分	UDP Flood
9	19.20件	約130万	11.71Gbps	LDAP及びDNS Amplification	43分	NTP Amplification
10	22.09件	約310万	23.09Gbps	LDAP及びDNS、NTPなどのAmplification	1時間56分	LDAP Amplification
11	13.36件	約70万	8.24Gbps	UDP Flood	25分	UDP Flood
12	10.38件	約607万	61.34Gbps	LDAP及びDNS Amplification	38分	NTP Amplification

＊9 zeroBS、"Analysing the DDOS-Threat-Landscape, Part 1: UDP Amplification/Reflection"（https://zero.bs/analysing-the-ddos-threat-landscape-part-1-udp-amplificationreflection.html）。
＊10 zeroBS、"New DDoS Attack-Vector via WS-Discovery/SOAPoverUDP, Port 3702"（https://zero.bs/new-ddos-attack-vector-via-ws-discoverysoapoverudp-port-3702.html）。
＊11 wizSafe、「wizSafe Security Signal 2019年8月 観測レポート」（https://wizsafe.iij.ad.jp/2019/09/746/）。

間に観測した3702/UDP宛てスキャン通信の合計を100%として正規化した当該通信の割合を記載しています。

図-2より、8月13日頃から当該ポートへのスキャンが増加していることが確認できます。また8月19日から8月末にかけて、当該ポートへのスキャン活動を実施する送信元IPアドレス数が増加しており、スキャン活動の増加は、BinaryEdgeの調査報告と概ね一致します。2月17日に発生した当該ポートへの通信を試みる送信元IPアドレス数の増加理由は定かではありませんが、2月19日にWS-Discoveryを用いたDDoS攻撃が発生していたことをBaidu, Inc.がレポートしています*12。このことから、WS-Discoveryを用いたDDoS攻撃は少なくとも2月頃には利用されていたと考えられます。しかし、国内で話題となったのは2019年9月になってからです。また、US-CERTのUDP Amplification Factorの資料においても、9月に公開された記事を引用する形で12月になってから追記されています*13。そのため、WS-DiscoveryがDDoS攻撃で本格的に利用されはじめたのは、実際に攻撃に利用され始めた数ヵ月後であったと考えられます。

2つ目のARMSは、Apple Remote Desktop（ARD）で利用されているサービスで、ARDはMacOSの複数端末をリモートから制御するためのアプリケーションです。ARMSは3283/UDPを通して、管理コンソールからのコマンドなどの命令を受信します。このARMSをインターネット上に公開している端末は、約4万台存在していたことが確認されています*14。図-3では、2019年にSOCで観測した当該ポートへのスキャン活動の推移を示します。なお、通信件数の値は年間に観測した

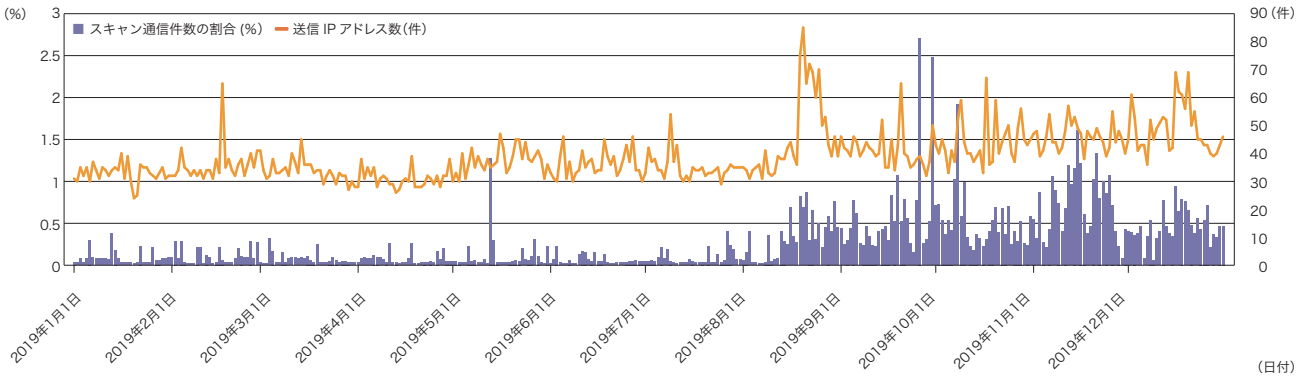

図-2 3702/UDPにおけるスキャン通信と送信元IPアドレス数の推移（2019年1月～12月）

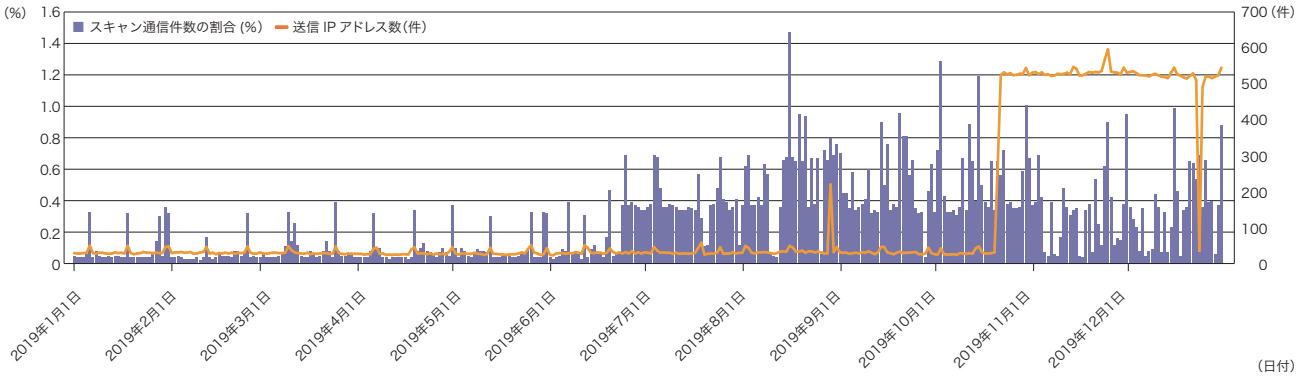

図-3 3283/UDPにおけるスキャン通信と送信元IPアドレス数の推移（2019年1月～12月）

*12 百度安全指数、"基于ONVIF协议的物联网设备参与DDoS反射攻击"（https://bsi.baidu.com/article/detail/128）。
*13 CISA、"Alert (TA14-017A)"（https://www.us-cert.gov/ncas/alerts/TA14-017A）。
*14 ZDNet、"macOS systems abused in DDoS attacks"（https://www.zdnet.com/article/macos-systems-abused-in-ddos-attacks/）。

3283/UDP宛てスキャン通信の合計を100%として正規化した当該通信の割合を記載しています。

図-3より、6月24日頃から当該ポートへのスキャンが増加していることが確認できます。また、10月22日頃からスキャン活動を実施する送信元IPアドレス数が増加しています。したがって、NetScout Systems, Inc.のレポートが公開される[*15]数日前からスキャン活動が増加していたことが分かります。

3つ目は、SYN/ACKリフレクション攻撃です。これは、TCPのthree-way handshakeの過程において、送信元アドレスを偽装したSYNパケットを多数のアドレスに同時に送信し、その応答であるSYN/ACKパケットを利用して送信元アドレスに対してDDoS攻撃を行う手法です。図-4にSYN/ACKリフレクション攻撃の全体像を示します。

SYN/ACKリフレクション攻撃の発生から、被害者に被害が発生するまでの流れを以下に説明します。図-4と併せてご覧ください。

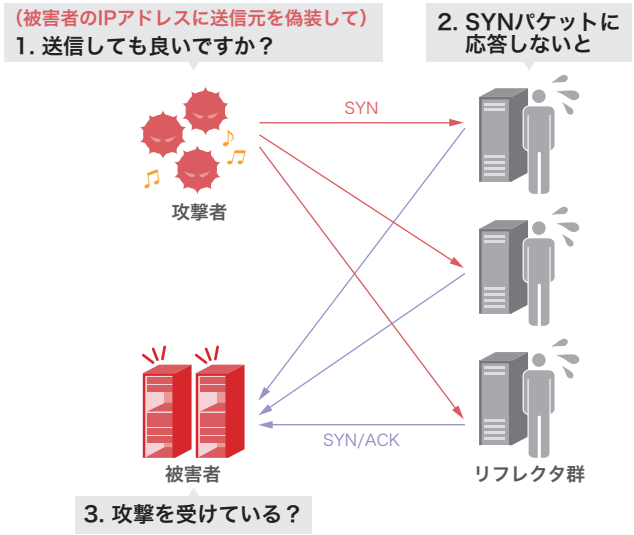

図-4 SYN/ACKリフレクション攻撃の全体像

1. 攻撃者は攻撃に使用するSYN/ACKパケットを発生させるために、送信元IPアドレスを攻撃対象に偽装し、送信元を偽装したSYNパケットをリフレクタに送信する。
2. three-way handshakeにおいて、そのSYNパケットに対してリフレクタはSYN/ACKパケットで応答する。
3. SYNパケットの送信元IPアドレスは偽装されていることから、リフレクタが返送したSYN/ACKパケットが被害者となるIPアドレス宛に届くことで攻撃が実現される。

本攻撃をSOCで観測したのは2018年で、小誌Vol.42の「1.2.2 SYN/ACKリフレクション攻撃」にて説明しています[*16]。このSYN/ACKリフレクション攻撃は、2006年頃にはTCP Amplification攻撃としてしられていた攻撃手法です[*17]。2014年には、SYN/ACKパケットやRSTパケット、PSHパケットを実装上発生しうる一般的な数よりも多く再送する端末がインターネット上で発見されました[*18]。実際に2014年に発見された端末を利用しているかは定かではありませんが、攻撃の原理としては同一のものです。SOCでは、SYN/ACKパケットを用いたTCP Amplification攻撃をSYN/ACKリフレクション攻撃と呼んでおり、7月頃から11月頃にかけて頻繁に観測されるようになりました。

これら2019年に話題となった3つの攻撃手法は、パケットの送信元を被害者に偽装し、リフレクタと呼ばれるDDoS攻撃を行うための踏み台を利用することが大きな特徴です。このような特徴を持つDDoS攻撃はDistributed Reflection Denial of Service（DRDoS）と呼ばれています。DRDoS攻撃の場合、攻撃者はリフレクタとして利用できるホスト及びポートを事前に調査し、悪用を試みます。そのため、DRDoSで利用できるポートが、インターネット上で誰からでもアクセスできてしまう場合、DRDoSのリフレクタとして利用されてしまう恐れがあります。WSDやARMSのようなDRDoS攻撃の場合には、送信元及び被害者だけに対策を求めるのではなく、リフレクタにおいても可能な対応策を取る必要があります。DRDoS攻撃の

＊15 NETSCOUT、"A Call to ARMS: Apple Remote Management Service UDP Reflection/Amplification DDoS Attacks"（https://www.netscout.com/blog/asert/call-arms-apple-remote-management-service-udp）。
＊16 Internet Infrastructure Review（IIR）Vol.42（https://www.iij.ad.jp/dev/report/iir/042/01.html）。
＊17 RFC 4732、"Internet Denial-of-Service Considerations"（https://tools.ietf.org/html/rfc4732#section-3.1）。
＊18 USENIX、"Hell of a Handshake: Abusing TCP for Reflective Amplification DDoS Attacks"（https://www.usenix.org/system/files/conference/woot14/woot14-kuhrer.pdf）。

場合、リフレクタとなっているサーバの管理者は、DDoS攻撃の標的ではありませんが、被害を受けているサーバやネットワークなどに対する攻撃に意図せず加担してしまいます。したがって、インターネット上で不必要にポートを開けていないかを確認し、意図したアクセスのみを許容する設定にしておくことが重要です。このことは、不正アクセスだけでなくDDoS攻撃で利用される可能性のあるリフレクタを削減することにも繋がるため、DDoS攻撃における攻撃者の行動を制限できる可能性が高まります。ただし、DRDoS攻撃の手法にはリフレクタとしてアクセス制御が容易でないものが存在します。それがSYN/ACKリフレクション攻撃です。その理由は次の「SOCによる観測状況」で説明します。

■ SOCによる観測状況

2019年には、先述した3つの攻撃手法による日本国内の組織やサービスを狙ったDDoS攻撃が発生しました。ここでは日本国内で発生したDDoS攻撃のうち、2019年に話題となった事例について、SOCで観測した情報を加えて解説します。日本国内の組織を狙ったWSD及びARMSを用いたDDoS攻撃は、2019年10月のJPCERT/CCの注意喚起に[*19]示されています。本事例は、DDoS攻撃にWSD及びARMSが用いられていただけでなく、仮想通貨を要求する脅迫メールが届く事例であることが判明しています。このような金銭目的と考えられる脅迫文を用いてDDoS攻撃を示唆する試みは、Ransom Denial of Service（RDoS）と呼ばれています。このRDoSは2019年だけでなく、2017年にも話題となった事例です[*20]。攻撃のアクターは両年の事例において同一であるかは定かではありませんが、少なくとも2017年には公表されていなかったWSDとARMSを用いたDDoS攻撃が2019年の事例で利用されていることは事実です。図-2及び図-3と併せて考えると、少なくともDDoS攻撃で用いられる攻撃インフラは悪用可能なプロトコルに順次適応していると考えられます。

SYN/ACKリフレクション攻撃が日本国内の企業に向けたDDoS攻撃で利用された事例として、表-2の6月の最大通信量・最長攻撃時間を記録した攻撃が挙げられます。SYN/ACKリフレクション攻撃の大きな特徴は、リフレクタとして任意のTCPポートが利用されるため、WSDやARMSのように特定のポートに紐づくサービスを悪用するものではありません。そのため、Webサーバで一般的に利用される80/TCPや443/TCPなどが利用されます。例えば、インターネット上に公開しているWebサーバ上のコンテンツは、幅広く閲覧してもらうために何処からでもアクセスできることが重要になります。この場合、ファイアウォールは意図して誰でもアクセスできる設定になっているはずです。この前提の上で、SYN/ACKリフレクション攻撃のリフレクタとして利用された場合、リフレクタ側では容易にアクセス拒否することが難しいと考えられます。図-5に、2019年にSOCで観測したSYN/ACKリフレクション攻撃のリフレクタとして利用されたTCPポートの割合を示します。

図-5から分かる通り、SYN/ACKリフレクション攻撃で利用されるTCPポートは、80/TCP及び443/TCP、またSimple Mail Transfer Protocol（SMTP）として利用される25/TCPです。これらは全体の95%以上を占めます。また、Othersには、21/TCPや22/TCP、587/TCPなどが多く含まれています。従って、SYN/ACKリフレクション攻撃で利用される踏み台となるポートは、インターネット上から比較的自由にアクセスしやすいTCPポートが対象となっていることが分かります。図-5で示したとおり、外部からのアクセスを許容するサービスが稼働しているTCPポートが踏み台とされやすいため、アクセス制御によって、リフレクタ側でSYN/ACKリフレクション攻撃に対処することが難しいものと考えられます。

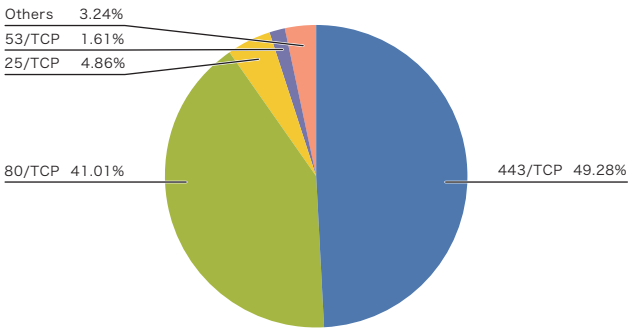

図-5 SYN/ACKリフレクション攻撃におけるリフレクタで利用されたポートの割合

*19 JPCERT CC、「DDoS 攻撃を示唆して、仮想通貨を要求する脅迫メールについて」（https://www.jpcert.or.jp/newsflash/2019103001.html）。
*20 radware、"Fancy Bear DDoS for Ransom"（https://security.radware.com/ddos-threats-attacks/threat-advisories-attack-reports/fancybear/）。

しかし、SYN/ACKリフレクション攻撃への対処の難しさはこれだけではありません。SYN/ACKリフレクション攻撃は、攻撃に関わりのある端末や被害者など、基本的にネットワーク全体を俯瞰してみなければ判断しにくいものです。実際にリフレクタとなっている個々のホストには攻撃端末からSYNパケットが大量に届くため、リフレクタして利用されているホストの管理者からはSYN Flood攻撃を受けているものと判断してしまう恐れがあります。その場合にブラックリスト方式でSYNパケットの送信元を永続的に遮断すると、DDoS攻撃が収束した後に被害者のIPアドレスから遮断設定を行ったホストへアクセスができなくなります。これはSYN/ACKリフレクション攻撃の副次的な被害と考えることが出来ます。

SYN/ACKリフレクション攻撃で国内の端末をリフレクタとして利用された事例は、SOCの情報発信サイト「wizSafe Security Signal[*21][*22][*23][*24]」にまとめています。なお、これらの事例は、リフレクタ視点で観測したSYN/ACKリフレクション攻撃であり、被害者視点の情報ではありません。そのため、SYN/ACKリフレクション攻撃における攻撃規模を全て示しているわけではないことにご留意ください。

1.3.3 Emotet

■ Emotetの概要

2019年の後半からは、メールを悪用して感染活動を行うEmotetと呼ばれるマルウェアが話題になりました。このマルウェアが最初に報告[*25]されたのは2014年、当時トレンドマイクロ社に在籍していたJoie Salvio氏によるものでした。当初、金融機関の情報を狙うバンキングトロジャンとして活動していたEmotetは、次第にその形態を変化させボットネット化していきました。更にモジュール化によるワーム機能の獲得により、様々なマルウェアやランサムウェアを拡散する能力を備えました。これにより近年では、金融機関の情報だけではなく、機密情報も盗み出すマルウェア（TrickbotやZeuSなど）をダウンロードさせるように変化してきました。また、Emotetを起点にダウンロードされた情報窃取の機能を持つマルウェアにより、ターゲットとなるシステムに侵入することで、最終的にRyukと呼ばれているランサムウェアのペイロードを実行する攻撃が報告されています。これらのマルウェアにより窃取した情報を元にターゲットとなるシステムに侵入し、Ryukと呼ばれるランサムウェアのペイロードを実行する多段攻撃triple threat[*26]についても報告が上げられています。このような変化に伴い、攻撃のターゲットも公共機関や民間企業へと変化を見せています。

国外の観測[*27]では、2019年6月頃からEmotetで使用されていたC2サーバの休眠が確認されていましたが、活動の休止は長くは続きませんでした。2019年8月末には当該サーバの活動再開が報告されており、同9月以降、IIJのメールゲートウェイサービス「IIJセキュアMXサービス」においてもEmotetへの感染を誘導する悪意あるメールの検出が増加していました。

SOCの観測では、Microsoft Word（doc）形式の添付ファイルを悪用した感染活動を多数確認しています。その後、別の感染経路として、メール本文にEmotetに感染させるdocファイルをダウンロードさせるURLを記載したメールが増加しました。

---

*21 wizSafe、「wizSafe Security Signal 2019年7月 観測レポート」（https://wizsafe.iij.ad.jp/2019/08/717/）。
*22 wizSafe、「Servers.comを狙ったDDoS攻撃の観測」（https://wizsafe.iij.ad.jp/2019/10/764/）。
*23 wizSafe、「2019年10月 TCP SYN/ACKリフレクション攻撃の観測事例」（https://wizsafe.iij.ad.jp/2019/12/820/）。
*24 wizSafe、「2019年11月 TCP SYN/ACKリフレクション攻撃の観測事例」（https://wizsafe.iij.ad.jp/2019/12/839/）。
*25 TREND MICRO、"New Banking Malware Uses Network Sniffing for Data Theft"（https://blog.trendmicro.com/trendlabs-security-intelligence/new-banking-malware-uses-network-sniffing-for-data-theft/）。
*26 cybereason、"RESEARCH BY NOA PINKAS, LIOR ROCHBERGER, AND MATAN ZATZ"（https://www.cybereason.com/blog/triple-threat-emotet-deploys-trickbot-to-steal-data-spread-ryuk-ransomware）。
*27 cBLEEPINGCOMPUTER、"Emotet Botnet Is Back, Servers Active Across the World"（https://www.bleepingcomputer.com/news/security/emotet-botnet-is-back-servers-active-across-the-world/）。

Emotetへの感染を誘導するdocファイルを開くと、Wordの初期設定状態では図-6に示すような「コンテンツの有効化」を促すメッセージが表示されます。そして「コンテンツの有効化」を許可することによりマクロが実行されます。なお、Wordの設定でマクロを有効化している場合には、図-6のような画面は表示されず、マクロは自動で実行されます。そして、マクロが実行されるとマルウェア配布サーバよりEmotetがダウンロードされ、感染へと繋がります。

一度感染すると、Emotetは感染を永続化するために、新しいサービスに自身をコピーし、自動実行されるように設定します。その後、感染したPCの情報窃取やC2サーバへの通信を行います。窃取される情報にはメール本文やアドレスのデータなどが含まれており、Emotetへの感染を誘導するメールにはこれらの情報を悪用し、メールの返信を装うものがあります。これが感染を広げる原因の1つになっています。前述のtriple threatと呼ばれる多段攻撃の例からも分かるとおり、Emotetは他のマルウェアの入り口として機能するため、最終的な被害は今後更に変化すると考えられます。

■ 観測情報

以下では、SOCにおけるEmotetの観測状況について報告します。

まず2019年9月から12月に検出した攻撃について、Emotetに関連する検出とその他の検出に分けた積み上げグラフを図-7に示します。図の横軸は日付を、縦軸は集計対象期間における合計検出件数を100%として正規化した割合を表します。

図中で、最初にEmotetの検出が目立つのは9月27日です。その後は10月の16日、17日、23日、24日に多く検知しています。11月に入ると、これまでの月と比べ多くの日数、割合でEmotetを検出しています。また、検出は平日に集中する傾向が見られます。12月3日から4日にかけて集計期間中の検出

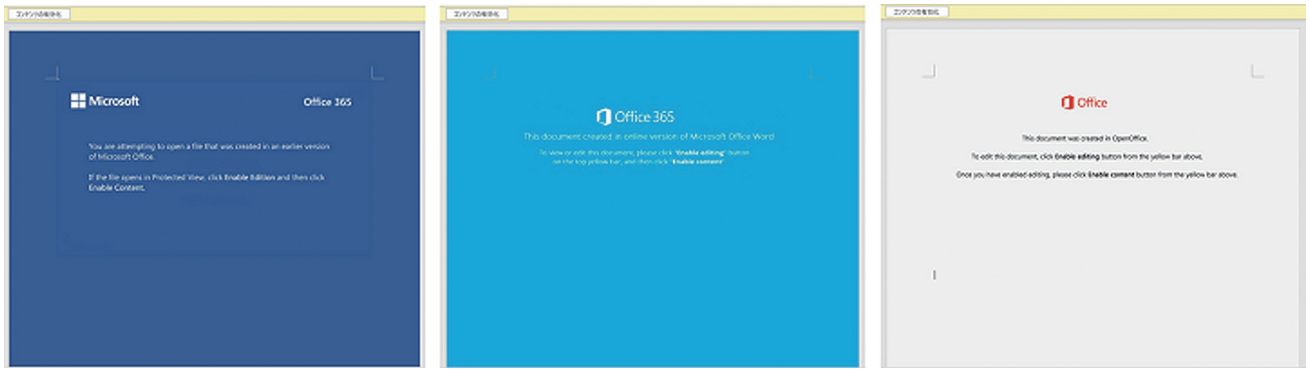

図-6「コンテンツの有効化」を促す画面の例

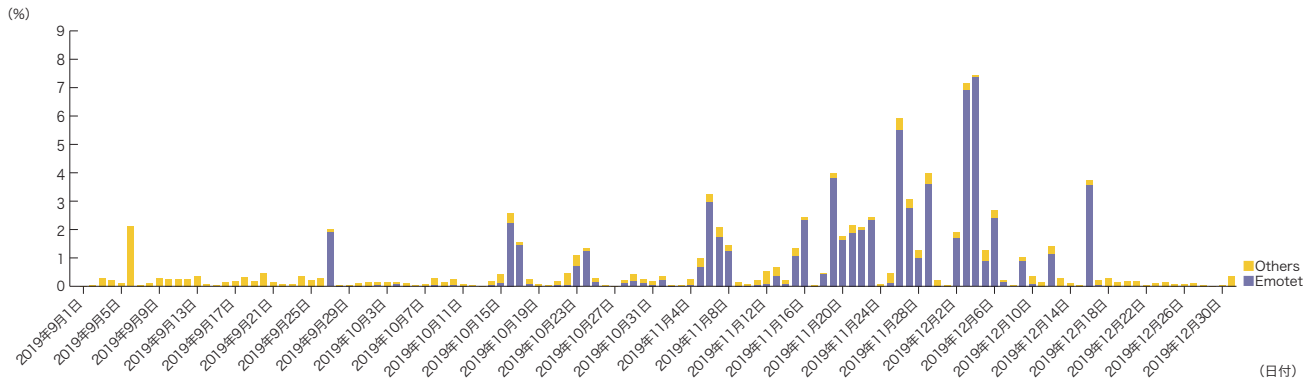

図-7 メール受信時のマルウェア検出傾向(2019年9月～12月)

ピークを迎え、その後16日のスパイクを最後に、12月中はIIJセキュアMXサービスにおける検出が落ち着きました。

しかし、それと入れ替わるように、IIJセキュアWebゲートウェイサービスでは、Emotetに関連する検出が増えています。図-8には横軸に日付、縦軸に12月中のEmotetに関連する検出合計を100%として正規化した当該通信の割合を示します。

図-7のメールでの検出が落ち着いた直後の12月17日から数日間、図-8ではEmotetに関連する検出の割合が増加しています。これらの通信は、Emotetに感染させるためのdocファイルをダウンロードする試みであることを確認しています。また、IPAからはEmotetに感染させるための不正なURLリンクを記載した日本語メールが12月10日頃より確認されたとの注意喚起[*28]が出されており、図-8に示した検出が開始した時期と一致しています。従って、図-8で検出した通信は、Emotetの拡散を目的としてメール本文に記載されたURLにアクセスしたものであると考えられます。

次に、SOCで観測したEmotetへの感染を誘導すると見られるメールのうち、件名に日本語を含むものを表-3に示します。なお、表-3では主要なものだけを取り上げており、全てのメールを網羅しているわけではない点にご注意ください。

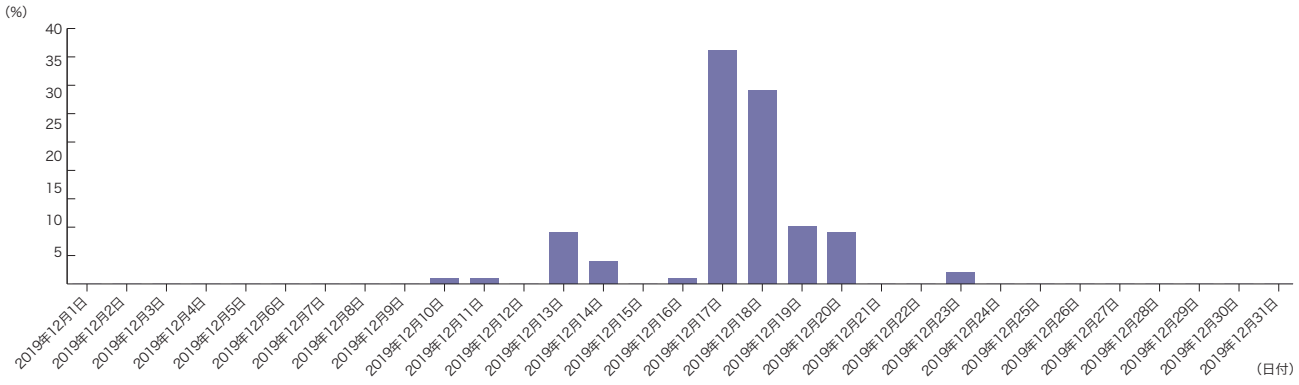

図-8 HEUR:Trojan.MSOffice.SAgent検出割合の推移(2019年12月)

表-3 件名に日本語を含みEmotetへの感染を誘導する不審メールの情報

件名	添付ファイル名	備考
12 月賞与	<日付>.doc	<日付>にはYYYYMMDDの形式でメール受信日が入ります 件名の末尾には日付、人名、組織名が入る場合があります
[本日 23:59 まで]amazon.com に更新割引クーポンが発行されました	<日付>_<ランダムな英数字>.doc	
ご入金額の通知	<ランダムな英数字> <日付>.doc	
ご請求書発行のお願い	<ランダムな英数字> _<日付>.doc	
ドキュメント	<ランダムな英数字>-<日付>.doc	
メッセージを繰り返す	賞与支払届.doc	
リマインダー	12月賞与.doc	
気づく	2019冬・業績賞与支給.doc	
最後のオプション	請求書送付のお願い <ランダムな数字>-<日付>.doc	
支払通知	メリークリスマス <日付>.doc	
助けて		
情報		
新バージョン		
請求書送付のお願い		
領収書		

*28 独立行政法人情報処理推進機構、「「Emotet」と呼ばれるウイルスへの感染を狙うメールについて」(https://www.ipa.go.jp/security/announce/20191202.html#L11)。

表-3に示したようにメールの件名には、「気づく」・「助けて」・「情報」など1単語だけの表現や請求書・領収書を装ったものなど様々なバリエーションがあります。また、ブラックフライデーのシーズンにはECサイトの割引クーポンを装った件名や、年末には件名もしくは添付ファイル名に「賞与」・「クリスマス」といった時期に合わせた単語が用いられるようになりました。

■ 対策

前述のとおり、Emotetは感染端末から窃取したメール情報を元に、返信を装うなどして感染を拡大させるためのメールを送付します。そのため、受信者が送信元メールアドレスや本文から違和感を感じたり、不審であるかどうかを判断したりすることが困難な場合があります。感染を予防し被害を最小限に抑えるためには、まず、Wordの設定を確認の上、マクロの自動実行を設定している場合には無効化します。また、問題がないと判断できないメールに添付されているファイルを不用意に開かないこと、添付ファイル内のマクロを手動で有効化しないことも重要です。更に、US-CERTでは入り口での予防として、マルウェアに利用されることのある拡張子やアンチウイルスソフトでスキャンできない形式のファイルを添付したメールの受信を禁止するポリシーも有効であるとされています[*29]。それ以外にも、適切な権限設定や送信ドメインの認証を行うことなどが推奨されています。

1.4 おわりに

本レポートでは、2019年に国内で話題となったセキュリティ事件を取り上げ、いくつかの事例をSOCで観測している情報と併せて紹介しました。本章で取り上げた事例のほかにも様々なセキュリティ脅威を日々観測しています。1.2節及び1.3節で取り上げた事件や事象に関わらず、適切に状況を把握し、対処することが重要となります。1.3.1項で取り上げたElasticsearchにおける事案などACLで対処できる内容のものもあれば、脆弱性におけるパッチの適用、1.3.3項のようなマクロの有効化を安易に実施しないという個人における対策も存在します。SOCでは、引き続き様々なセキュリティ事件や脅威について、wizSafe Security Signal（https://wizsafe.iij.ad.jp）にて定期的に情報共有しており、日々のセキュリティ業務に還元していただけたら幸いです。

執筆者：
守田 瞬（もりた しゅん）
IIJ セキュリティ本部 セキュリティビジネス推進部 セキュリティオペレーションセンター データアナリスト

執筆者：
本部 栄成（ほんぶ えいせい）
IIJ セキュリティ本部 セキュリティビジネス推進部 セキュリティオペレーションセンター データアナリスト

執筆者：
山口 順也（やまぐち じゅんや）
IIJ セキュリティ本部 セキュリティビジネス推進部 セキュリティオペレーションセンター データアナリスト

*29 CISA、"Increased Emotet Malware Activity"（https://www.us-cert.gov/ncas/current-activity/2020/01/22/increased-emotet-malware-activity）。

Windowsのメモリイメージ取得の注意点

2.1 Windowsにおけるメモリイメージ取得

本レポートVol.45では、Linuxにおけるフォレンジック向けメモリイメージ取得について解説しました[*1]。今回はWindowsのメモリイメージ取得について解説します。

Windowsでシステム全体のメモリイメージを取得するには、表-1にあるようなツールを使用します。取得したメモリイメージの解析には、Volatility Framework[*2]やRekall Memory Forensic Framework[*3]を使用します。

しかし、Windowsのバージョンアップに伴い、セキュリティやシステムパフォーマンス向上のため、メモリ管理の仕組みが変更される場合があります。従って、メモリイメージの取得や解析に使用するツールも新しい仕様に合ったものを使う必要があります。本稿では、単にメモリイメージを取得するツールを紹介するのではなく、メモリイメージの取得時や解析時に注意すべき点について解説します。また、個々のプロセスダンプを確実に取得する方法も提案します。

2.2 メモリイメージ取得及び解析時の注意点

今回は、ページングファイル、メモリ圧縮、Virtual Secure Modeの3つの機能について、対応方法を解説します。ページングファイルはWindows 10より前のバージョンから存在していましたが、その他の機能は、Windows 10がリリースされた後のアップデートで追加された機能です。

■ ページングファイル

Windowsはページアウトするプロセスの仮想メモリのページを、C:¥pagefile.sysというページングファイルに保存します。図-1はWindows 10 1809のメモリイメージからVolatilityのprocdumpプラグインでnotepad.exeを抽出しようとした結果です。この時点ではnotepad.exeが起動直後であったため、プロセスのダンプに成功しています。一方、図-2は同じ環境でメモリ使用率が高くなった後のメモリイメージから、notepad.exeをダンプしようとした結果ですが、ページアウトによってプロセスダンプが失敗しています。なお、図-1、図-2で解析に使用したメモリイメージはVMware Workstationのメモリスナップ

表-1 メモリイメージ取得ツールの例

ツール	入手先
WinPmem memory imager	https://winpmem.velocidex.com/
Comae Technologies	https://www.comae.com/
MAGNET RAM Capture	https://www.magnetforensics.com/resources/magnet-ram-capture/
Belkasoft RAM Capturer	https://belkasoft.com/ram-capturer

```
>vol.py --profile Win10x64_17763 -f "Windows 10 1809 Feb x64 en-Snapshot82.vmem" procdump --pid=4596 -D procdump
Volatility Foundation Volatility Framework 2.6.1
Process(V)         ImageBase          Name                 Result
------------------ ------------------ -------------------- ------
0xffffaf86ab950480 0x00007ff7413c0000 notepad.exe          OK: executable.4596.exe
```

図-1 Volatility procdumpの成功例

```
>vol.py --profile Win10x64_17763 -f "Windows 10 1809 Feb x64 en-Snapshot83.vmem" procdump --pid=4596 -D procdump
Volatility Foundation Volatility Framework 2.6.1
Process(V)         ImageBase          Name                 Result
------------------ ------------------ -------------------- ------
0xffffaf86ab950480 0x00007ff7413c0000 notepad.exe          Error: ImageBaseAddress at 0x7ff7413c0000 is unavailable (possibly due to paging)
```

図-2 Volatility procdumpの失敗例

＊1 Internet Infrastructure Review (IIR) Vol.45 2章フォーカス・リサーチ Linuxのフォレンジック向けメモリイメージ取得(https://www.iij.ad.jp/dev/report/iir/045/02.html)。
＊2 The Volatility Foundation(https://www.volatilityfoundation.org/)。
＊3 Rekall Forensics(http://www.rekall-forensic.com/)。

ショットです。本稿の執筆にあたり、ある程度意図した状態のメモリイメージを容易に取得するために使用しました。

ページングファイルにはページアウトされたメモリデータが保存されているため、可能であれば解析に活用したいところです。しかし、多くのメモリイメージ取得ツールは、このファイルを収集しません。WinPmemであれば、「-p」オプションでpagefile.sysを指定することで、メモリイメージとページングファイルをほぼ同時に取得することができます(図-3)。The Sleuth KitやFTK Imagerなどを使用して取得することもできますが、メモリイメージとpagefile.sysを取得する間隔はなるべく短い方が、内容の齟齬が生じにくいでしょう。

pagefile.sysを取得しても、メモリイメージ解析ツールが対応していなければ意味がありません。Rekallはメモリイメージとpagefile.sysを併せて透過的に1つのメモリイメージとして解析することができます。図-2の状態の直後に図-3のコマンドによりWinPmemでメモリイメージとpagefile.sysを取得し、Rekallのprocdumpプラグインを使って、それらのファイルからnotepad.exeをプロセスダンプすることができました(図-4)。ファイルの先頭部分を確認すると、MZヘッダであることが分かります(procdumpプラグインは指定したプロセスをPEフォーマットでダンプします)。

一方、Volatility 2は、Rekallのようにpagefile.sysを解析することはできません。しかし、現在開発中のVolatility 3に関するOSDFCon 2019での発表資料[*4]には、ページングファイルと後述するメモリ圧縮に対応することを示唆する記述があるため、今後はVolatilityでもページングファイルを使用した解析が可能になると思われます。

なお、図-3で使用したWinPmemはバージョン2.1 post4です。本稿執筆時(2020年2月)のWinPmemの最新バージョンは3.3 rc3ですが、これで生成したaff4ファイルをRekallで解析しようとすると、図-6のエラーが発生します。関連するソースコードをすべて調査したわけではありませんが、このエラー

```
>winpmem-2.1.post4.exe -p c:\pagefile.sys -o memdump.aff4
```

図-3 WinPmemでメモリイメージとpagefile.sysを取得

```
$ rekal -f memdump.aff4 procdump --proc_regex="notepad*" --dump_dir="./"
Webconsole disabled: cannot import name 'webconsole_plugin'
                    _EPROCESS                     Filename
------------------------------------------------------ --------
0xaf86ab950480 notepad.exe                    4596 executable.notepad.exe_4596.exe
```

図-4 pagefile.sysを利用したprocdumpの実行例

```
$ hexdump -C executable.notepad.exe_4596.exe | head -10
00000000  4d 5a 90 00 03 00 00 00  04 00 00 00 ff ff 00 00  |MZ..............|
00000010  b8 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00  |........@.......|
00000020  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000030  00 00 00 00 00 00 00 00  00 00 00 00 f8 00 00 00  |................|
00000040  0e 1f ba 0e 00 b4 09 cd  21 b8 01 4c cd 21 54 68  |........!..L.!Th|
00000050  69 73 20 70 72 6f 67 72  61 6d 20 63 61 6e 6e 6f  |is program canno|
00000060  74 20 62 65 20 72 75 6e  20 69 6e 20 44 4f 53 20  |t be run in DOS |
00000070  6d 6f 64 65 2e 0d 0d 0a  24 00 00 00 00 00 00 00  |mode....$.......|
00000080  15 48 94 65 51 29 fa 36  51 29 fa 36 51 29 fa 36  |.H.eQ).6Q).6Q).6|
00000090  58 51 69 36 4f 29 fa 36  34 4f f9 37 52 29 fa 36  |XQi6O).640.7R).6|
```

図-5 Rekall procdumpでダンプしたファイル

```
$ rekal -f winpmem3.aff4 pslist
Traceback (most recent call last):
  File "/home/user01/Downloads/rekall/rekall-core/rekall/addrspace.py", line 519, in read_partial
    data = self._cache.Get(chunk_number)
  File "/home/user01/Downloads/rekall/rekall-lib/rekall_lib/utils.py", line 147, in NewFunction
    return f(self, *args, **kw)
  File "/home/user01/Downloads/rekall/rekall-lib/rekall_lib/utils.py", line 336, in Get
    raise KeyError(key)
KeyError: 0
(snip)
  File "/home/mkobayashi/envs/win10_rekall2/lib/python3.6/site-packages/pyaff4/aff4_image.py", line 432, in _ReadChunkFromBevy
    "Unable to process compression %s" % self.compression)
RuntimeError: Unable to process compression https://tools.ietf.org/html/rfc1951
```

図-6 RekallはWinPmem 3.xで取得したaff4ファイルの解析時にエラーが発生する

*4 Volatility 3 Public Beta: The Insider's Preview(https://www.osdfcon.org/events_2019/volatility-3-public-beta-the-insiders-preview/)。

は、WinPmem 3.3 rc2以降の保存データのデフォルト圧縮フォーマットがdeflateに変更され、Rekallがそれに未対応であることが原因のようです（WinPmem 2.xのデフォルトはzlibでした）。WinPmemの「-c」オプションで圧縮フォーマットを指定できますが、zlibを指定しても同様のエラーで解析することはできませんでした。なお、WinPmem 3.3 rc1以前のWinPmem 3.xもzlibがデフォルトの圧縮フォーマットですが、これらのバージョンでページングファイルを含むaff4ファイルを生成し、Rekallで解析した場合、上記とは異なるエラーが発生します（ページングファイルを含まないaff4ファイルは解析できます）。

従って、解析ツールとしてRekallを選択する場合、WinPmem 2.1 post4を使用する方が解析時に問題が起きにくいメモリイメージを作成することができます。しかし、WinPmem 2.xは開発が終了していること、Windows 10環境では強制終了してしまう場合があること、また、後述するVirtual Secure Modeに対応していないこともあり、使用はお勧めできません。なお、Rekallも2017年12月以降、新しいバージョンがリリースされておらず、事実上の開発停止状態にあります。

今後、VolatilityやRekallの開発が進めば、WinPmem 3.xで生成したaff4ファイルも解析できるようになることが期待できますので、それまではWinPmem 3.xでメモリイメージとページングファイルを取得し、図-7のようにエクスポートして解析するのがベターな対応でしょう。エクスポートしたメモリイメージはRAWフォーマットなので、VolatilityとRekallのどちらでも解析することができます。

```
>winpmem_v3.3.rc3.exe -dd -e */PhysicalMemory -D <export_dir> <image_file>.aff4
```

図-7 WinPmem 3.xで作成したaff4ファイルから メモリイメージをエクスポート

■ メモリ圧縮

プロセスの仮想メモリのページングにはページングファイルの読み書きが発生するため、その分、どうしてもシステムのパフォーマンスが低下してしまいます。近年はSSDが普及したため、HDDほどのレイテンシは発生しませんが、それでもパフォーマンスが低下することに違いはありません。そこで、メモリ上に専用の領域を設け、そこにページアウトされるページを圧縮して保存することで、ページイン及びページアウト時のパフォーマンスの向上が図られました。圧縮されたページの容量はタスクマネージャのパフォーマンスタブ内にあるメモリの項目で確認できます（図-8中の赤枠）。このフレームワークは、Windows 10 1511以降で採用されています。同様のフレームワークは、macOSやLinuxにも存在します。

メモリ圧縮のデータが含まれたメモリイメージの解析には、当然ながら、それに対応した解析ツールが必要になります。残念ながら、現状のVolatility 2やRekallでは各OSのメモリ圧縮のデータを他のメモリページと併せて、透過的に解析することはできません。

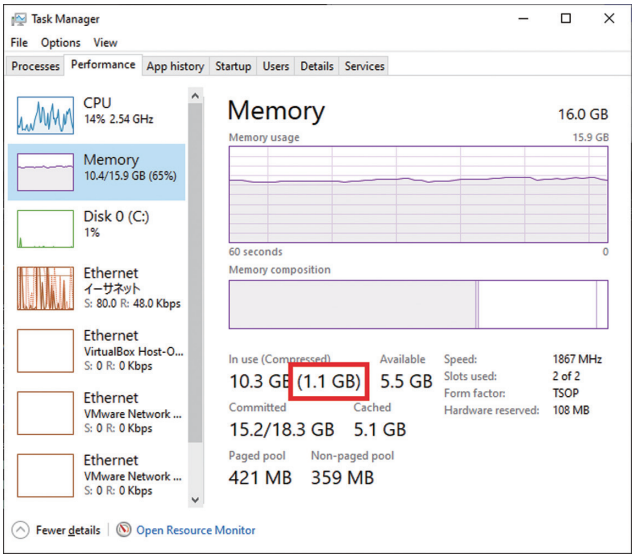

図-8 圧縮されたメモリ容量

しかし、FireEye社のOmar Sardar氏とDimiter Andonov氏がSANS DFIR Summit Austin 2019[*5]及びBlackHat USA 2019[*6]で、Windows 10のメモリ圧縮に対応したVolatility[*7]とRekall[*8]の実装を発表しました。

なお、両実装ともWindows 10 1607から1809までしかサポートしていませんので、Windows 10 1903以降のメモリイメージを解析することはできません。本稿執筆時点で、発表の成果が開発元のソースコードに取り込まれる様子はありませんが、上記したようにVolatilityは新しいバージョンで対応する予定があるようです。

図-9と図-10は、それぞれオリジナルのVolatilityとメモリ圧縮に対応したVolatilityのhashdumpプラグインの実行結果です。hashdumpプラグインはメモリに読み込まれたレジストリハイブから、ユーザのパスワードハッシュを取得します。ユーザパスワードのハッシュがメモリ圧縮に保存されているため、オリジナルのVolatilityでは実行結果に何も出力されていませんが、メモリ圧縮に対応したVolatilityではハッシュが出力されています。

■ Virtual Secure Mode

Windows 10 1511以降のEnterpriseエディションとEducationエディション、及びWindows Server 2016以降では、Virtualization Based Security(VBS)と呼ばれる仮想マシンを使用する分離機構が導入されています。Device GuardやCredential Guardといったセキュリティ機構は、VBSを利用し、Virtual Secure Mode(VSM)と呼ばれる特定機能用の仮想マシンを実行することで実装されています。これらの機能により、ドライバのロード時の検証やアプリケーションの実行制限、また、認証情報を保護することができます。

WimPmem 2.xのような、VSMに対応していないツールを実行すると、図-11のようにブルースクリーンが発生します。

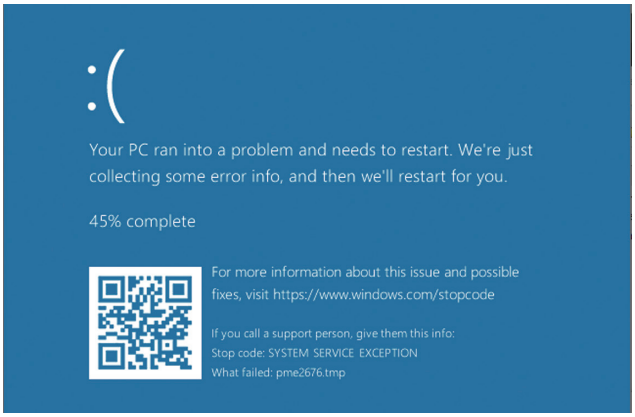

図-11 VSM未対応のメモリイメージ取得ツールを実行するとブルースクリーンが発生する

```
> vol.py --profile Win10x64_17763 -f "Windows 10 1809 Feb x64 en-Snapshot64.vmem" hashdump
Volatility Foundation Volatility Framework 2.6.1
```

図-9 オリジナルのVolatilityのhashdumpプラグインの実行結果

```
> vol.py --profile Win10x64_17763 -f "Windows 10 1809 Feb x64 en-Snapshot64.vmem" hashdump
Volatility Foundation Volatility Framework 2.6.1
localuser01:1001:8492e81f418ee4da82b19ef1f27d39af:17e26cdcb1cf786246e7cf8373a540ca:::
```

図-10 メモリ圧縮対応VolatilityのhashdumpプラグインのIn実行結果

*5 SANS DFIR Summit 2019 (https://www.sans.org/event/digital-forensics-summit-2019/summit-agenda)。
*6 Paging All Windows Geeks ? Finding Evil in Windows 10 Compressed Memory(https://www.blackhat.com/us-19/briefings/schedule/#paging-all-windows-geeks--finding-evil-in-windows--compressed-memory-15582)。
*7 win10_volatility(https://github.com/fireeye/win10_volatility)。
*8 win10_rekall(https://github.com/fireeye/win10_rekall)。

ブルースクリーンが起きた場合、デフォルトではホストは自動的に再起動されます。当然ながらメモリの内容はすべて消えてしまうため、使用しているツールがVSMに対応しているか事前に検証しておかなければなりません。なお、表-1のツールの最新バージョンであればブルースクリーンは発生しませんので、古いバージョンを使用している場合はアップデートを検討してください。

■ どのメモリイメージ解析ツールを使うべきか

ここまでに解説したメモリイメージ解析ツールが対応しているデータの種類を表-2にまとめました。この中で開発が継続されているのはVolatility 2のみであるため、基本的にこれを使用することをお勧めします。ただし、Windows 10 1809以前のメモリイメージを解析する場合は、メモリ圧縮対応版Volatilityを使用する方がよいでしょう。

ページングファイルも解析したい場合、Rekallを使う必要がありますが、開発が停止しており、また、WinPmem 3.xとの互換性の問題もあるため積極的に使う場面はあまりないと思われます。残念ながら、あらゆるケースに対応できるツールは現状ありません。しかし、Volatility 3がリリースされれば、この状況は改善されると思われます。

2.3 プロセスを確実にダンプする

ここまで、メモリイメージを取得する際の注意点とその対策を解説してきました。しかし、このような対策を行っても、整合性のとれたメモリイメージの取得は難しいのが現状です。解析対象ホストが仮想マシンであれば、スナップショットを取ることで、その時点の完全な状態のメモリイメージを取得することができます。しかし、ライブシステムのメモリイメージを取得する場合、その最中にも様々なプロセスが動作しているため、メ

表-2 メモリイメージ解析ツール比較

解析ツール	メモリイメージ		ページングファイル	メモリ圧縮	備考
	RAW	AFF4			
Volatility 2	✓				
Rekall	✓	✓(※1)	✓(※2)		開発停止
メモリ圧縮対応版 Volatility	✓			✓	Windows 10 1809まで対応
メモリ圧縮対応版 Rekall	✓	✓(※1)	✓(※2)	✓	Windows 10 1809まで対応

(※1) WinPmem 3.3 rc2 以降で作成した AFF4 ファイルは解析できない。
(※2) WinPmem 3.x で AFF4 ファイルに含めたページングファイルは解析できない。

モリ中のデータ変更やページアウトなどが発生します。このため、メモリイメージを解析しても、プロセスのメモリ内容の整合性がとれていない可能性があります。

図-12は図-1でダンプしたファイル(左側)と図-4でダンプしたファイル(右側)の.textセクションを比較しているところです。図-5で示したように、Rekallでダンプしたファイルの MZヘッダは正しく抽出されましたが、両ファイルの.textセクションを比較するとRekallでダンプしたファイルではデータが0x00になってしまっていることが分かります(図-12の赤い領域)。上記のとおり、このような状態は致し方ないのですが、プロセスを解析する際に障害となる可能性があります。このような場合、ユーザランドよりプロセスを個別にダンプすることで整合性のとれたプロセスダンプを取得することができます(プロセスダンプの際にメモリアクセスが誘発されページアウトしているページがOSによりページインされるため、プロセスの仮想メモリのページをすべて取得することができます)。

プロセスダンプを行うためのツールはいくつかありますが、代表的なツールはWindows Sysinternals ProcDump[*9]です。VolatilityやRekallにも、同じ名前のプラグインがありますが、それらとは異なります。また、VolatilityとRekallのprocdumpはPEフォーマットのファイルを出力しますが、Sysinternals ProcDumpはクラッシュダンプフォーマットのファイルを出力します。

図-13のように実行すると、プロセスIDが4596のプロセスをダンプすることができます。プロセスIDの代わりにプロセス名を指定することもできます。また、プロセスが使用するすべてのメモリをダンプする「-ma」オプションも指定すると良

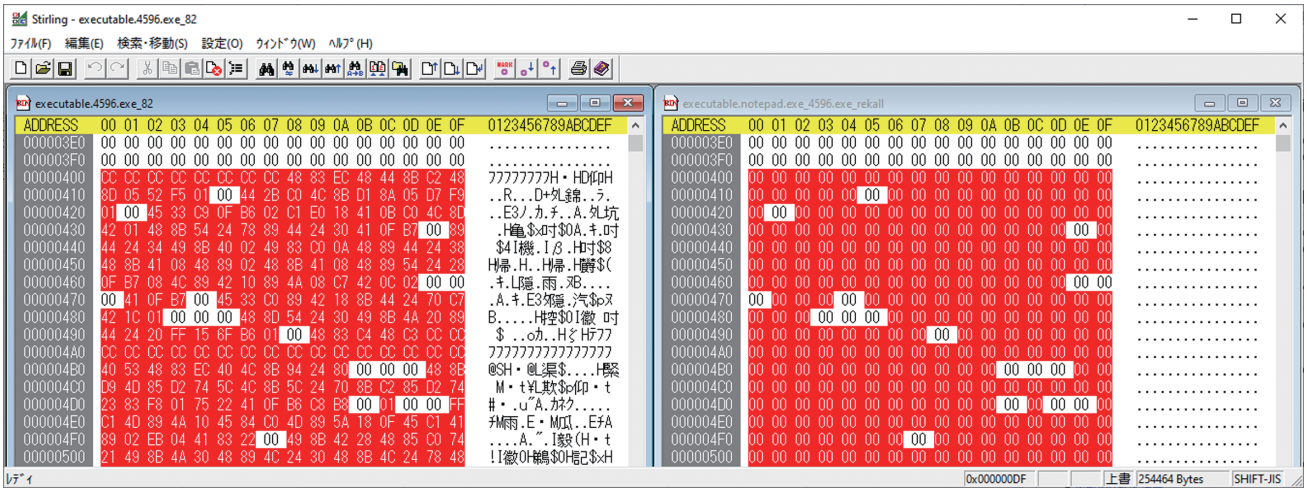

図-12 プロセスダンプの差異

```
>procdump64.exe -ma 4596

ProcDump v9.0 - Sysinternals process dump utility
Copyright (C) 2009-2017 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com

[12:05:29] Dump 1 initiated: C:\Users\localuser01\Desktop\tools\notepad.exe_200206_120529.dmp
[12:05:29] Dump 1 writing: Estimated dump file size is 107 MB.
[12:05:32] Dump 1 complete: 107 MB written in 3.8 seconds
[12:05:33] Dump count reached.
```

図-13 ProcDumpコマンドでプロセスをダンプする

*9 ProcDump - Windows Sysinternals(https://docs.microsoft.com/en-us/sysinternals/downloads/procdump)。

いでしょう。生成されたクラッシュダンプはWinDbg[*10][*11]などのツールで読み込むことが可能です。図-14は、図-4の後にProcDumpで生成したクラッシュダンプをWinDbgで読み込んだスクリーンショットです。Rekallのprocdumpで欠けていたコードも含まれていることが分かります（図-14中の赤枠）。このように、ProcDumpを使用すれば、確実にプロセスダンプできます。

2.4 プロセスダンプのスクリプト化とアーティファクト保全の順番

PowerShellなどでスクリプトを作成すれば、容易にすべてのプロセスに対してProcDumpを実行することができます。ただし、すべてのプロセスをダンプすると数十GB以上の容量になります。データ保全先の容量に余裕がある場合は問題ありませんが、容量が小さなUSBメモリや外付けSSDに保全する場合はダンプを取得した後に7-Zip[*12]などで随時、圧縮を行う方が良いでしょう。また、ファイル圧縮時に一時ファイルが作られますので、それらが調査対象のディスクに書き込まれないよう、圧縮コマンドのオプションなどで指定する必要があります。7-Zipの場合、「-w」オプションで作業フォルダを指定できますので、アーティファクトを保全するディスクを指定すると良いでしょう。

ProcDumpを実行する際、SystemやRegistryなどの保護されたプロセスはスキップするようにします（プロセスダンプしようとしてもエラーになります）。また、このようなスクリプトは様々な環境で動作しなければならないため、Windowsにデフォルトでインストールされ ているPowerShellやバッ

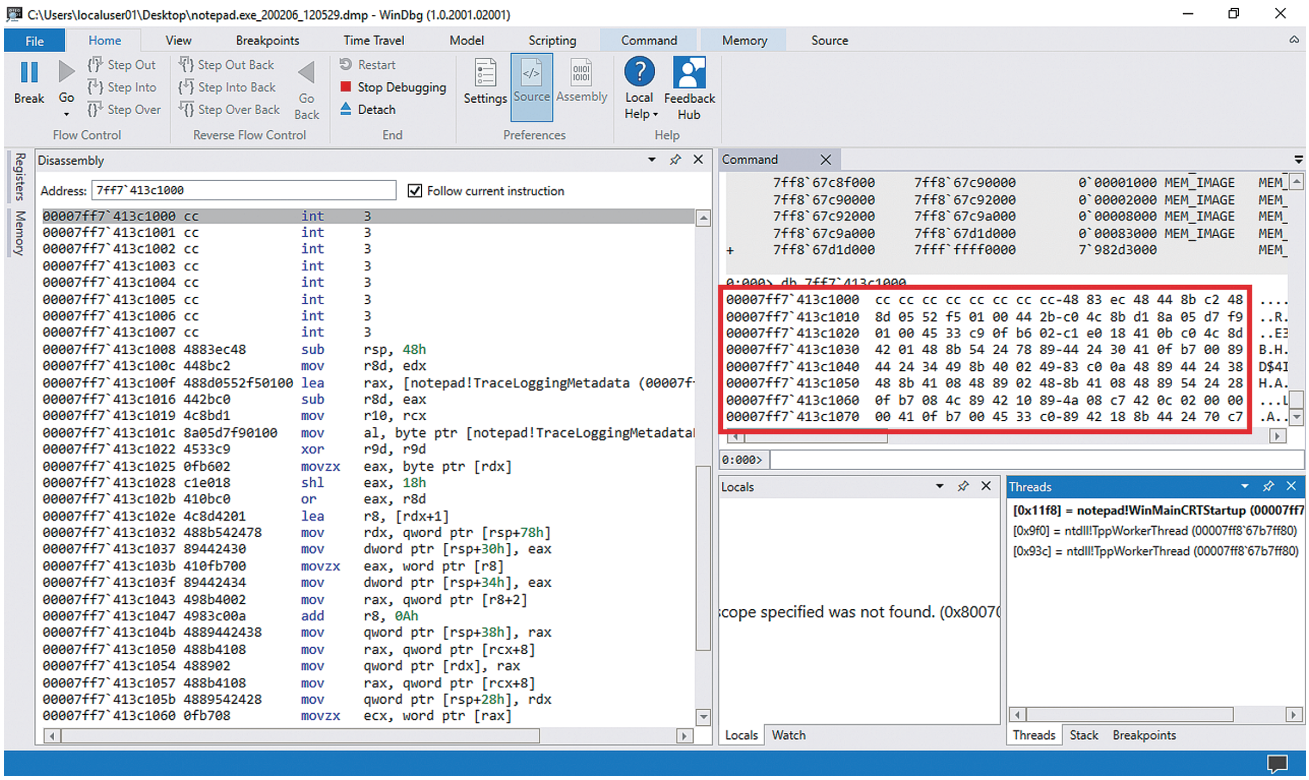

図-14 WinDbgでプロセスダンプを確認

*10 Download Debugging Tools for Windows - WinDbg（https://docs.microsoft.com/en-us/windows-hardware/drivers/debugger/debugger-download-tools）。
*11 Debugging Using WinDbg Preview（https://docs.microsoft.com/en-us/windows-hardware/drivers/debugger/debugging-using-windbg-preview）。
*12 7-Zip（https://www.7-zip.org/）。

チファイルで作成することが多いと思いますが、注意が必要な点があります。

PowerShlellのプロンプト(powershell.exe)やコマンドプロンプト(cmd.exe)を起動すると、コンソールウィンドウホスト(conhost.exe)も起動します。ProcDumpを実行するPowerShellスクリプトやバッチファイルを起動したプロンプトに対応するconhost.exeをダンプしようとすると、ProcDumpの処理が停止してしまいます。これは、ProcDumpがプロセスダンプを行う際、該当のプロセスをサスペンドさせることが原因です。conhost.exeはコンソールの入出力バッファや表示などを処理するプロセスであるため、このプロセスをサスペンドすることで、それを利用しているProcDumpも停止してしまいます(図-15)。必要であればWinPmemなどで取得したメモリイメージで、conhost.exeを解析することが可能です。

スクリプトを実行すると環境によっては数百回以上、プロセスダンプとファイル圧縮を行うことになります。従って、フォレンジックの観点から考えると、一例として、図-16の順番でアーティファクトの保全を図るのが良いと考えられます。

2.5 まとめ

今回はWindowsでメモリイメージの取得と解析を行う際の注意点を解説しました。また、メモリイメージ取得時の整合性を補完する観点でプロセスダンプにも触れました。メモリフォレンジック関連の記事では、WinPmemなどを使ってメモリイメージを取得することで保全を完了とするものが多いですが、それだけでは十分ではない場合があることを理解いただけたのではないかと思います。ただし、すべてのプロセスのダンプと圧縮には時間がかかりますので、インシデント対応時の状況やポリシーに応じて、実施の可否を検討する必要があります。

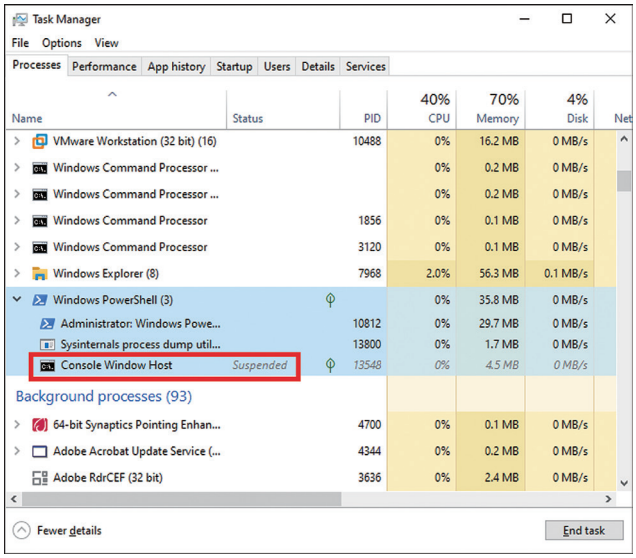

図-15 conhost.exeをダンプするとProcDumpが停止

1. メモリイメージ
2. MFTやPrefetch、イベントログなどのディスクからファイル単位で保全できるアーティファクト
3. プロセスダンプ
4. ディスクイメージ

図-16 アーティファクトを保全する順番の例

執筆者:
小林 稔(こばやし みのる)
IIJセキュリティ本部セキュリティ情報統括室フォレンジックインベスティゲーター。
IIJ-SECTメンバーで主にデジタルフォレンジックを担当し、インシデントレスポンスや社内の技術力向上に努める。
Black HatやFIRST TC、JSAC、セキュリティキャンプなどの国内外のセキュリティ関連イベントで講演やトレーニングを行う。

解析対象への前提知識を必要としないバイナリプログラム解析技術

3.1 はじめに

本稿ではIIJ-II技術研究所で行っている、解析対象に対しての前提知識を必要としないバイナリプログラム解析技術について解説します。

プログラム解析とは、対象となるプログラムがどのような振る舞いを行うのかを分析する技法で、大別すると実際にプログラムを実行してその挙動を調べる「動的解析」と、プログラムを実行せずにプログラムの構成を解析する「静的解析」に分けられます。

動的解析の例としては、プログラムの各機能の整合性をチェックする単体テストや、ランダム生成された入力データを与えて挙動をテストするファジングツールなどがあります。静的解析の例としては、不要となる処理を排除したり、事前に計算可能な処理を行ったりといったやり方で実行時の計算処理効率を高める最適化解析や、プログラムがデータを意図しない扱いをすることで実行時に起こるエラーを回避するために、データ型の整合性を保証する静的型検査などがあります。動的静的を問わず、こうした様々なプログラム解析技術は統合開発環境(IDE)に組み込まれて、プログラム開発の効率化やバグの削減などに役立っています。

このようにプログラム解析技術は、主に開発する側が使用することが前提となっています。一方で、既に作成され配布されたプログラム(バイナリプログラム)に対して、その挙動の解析が求められる場合があります。例えば、マルウェアの疑いのあるプログラムの振る舞いを知りたい場合や、サードパーティから提供されたファームウェアが不審な挙動を起こさないか調べたい場合などが挙げられます。このような場合、対象となるプログラムのソースコードや、どのようなコンパイラを用いて作成されたかといった情報が、解析する側に必ずしも与えられているとは限りません。そのような場合でも動的解析は可能です。例えば隔離されたサンドボックス環境でマルウェアの疑いのあるプログラムを実行し、その挙動を解析する検疫システムなどが実用的に運用されています。しかし、動的解析の問題点として、解析できるのは実際に実行された制御パスだけである、ということが挙げられます。そのため、実行環境を判定して振る舞いを変更するアンチ解析マルウェアや、特定の入力に応じて挙動を変更するバックドアが仕込まれたファームウェアなどの解析は動的解析だけでは難しいと言えます。

従って網羅的に挙動を調べるためには、動的解析だけではなくバイナリプログラムを対象とした静的解析が必要とされています。バイナリプログラムの静的解析では、そのプログラムがどのように作成されたかなどの前提知識がどの程度与えられているかによって、その困難さが変わってきます。例えば、バイナリプログラムを作成するのに使用されたコンパイラが推定できる場合、そのコンパイラが生成するコードパターンに基づいて、元のソースコードを復元できる場合があります。このような手法を「逆コンパイル」と呼びます。逆コンパイルができれば、復元されたソースコードに既存の静的解析手法を適用することで、プログラムの振る舞いを解析することができます。

しかし、このような前提知識が常に得られる、あるいは、得られたとしても信用できるとは限りません。本研究ではそのような前提知識をほとんど得られない、つまり「得体の知れない」バイナリプログラムであっても解析が行える静的解析技術の開発を行っています。

次節以降では静的解析でバイナリプログラムを対象とすることの難しさ、そして前提知識の有無によって、なぜ更に困難さが増すのかを解説します。

3.2 バイナリプログラム解析

C/C++などのプログラミング言語で記述されたプログラム(「ソースコード」と呼ばれます)は、コンパイラと呼ばれるソフトウェアによって、CPUが処理できる単純な機械命令の列に変換されます。この機械命令の列は、CPUによって定められた符号化方法によって整数値バイトデータへとエンコードされます。こうしてバイトデータの列として表現されたプログラムを「バイナリコード」と呼びます。

作成されたバイナリコードは「実行ファイル」と呼ばれるファイル形式の中に埋め込まれています。実行ファイルにはバイナリコードの他に、実行時にメモリのどの位置にプログラムを配置するか、配置されたプログラムのどこから実行を開始(エントリアドレス)するか、そして、実行中にどのような外部ライブラリ関数を呼び出すか、などのメタ情報も記述されています。このような実行ファイルを本稿ではバイナリプログラムとして扱います。ソースコードを用いずバイナリプログラムのみを用いて行う静的解析を「バイナリコード解析」と呼びます。

身近な例で言えば、標準的なウイルス検知ソフトウェアでは、既知のマルウェアからシグネチャと呼ばれる特徴的なバイトデータ列を抽出したデータベースを利用して、バイナリコード内に既知のシグネチャが含まれているかどうかを判定しています。近年では実行ファイルに含まれるその他のメタ情報を含めて機械学習を行い、未知のマルウェアの検出を行うシステムもあります。

このようにバイナリコードを単なるバイトデータとして解析する手法はマルウェア検知のようにプログラムの自動分類を行うことには適していますが、プログラムがどのような挙動をするか詳細に知りたい場合にはバイナリコードを単なるデータ列としてではなく、プログラムとして解析する必要があります。このような解析を本稿では「バイナリプログラム解析」と呼びます。

バイナリプログラム解析ではバイナリコードからプログラムの制御構造を抽出して再構成する必要があります。そのための第一歩として「逆アセンブラ」という手法があります(図-1)。前述のように、各機械命令はアーキテクチャごとに定められた規則に従いバイトデータに変換されますが、これを逆向きに行い、バイトデータから機械命令の表現へ変換することを逆アセンブルと言います。

単純な逆アセンブラではバイナリコードの先頭から順次逆アセンブルする「線形走査型」という手法が利用されます。線形走査型の場合、途中にプログラムではないデータが混入していると、その時点から正確な逆アセンブラができなくなる可能性があります。一方、IDA Proなどの高度な逆アセンブラでは、エントリアドレスから再帰的に直接ジャンプ命令(ジャンプの行き先アドレスが命令中に実値で指定されている制御命令)を辿る「再帰探索型」という手法が用いられています(図-2)。再帰探索型の逆アセンブラでは、間接ジャンプ命令(ジャンプの行き先アドレスがレジスタやメモリに格納されている制御命令)に行き当たった場合は、それ以上の探索を行いません。これは一般的に間接ジャンプ命令の行先アドレスを静的に解決することが決定不能問題であり理論的に困難なためです。

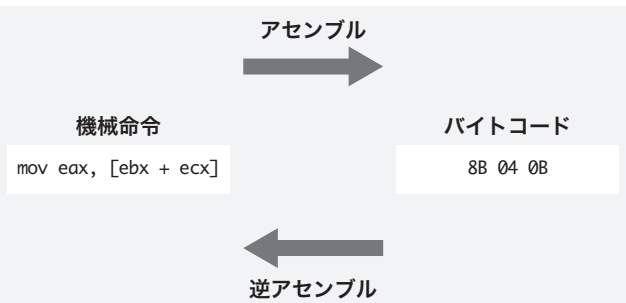

図-1 逆アセンブラ

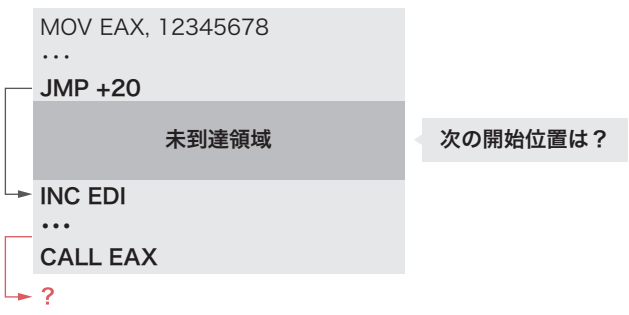

図-2 再帰探索型逆アセンブラ

よって、再帰的探索では直接ジャンプ命令だけでは辿れない未到達領域ができます。IDA Proなどの逆アセンブラは様々なヒューリスティクスを用いて、この未到達領域から適切な開始位置を推定して再帰的探索を再開します。そのようなヒューリスティクスの1つに「関数位置同定」と呼ばれる手法があります。

一般的にプログラム開発においては、「関数」もしくは「手続き」と呼ばれる独立した機能に分割して組み合わせることで、再利用性を高め、開発効率を高めることが行われています。各関数にどのように引数(パラメータ)を渡し、また計算結果を返り値としてどのように受け取るかは、CPUやオペレーションシステムなどによって「呼出規約」として定められています。内部関数がコンパイルされる場合には、この呼出規約に応じて必要な前処理と後処理が挿入されます。これらの処理はコンパイラによって固有のパターンがあるので、このパターンを見つけ出すことで、内部関数の位置を推定することができます。このような解析を関数位置同定と呼びます。

関数位置同定によってプログラムを関数単位に分割し、関数ごとに再帰探索を行うことで、プログラム全体の逆アセンブルが行えます(図-3)。

関数位置同定によって関数の開始位置を特定するためには、プログラムが呼出規約に従ってコンパイラで作成されていることが前提となっています。この前提が満たされない場合、逆アセンブラでは正確なプログラムの構造が再構成できなくなります。また、たとえコンパイラで生成されたプログラムであっても、強い最適化を行うと関数内の前処理や後処理のコードパターンが省略され、関数位置の同定が困難になる場合があることが報告されています[*1]。

逆アセンブラだけでプログラムの構造を再構成できない理由の1つは間接ジャンプ命令です。間接ジャンプ命令の行先アドレスを、静的データ解析を用いてできる限り静的に解決する解析手法を「制御フロー再構成」と呼びます。前述のとおり、関節ジャンプ命令の行先を静的に解決することは決定不能問題なので、完全解は得られません。CodeSurfer/x86[*2]やJakstab[*3]などの先行研究では抽象解釈(Abstract interpretation)と呼ばれる解析手法を用いて行先アドレスの近似解を求めています。

しかし、制御フロー再構成にはもう1つ困難な点があります。前述のとおり、プログラムは複数の関数によって構成されています。事前に関数位置同定が行われてプログラムが関数単位に分割されている場合は、関数ごとに独立して解析を行う手続内プログラム解析(Intra-procedural program analysis)を適用できます。十分な前提知識がなく関数位置同定が正確に行えない場合は、プログラム全体を解析する全体プログラム解析(Whole program analysis)を行うことになります。事前に関数位置の同定はできていなくても実際にはプログラムはいくつかの関数に分割されています。そのため、全体プログラム解析では「文脈依存性」を考慮する必要があります。

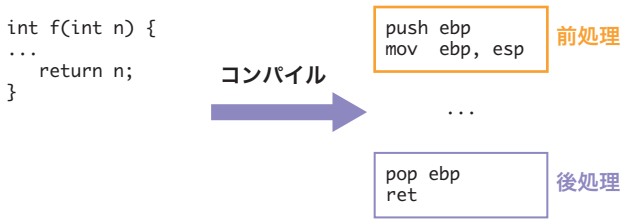

図-3 関数の前処理／後処理

*1 Andriesse2016:Andriesse, Dennis, et al. “An in-depth analysis of disassembly on full-scale x86/x64 binaries.” 25th {USENIX} Security Symposium ({USENIX} Security 16). 2016.
*2 Balakrishnan2005: Balakrishnan, G., Gruian, R., Reps, T., & Teitelbaum, T. (2005, April). CodeSurfer/x86—A platform for analyzing x86 executables. In International Conference on Compiler Construction (pp. 250-254). Springer, Berlin, Heidelberg.
*3 Kinder2008:Kinder, J., & Veith, H. (2008, July). Jakstab: A static analysis platform for binaries. In International Conference on Computer Aided Verification (pp. 423-427). Springer, Berlin, Heidelberg.

例えば複数のプログラム位置から呼ばれる関数があった場合、制御はそれぞれの関数呼出元から関数へ処理が移り、関数の処理が終わった後、それぞれの復帰先へと処理が戻ります。復帰後の状態から関数を呼び出す前の状態を参照するためには、呼出と復帰の経路が一致している必要があります。このように経路に解析が依存することを文脈依存性と呼びます。文脈依存性を考慮しない場合、複数ある呼出元の区別がつかないため、復帰後の状態を分析するときに無関係な文脈の情報が混入し、解析の精度が著しく低下してしまいます。文脈依存性を解決するためには、スタックを用いて関数呼出/復帰の対応の整合性を保証するなどの処理が必要になります。このような解析を「手続間プログラム解析(Inter-procedural program analysis)」と呼びます。

従って事前に関数位置が同定できていない場合においても精度の高いバイナリプログラム解析を行うためには、解析の過程において関数呼出/復帰の位置を推定する必要があります。既存の静的解析手法では、最低限の呼出規約(例えばIntel x86アーキテクチャではCALL命令を関数呼出、RET命令を関数復帰に使用しているなど)を仮定することで関数の位置を推定しています。

しかし、対象となるプログラムに前提知識がない場合、この最低限の呼出規約ですら保証することはできません。例えばCALL/RET命令を関数呼出/復帰以外に使用したり、逆にこれらの命令を他の命令に置き換えている可能性もあります。このように既存の静的解析手法をバイナリプログラム解析に適用するためには、対象となるプログラムの「素性の良さ」を保証する前提知識が必要になります。

これまでに述べてきたバイナリプログラム解析の困難さをまとめると以下のようになります。

1. 逆アセンブラでは関節ジャンプ命令の行先が分からない。
2. 関節ジャンプ命令の行先を既存の静的解析で決定するためには、事前にプログラムが関数に分割されている必要がある。
3. バイナリプログラムから関数位置を特定するには、プログラムがどのようなコンパイラを使用したか、呼出規約に従っているかなどの前提が必要になる。

こうしたことから、既存のバイナリプログラム解析は対象となるプログラムに対して作成条件などの前提知識があること、そしてそれらが信頼できることを必要としています。

本研究手法[*4]では、独自に考案した中間表現を用いた解析により、制御構造の再構成を行いながら、同時に関数と見なせるプログラム部分の特定を行うことで、対象に対する前提知識がなくても(つまり「素性の悪い」プログラムであっても)、静的プログラム解析が適用できることを目標としています。

*4 Izumida2018:Izumida, T., Mori, A., & Hashimoto, M. (2018, January). Context-Sensitive Flow Graph and Projective Single Assignment Form for Resolving Context-Dependency of Binary Code. In Proceedings of the 13th Workshop on Programming Languages and Analysis for Security (pp. 48-53).

3.3 射影的単一代入形式を用いたバイナリプログラム解析

ここでは本研究手法の概要を説明します。例として、32ビットIntel X86アーキテクチャ用の図-4のようなプログラムを用います。

[具体例]

```
00401000:   xor ecx, ecx
00401002:   mov ebx, 0x40100c
00401007:   jmp 0x401017
0040100c:   mov ebx, 0x401016
00401011:   jmp 0x401017
00401016:   hlt

00401017:   inc ecx ; (A)
00401018:   jmp ebx
```

図-4 32ビットIntel X86アーキテクチャにおける具体例

図-4の例では、関数に相当するコード(A)が2度呼び出されていますが、CALL/RET命令を用いている代わりに帰りアドレスをEBXレジスタに格納してから(A)へジャンプを行い、ECXレジスタに1を加えた後にEBXレジスタを行き先に指定してジャンプを行い呼び出し後のアドレスに復帰しています。標準的な関数呼出のようにCALL/RET命令を使用していないため、IDA Proなどの既存の解析ツールではこれらのコードが関数呼出であることを認識できません(図-5)。

本研究手法では、各機械命令を単純な代入文列に変換した後、更に「静的単一代入(Static Single Assignment、SSA)」と呼ばれる表現形式に変換します。SSAはコンパイラの最適化解

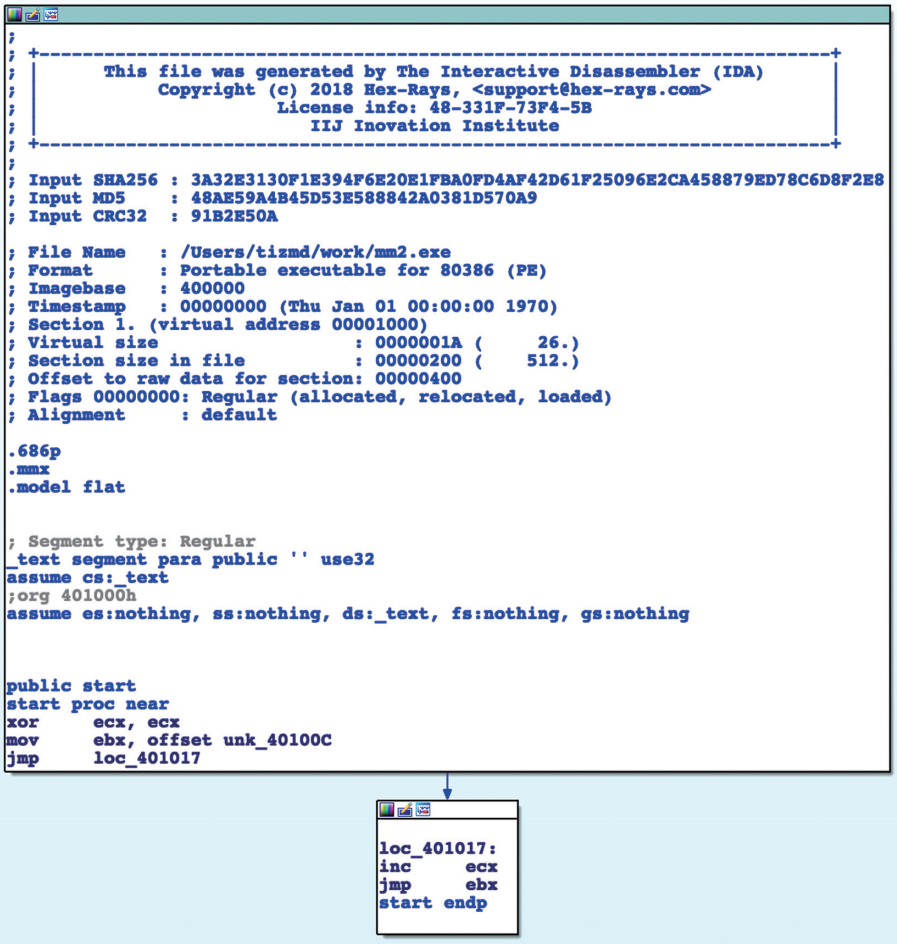

図-5 *IDA Pro*で逆アセンブルした例

析で用いられる内部表現形式で、各変数の定義が一意になるように変数名の変更を行います。これにより各変数の定義と使用(def-use)関係が明確になり、情報の流れを把握することが容易になります。例えば図-6では、2ヵ所あるECXへの代入をECX_1、ECX_4と区別しています。

ここではEBXレジスタに保存されている帰りアドレスへの間接ジャンプ命令で終了しています(ジャンプ命令はプログラムカウンタEIPへの代入として表現されています)。この場合、右辺のEBX_2の定義を遡ると0x40100cであることが容易に分かります。従って、この間接ジャンプ命令の行き先アドレスを0x40100cとして、更に展開します。

図-7は2度目の(A)の呼び出しを終えるところまでの図です。SSAではΦ関数と呼ばれる疑似関数を導入して情報の合流を表現します。例えば、$EBX_8 \leftarrow \Phi_8(3{:}EBX_2, 7{:}EBX_6)$という代入文ではノード3から来たEBXレジスタの値EBX_3とノード7から来た値EBX_6が合流して新たにEBX_8という変数に代入されています。先ほど同様にEBX_2の定義を遡ると、EBX_2はΦ関数を用いて、$\Phi_8(3{:}0x40100c, 7{:}0x401016)$として表現されます。これはすなわち、制御がノード8にノード3から来た場合のEBXレジスタの値は0x40100c、制御がノード7から来た場合は0x401016と情報の合流があることを表しています。このように特定の時点で合流した情報によって行き先アドレスが変わる場合を、本研究では「文脈依存性」として定義します。この場合、0x401017から0x401018の範囲のコードは、複数の文脈から再利用されているので、ここが「関数」であると推定することができます。

こうして文脈依存性が検出できた場合、本手法では更にΠ関数と呼ばれる疑似関数を挿入します(図-8)。Π関数はΦ関数に対する射影関数として働きます。例えば$\Pi_{3\to8}(\cdots)$という式

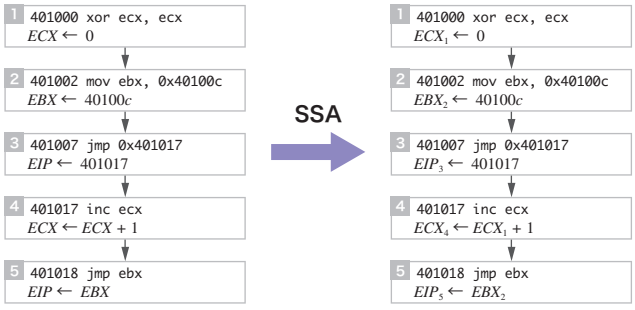

図-6 SSA形式:最初の(A)の呼出を終えるところまで

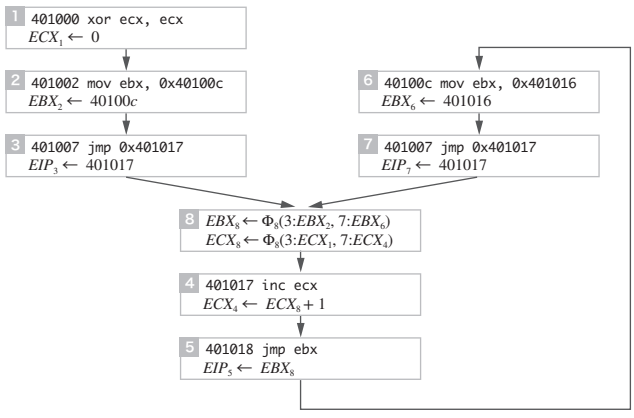

図-7 SSA形式: 2度目の(A)の呼出を終えるところまで

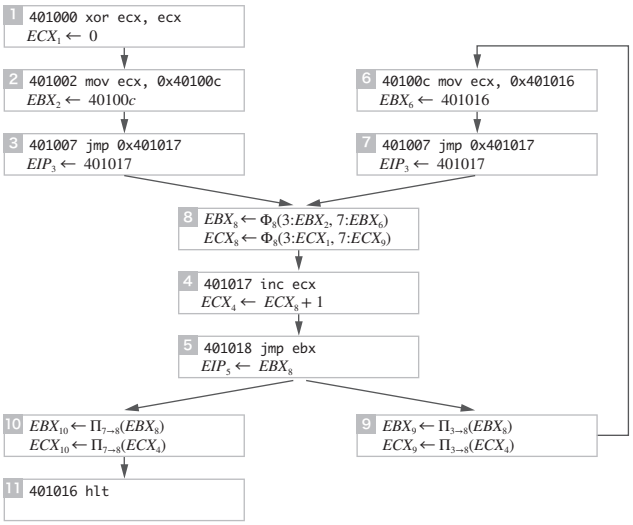

図-8 PSA形式:Π関数の挿入

はノード8で合流した情報からノード3から来た情報を取り出す、という意味をもち、$\Pi_{3\to8}(\Phi_8(3{:}X,7{:}Y))\Rightarrow X$のように評価されます。このΠ射影関数によるSSAの拡張は「射影的単一代入(Projective Single Assignment、PSA)」と呼ぶ本研究独自の表現形式です。

Π関数を利用すると、例えばプログラム終了時(ノード11)のECXレジスタの値は以下のように計算できます。

$$ECX_{10}\Rightarrow \Pi_{7\to8}(ECX_4)\Rightarrow \Pi_{7\to8}(ECX_8)+1\Rightarrow \Pi_{7\to8}(\Phi_8(3{:}ECX_1,\,7{:}ECX_9))+1$$
$$\Rightarrow ECX_9+1\Rightarrow \Pi_{3\to8}(ECX_4)+1\Rightarrow \Pi_{3\to8}(ECX_8)+2\Rightarrow \Pi_{3\to8}(\Phi_8(3{:}ECX_1,\,7{:}ECX_9))+2$$
$$\Rightarrow ECX_1+2\Rightarrow 2$$

このように本研究手法では呼出規則に従っていないようなプログラムでも、再構成の過程で文脈依存性を検出し、解決することが可能となっています。

3.4 応用:バッファオーバーフローの安全性証明

射影的単一代入では射影関数Πだけでなく、各条件分岐における分岐条件を値に付与する条件関数Γも追加されます。これを応用すると、関数内でループによってスタック上のデータを書き換えるような処理を行っている場合、ループの中でスタック上に記録されている帰りアドレスが上書きされる可能性があるかどうかを調べることができます。

例えば、図-9のプログラムをPSA形式に変換して簡略化すると図-10のようになります。

ここでLd(M,A,N)はメモリ状態Mのアドレスからのnバイトの値の読み出しを、St(M,A,X,N)はメモリ状態Mのアドレスへのnバイトの値Xの書き込みを表現しています。ここでは関数の終了時にスタックメモリ上のEBP_1+4と表現された位置に格納された4バイトの復帰アドレス値へジャンプしています。この領域がforループの中で上書きされる条件は、PSA形式を用いて以下のように表現されます。

$$\Gamma(i_2<10,\ EBP_1+4\le EBP_1-10+i_2\le EBP_1+8)$$

これは$i_2<10$の条件のもとで$EBP_1+4\le EBP_1-10+i_2\le EBP_1+8$が成立するか、ということを表します。$i_2$はΦ($i_1$,$i_4$)と定義されており、$i_4$がループ内で定義されているため、$i_2$はループ内で変動する値であることが表現されています。この条件を満たすi_2の値があれば、ループ中に復帰アドレスの値が上書きされる可能性があります。しかし、この場合は容易にそのようなi_2の値が存在しないことが分かります。つまり、このループによって復帰アドレスの領域が変更されることはないことが保証されます。

実際のプログラムではループ条件もメモリ書換条件もより複雑な形で現れます。ループ内で変動するこれらの条件が充足可能か否かを決定するためには、ループ不変条件(Loop invariant)と呼ばれるループの中では変わらない条件式を見つけることが重要になります。近年こうしたループ不変条件を自動的に算出する解析手法の研究が進み、Z3などSMTソルバ内に実装されています。本研究ではバイナリプログラムからループ条件や

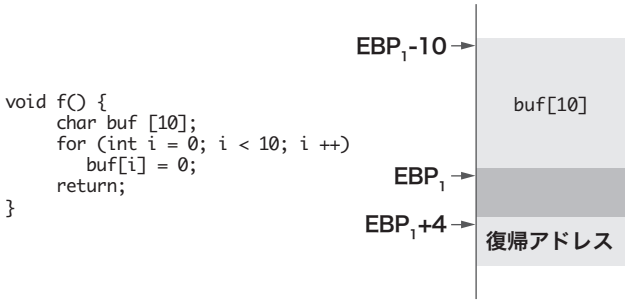

図-9 プログラム例

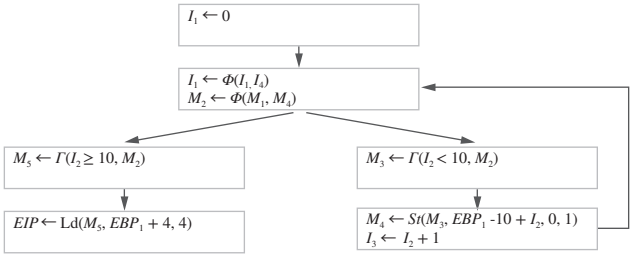

図-10 ループ

メモリ書換条件を抽出し、SMTソルバを用いてループ不変条件を自動算出することを目標としています。このような解析は必ずしも一定時間内に決定できるとは限りませんが、もし上書きされる可能性がないと判定できた場合は、バッファオーバーフローが起こらないということを保証できることになります。また、上書きされる可能性があると判定した場合はどのような条件下で上書きが起こり得るのかを調べることができます。

現在本手法を用いてAIエッジデバイスのような組込ファームウェアに対し、バッファオーバーフローのなどの脆弱性やトロイの木馬のようなバックドアの有無を検出する技術への応用を研究しています。

3.5 おわりに

本稿では技術研究所で開発を行っている、前提知識を必要としないバイナリプログラム解析技術について解説しました。既存の静的解析ではプログラムを事前に関数単位で分割する必要がありますが、そのためにはプログラムがどのように生成されたかについて前提知識を必要としました。本研究手法では、独自拡張されたSSA形式を用いて間接ジャンプの行き先を評価すると同時に文脈依存性を検出することで関数領域の推定を行うことができるため、前提知識が得られない「素性の悪い」プログラムであってもプログラム解析を行うことが可能となっています。

しかし、静的バイナリプログラム解析には決定不能問題が含まれているため、完全解が得られる解析手法は存在しません。本手法においても、評価式が肥大化し静的な解決が困難になりそうな場合にはそれ以上の評価を行わないことで、近似解を出しています。

バイナリプログラム解析に用いられる手法としては他に記号実行(Symbolic execution)が挙げられます。記号実行は静的解析と動的解析の中間に位置する解析手法で、2016年にDARPAが主宰した情報セキュリティの自動化コンテストCyber Grand Challengeでは、mayhem[*5]やangr[*6]と行った記号実行を用いた解析ツールが上位に入りました。記号実行ではプログラムがバッファオーバーフローなど危険な状態になりうるかどうかを検出することができます。しかし、プログラムが安全である、すなわち危険な状態にはなり得ないことを証明するには網羅的な実行が必要になるため記号実行には向いていません。その点で記号実行と本研究手法は補完的な役割を果たすと考えられます。

今後は本研究と記号実行や動的解析など他の解析手法を統合したバイナリプログラム解析ツールの開発を目指していきます。

《 謝辞 》
この研究は国立研究法人新エネルギー・産業技術総合開発機構(NEDO)「AIエッジデバイスの横断的なセキュリティ評価に必要な基盤技術の研究開発」の委託業務の一部として行っています。

執筆者:
泉田 大宗(いずみだ とものり)
IIJ-II技術研究所 研究所員(2015年より)博士(情報科学)。

*5 Cha2012:Cha, S. K., Avgerinos, T., Rebert, A., & Brumley, D. (2012, May). Unleashing mayhem on binary code. In 2012 IEEE Symposium on Security and Privacy (pp. 380-394). IEEE.
*6 Wang2017:Wang, F., & Shoshitaishvili, Y. (2017, September). Angr-the next generation of binary analysis. In 2017 IEEE Cybersecurity Development (SecDev) (pp. 8-9). IEEE.

March 2020 Vol.46

IIJ
Internet Initiative Japan

株式会社インターネットイニシアティブ(IIJ)について

IIJは、1992年、インターネットの研究開発活動に関わっていた技術者が中心となり、日本でインターネットを本格的に普及させようという構想を持って設立されました。
現在は、国内最大級のインターネットバックボーンを運用し、インターネットの基盤を担うと共に、官公庁や金融機関をはじめとしたハイエンドのビジネスユーザに、インターネット接続やシステムインテグレーション、アウトソーシングサービスなど、高品質なシステム環境をトータルに提供しています。
また、サービス開発やインターネットバックボーンの運用を通して蓄積した知見を積極的に発信し、社会基盤としてのインターネットの発展に尽力しています。

本冊子の情報は2020年3月時点のものです。

IIJ-MKTG019-0046

株式会社インターネットイニシアティブ
〒102−0071 東京都千代田区富士見2-10-2　飯田橋グラン・ブルーム
E-mail：info@iij.ad.jp　URL：https://www.iij.ad.jp