博士論文

IT システムおよびネットワークにおける
情報セキュリティ確保技術に関する一研究

大久保一彦

2018 年 12 月

IT システムおよびネットワークにおける情報セキュリティ確保技術に関する一研究

大久保一彦

内容梗概

この数十年間にわたる IT の技術革新ならびに世の中への浸透に伴い，機密性・完全性・可用性といった情報セキュリティの 3 大要素を確保するための観点や技術も，逐次変貌を遂げている．その中で，情報セキュリティに関する優先的な課題を見極め，セキュリティ・バイ・デザインの考え方から，如何に方策を打つかが肝要になる．本論文では，IT システムおよびネットワークが置かれた時代・環境やその際のユーザ・市場ニーズを踏まえ，情報セキュリティの確保技術について考察する．第 1 章では，初のコンピューター・ウィルスの出現から現在までにおける環境とセキュリティ上の課題の変遷と，昨今の具現化する脅威と必要なセキュリティ対策技術を網羅的に述べる．第 2 章では，インターネットがブレイクする以前に期待が高まっていた，双方向マルチメディアサービスについて，特に当時のセキュリティにおける可用性重視の観点から，そのシステムおよびネットワークの設計手法を述べる．第 3 章から第 5 章では，永遠のイタチごっこの如く，技術の高度化が持続的に求められる IT セキュリティの領域にフォーカスし，開発が嘱望される 3 つのサイバー攻撃対策技術（情報セキュリティ確保のための技術），すなわち「通信ログ分析に基づくマルウェア感染検知」,「悪性 Web サイトにおける Evasive コード特定」,「64 ビット版 Windows のメモリダンプからのスタックトレース構築技術」について述べる．第 6 章では，本論文のまとめ，ならびに今後のセキュリティ研究について展望を示す．

A Study on Technologies related to Information Security Assurance for IT Systems and Networks

Kazuhiko Ohkubo

Abstract

With innovation of the IT(Information Technology) and its penetration to the world for these dozens of years, viewpoints and technologies for assuring 3 primary elements of information security, so called, confidentiality, integrity and availability have been appropriately transformed. Under such situations, it is very important how to take measures from a security-by-design point of view after having ascertained preferential problems on information security. In this thesis, the author studies technologies related to information security assurance taking into consideration not only the times and the environment where IT systems and networks were introduced but also security related needs or concerns from the users and market. The chapter 1 describes, first of all, changes of environment and security issues from the appearance of the 1st virus to the present, then comprehensively describes menaces to embody and the security technologies to be developed as necessary. The chapter 2 proposes, making much of availability, a design methodology of multi-media service system and network into which expectation was sublimed in those days before a break of the Internet. In the chapter 3 through 5, respectively, three promising cyber-attack countermeasure technologies (for information security assurance), that is, “detection of malware infection using communication log analysis,” “specification

of evasive code on malicious web sites" and "advanced stack tracing in memory forensic" are proposed focusing on a field of IT security where a mouse-and-cat game will be ever-lasting. The chapter 6 describes a summary of the thesis and prospects of the future study on security technologies.

目次

図目次

表目次

第1章

緒言

本章では，初のコンピューター・ウィルスが出現したと言われる1985年から現在までにおける，サイバーを取り巻く環境変化とセキュリティ上の課題の変遷を踏まえつつ，昨今の具現化する脅威と必要なセキュリティ技術について，領域網羅的に概説を加えたうえで，本論文の構成について述べる.

1.1. サイバーを取り巻く環境変化と課題の変遷

1985 年にソフトウェアの不正コピーに抗議する目的でパキスタンのプログラマーが作成したコンピューター・ウィルス「Brain」が，初のコンピューター・ウィルスと言われている[1-1]. サイバー攻撃者は，図1-1に示されるように[1-2]，古くは不正プログラムなどにより，保存されたファイルを削除したりプログラムの実行を妨げたりしていた. その目的は，愉快犯的なものから始まるものの，インターネットがつながり始めてからは（1995年が日本の「インターネット元年」とされているが），DDoS（Distributed Denial of Services：分散型サービス拒否）攻撃などによってサービス停止を引き起こし，インターネット通信を妨害したりすることで，ビジネスの阻害や復旧の代償に金銭を要求するといった悪意ある行為が行われてきた. 特に国内では，2011年当たりから，重要情報の入手を最終目標とし，時間，手段，手法を問わず，目的達成に向けて特定の組織を標的として継続して行われる「標的型サイバー攻撃」も顕在化してきた. 現在では，標的型サイバー攻撃に限らず，企業や組織，個人の持つ重要な情報をさまざまな手口で盗み出し，不正売買や漏えい情報の開示をネタに恐喝を行うといった深刻なサイバー犯罪被害の報告も後を絶たない. さらに，思想信条を背景としたテロリストが行うサイバーテロ行為だけでなく，軍隊・情報機関による高度なクラッキング攻撃等を伴う，ナショナリストによるサイバー戦争にもエスカレートしている.

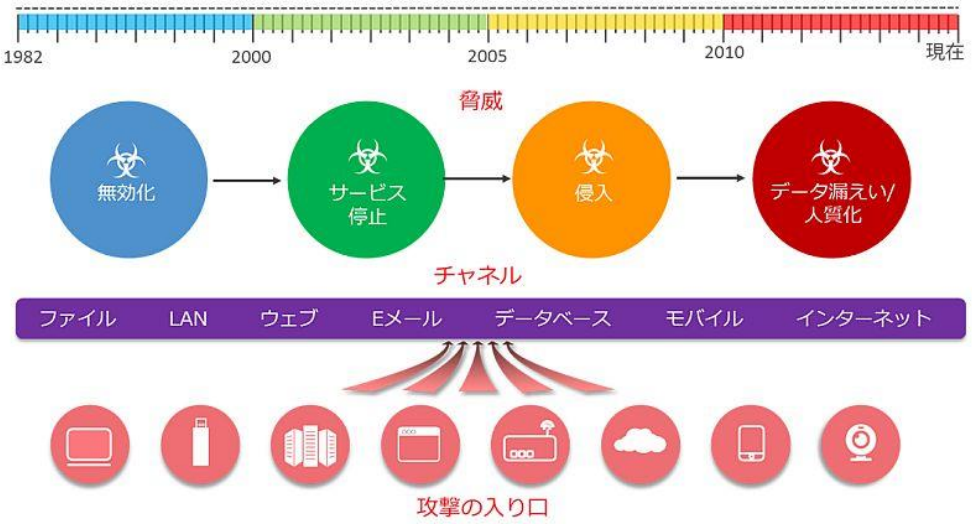


図 1-1. サイバー脅威の変遷

情報セキュリティとは[1-3]，組織等の情報システムを取り巻くさまざまな脅威から，情報資産を，機密性・完全性・可用性といった 3 要素の確保を行いつつ，正常に維持することである．機密性の確保は，情報資産において正当な権利を持った人だけが使用できる状態にしておくことで，対策としては，情報漏えい防止，アクセス権の設定，暗号の利用などが挙げられる．また，完全性の確保は，情報資産が正当な権利を持たない人により変更されていないことを確実にしておくことで，対策としては，改ざん防止，検出などが挙げられる．一方，可用性の確保は，情報資産を必要なときに使用できることで，対策としては，充分なシステム設計，システムの二重化，バックアップ，災害復旧計画などが挙げられる．

これら情報セキュリティの 3 要素を，ここ数十年間のサイバー脅威の変遷に照らし合わせて考えると，少々乱暴と考えられるものの概ね，1990 年代は比較的可用性を重視しており，2000 年代に入ると機密性・完全性の確保が課題に，そして 2010 年代は，重要インフラや制御システムへの攻撃を鑑みると，再び可用性の配慮も取り沙汰されてきていると言える．

こういった中，情報セキュリティの 3 要素確保のいずれにせよ，セキュリティ・バイ・デザイン[1-4]の考え方を，きちんと弁えておかなければならない．セキュリティ・バイ・デザインは，情報セキュリティを企画・設計段階から確保するための方策であり，そのメリットは，システム等の開発の早い段階から入れ込めることから，①手戻りが少なく納期を守れる，②運用を含めたトータルコストを少なくできる，③保守性の良いソフトウェアができる，点にある．しかしながら，（１）セーフティ設計に比べ，セキュリティ設計の歴史が浅く，上流工程の開発プロセスが定まっていない，（２）セキュリティが非機能要件なので，コンセプトを決める企画・設計段階では考慮がされづらい，といったことが，セキュリティ・バイ・デザインの適用が一般に難しい理由として挙げられている．

1.2. 具現化する脅威と今後必要になるセキュリティ技術

本項では，「IT」「IoT/OT」「重要インフラ」のそれぞれの領域において，具現化する脅威やセキュリティの問題を概観したうえで，それらを解消すべく開発が急がれるセキュリティ技術について述べる．

1.2.1. Cat-and-Mouse の IT セキュリティ

2018 年 2 月に開催された平昌冬期五輪では，1～2 ヶ月前辺りからのオリンピック関連組織への標的型攻撃や，開会式前後のインシデント被害がいくつか発生している．具体的には，公式 web サイトの一時停止，プレスセンターのネット接続等の一時停止，スタジアムの無線 LAN の一時停止，開会式で使われる予定だったドローンが飛ばなかった等である．サイバー攻撃に使われたマルウェア（Olympic Destroyer）はインターネットに公開されていたため，独自に入手・分析をした結果，図 1-2 示すように，当該マルウェアの攻撃の手口は複雑であり，その末路は PC 等の破壊活動を誘発するものであったことから，サ

イバー攻撃の目的・意図は明らかに「大会そのものを混乱させること」と考えられる.

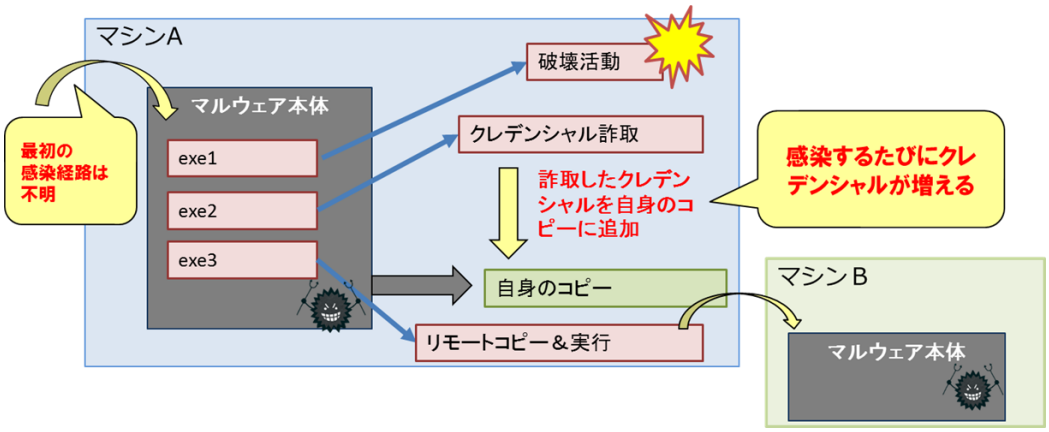


図 1-2 Olympic Destroyer について

このように，昨今のマルウェアの手口やそれによるサイバー攻撃は巧妙化するとともに，機器に感染するマルウェア（ボット）を通じて仕掛けられるボットネットによる高トラヒックの DDoS 攻撃も多数発生しており，サイバー攻撃は大規模化しつつもある．このように巧妙化・大規模化するサイバー攻撃に対抗するためには，従来の監視対象である法人及びホームネットワークや ISP ネットワークの領域において引き続き，攻撃に追随すべく，悪性 Web サイト検知やマルウェア感染検知，ボットプロファイリング，ドメインレピュテーションといった対策技術の高度化を恒常的に図る必要がある.

これに加えて更に，よりミクロやマクロの観点から，エンドポイントならびにバックボーンネットワークといった領域へ監視対象を拡大する必要がある．具体的には，エンドポイントにおいて，フォレンジックやテイント解析等の技術を駆使したマルウェアの動作解析により，高精度な IOC（Indicator Of Compromise）を生成し，MDR（Managed Detection and Response）製品などで活用することが有効である．また，バックボーンネットワークにおいては，大量なフロー情報を分析することでボットネットの全体構造（Herder，C2C サー

バ，ボット端末）が浮き彫りになり，より高性能に DDoS 攻撃が検知されて，適材適所の対策を講じることが可能になる．

また従来は，システムやネットワークのセキュリティに着目したセキュリティ技術が主に開発されてきた一方で，「実際に騙されてしまうのはユーザである」という視点から，ヒューマン・コンピュータ・インタラクションに着目した新たな技術分野「ユーザブルプライバシー＆セキュリティ」が注目されつつある．具体的には，ユーザの認識違いや不適切な動作の原因（ギャップ）を解明し，システムやアプリケーション，GUI の改善を図ることで，セキュリティ向上に繋げることができる．また，プライバシー＆セキュリティにおけるギャップを踏まえ，人の動作を模擬する高度なハニーポットやセキュリティ熟練者のためのインテリジェンス生成という形での技術の高度化にも大いに期待できる．

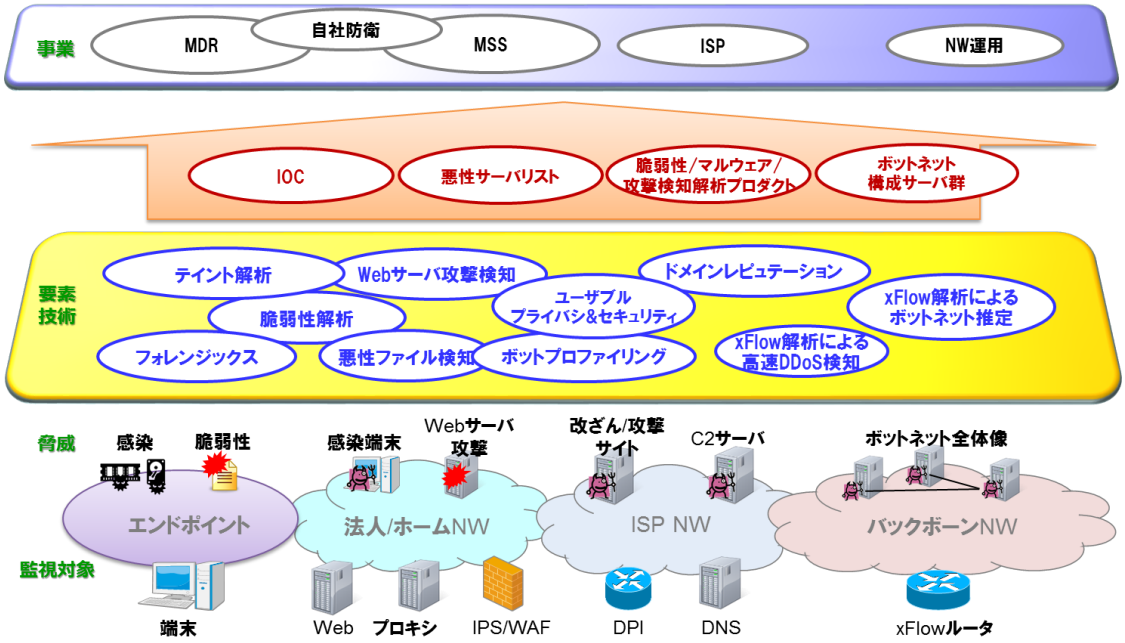


図 1-3. 巧妙化・大規模化するサイバー攻撃への対抗

巧妙化・大規模化するサイバー攻撃へ対抗するための一連の要素技術について，図 1-3 にまとめる.

1.2.2. 非 IT 分野 ”IoT/OT”のセキュリティ

IoT 機器に感染するマルウェアとして代表的な Mirai が最初に使われた大がかりなサイバー攻撃は，2016 年 9 月のセキュリティブログ「KrebsOnSecurity」への DDoS であり，その規模は当時最大級の 620Gbps だったと言われている．その後直ぐ，Mirai のソースコードが Anna-senpai という者によって世に放たれ，結果として亜種が沢山作られるようになり，事或るごとに大規模な DDoS 攻撃が発生し続けている．その際，IoT 機器の持ち主は自身の機器が大規模 DDoS 攻撃に使われていることを殆ど認識していないのが実情である．一方で，IoT 機器の持ち主が困り果ててしまう，ランサムウェアいわゆる，身代金ウィルスとも呼ばれるマルウェアもある．IoT 機器がランサムウェアに感染し，ビットコイン等を身代金的代償として支払わなければ，機器が正常に動作しないような状況が作れることは既に実験室レベルで実証されている．従って，今後，スマートホーム等の進展に伴い，このような状況に陥ってしまうことも時間の問題と考えられる.

2018 年 3 月に新聞記事でも話題になったが，NTT 東日本及び西日本が提供する法人向けルータにおいて，インターネットから管理画面が見えてしまう数百台にも及ぶ当該ルータが発見された．これは何らかの原因による事象と考えられるものの，その根本的な問題は，当該ルータのみにあるのではなく，IoT 機器全般にあり得ることであり，（いずれも WebUI が外部にオープンになっていることが前提であるが）以下の通りである.

・デフォルト ID/PW が極端に簡単

・デフォルト ID/PW を変更しなくても運用可

・デフォルト ID/PW が製品もしくはベンダ共通で設定されている

・デフォルト ID/PW を記述したオンラインマニュアルを誰でも閲覧可

この問題への対応として日本では，総務省が電気通信事業者法等の法改正を 2018 年度中に行い，施策を開始すべく動いている．そのひとつに，IoT 機器などを悪用したサイバー攻撃の深刻化を踏まえ，国立研究開発法人・情報通信研究機構（NICT）の業務に，パスワード設定に不備のある IoT 機器の調査等を追加（5 年間の時限措置）する等を内容（図 1-4）とする NICT 法の改正[1-5]がある．

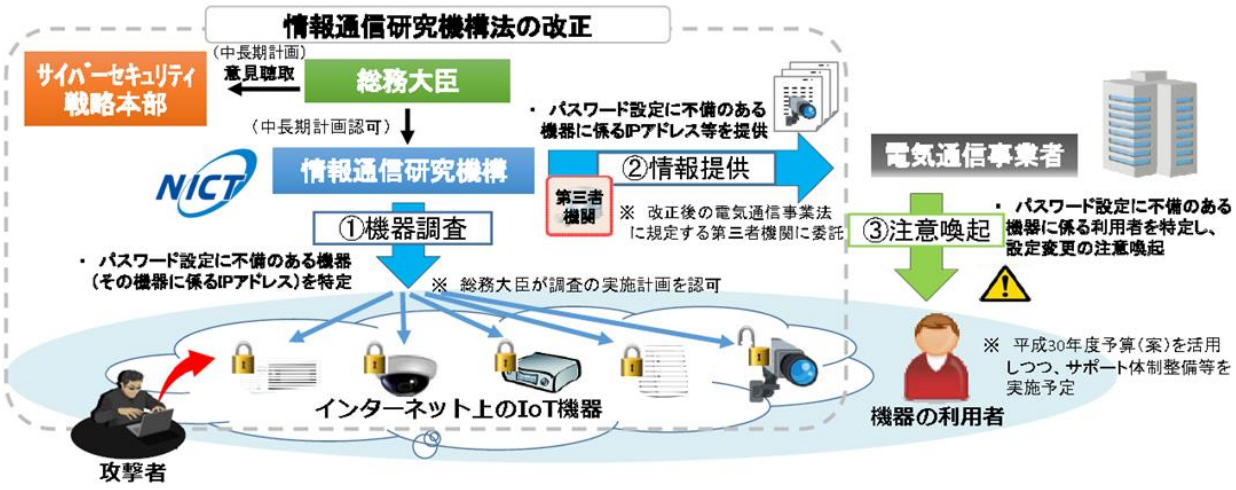


図 1-4. NICT 法の改正

技術開発の視点では，例えば，アンチウィルスソフトのような IT セキュリティの機能をそのまま，計算機リソース（CPU，メモリ・ディスク，電源等）がプアな IoT 機器に適用できないことから，IoT 機器向けの，「認証・認可」「構成管理」「検知」「対処（オーケストレーション）」といった一連のセキュリティ技術をフロム・スクラッチで確立することが必要になる．

「認証・認可」については，例えば，サーバ側でパスワードを管理しなくてよい次世代認証技術が挙げられる（図 1-5）．これは，クライアントの初期登録

時にデバイス側に秘密情報を払い出し，それと（キャッシュカードの暗証番号のような簡単なものでも十分であるが）デバイス固有の ID を使って暗号演算を施すことにより認証を実現する方式である．これによれば，IoT 機器のパスワード運用をいちいちしなくてもよく，また認証に必要な（大多数に及ぶ）証明書の発行・運用等のコストもかからないで済む．

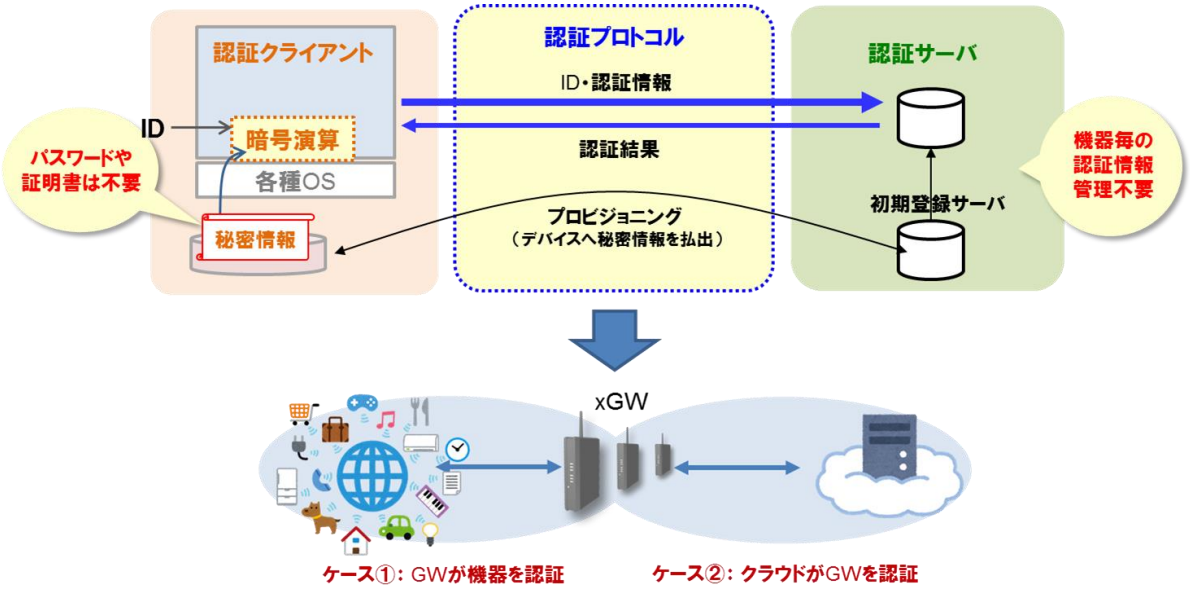


図 1-5. IoT/OT 向け次世代認証技術

「構成管理」「検知」「対処（オーケストレーション）」については，例えば，GW（Gateway）配下に多種多様な IoT 機器が繋がる状況下で，一般に利用されている ARP フレームの出力特性解析やノイズの除去により，運用条件の厳しい LAN 環境においても，精度良く機器を特定・推定して構成把握ができるうえ，IoT 機器の特徴情報から，脆弱性を有する機器を発見することができる．また，グラフ理論等を活用し，平常時の通信相手（ホワイトリスト）から逸脱したトラヒックをアノマリーな状態として検知することで適宜，サイバー攻撃等による異常通信に対する通信制御（アラート及び遮断等）が可能になる．

OT についてもセキュリティ技術を一から確立することが必要になることは同様であり，特に OT の領域では，産業用プロトコルなど特殊なものへの対応も更に必要になってくる．このような OT 向けセキュリティ対策技術の一例として，NTT と三菱重工の企業連携によって開発された InteRSePT[1-6]と呼ばれるサイバー攻撃監視防御システムを図 1-6 にて紹介する．

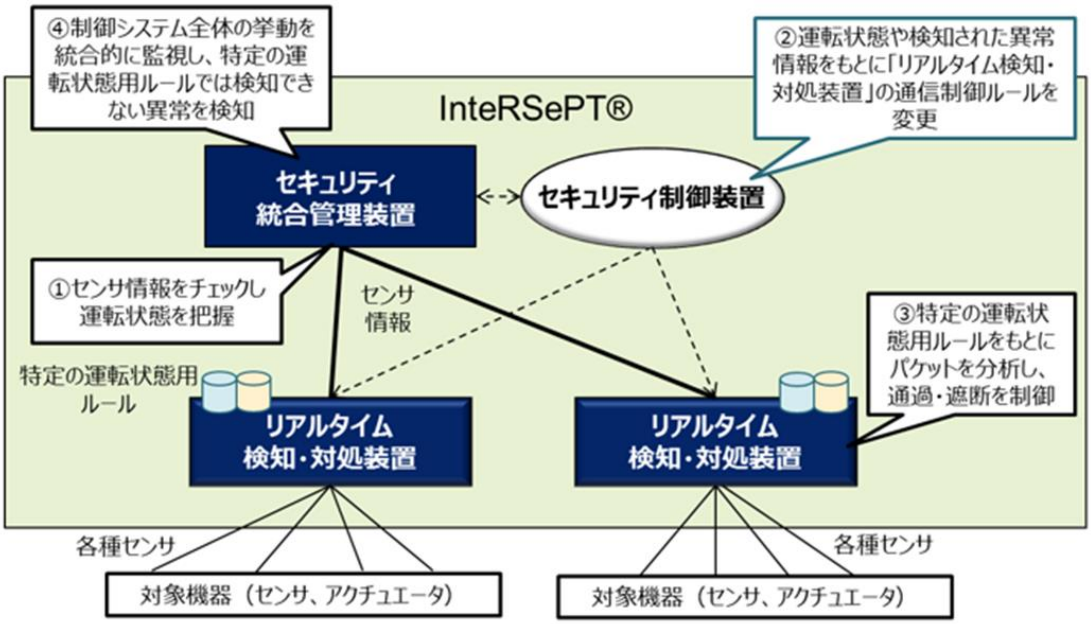


図 1-6. 産業用プロトコル向け攻撃監視防御技術の例

InteRSePT は，「リアルタイム検知・対処装置」と「セキュリティ統合管理装置」で構成され，ネットワークに流れるセンサ情報等のデータを統合的に監視し，従来の技術では対応が困難だった制御指令を悪用したサイバー攻撃を検知できる．また，対象機器の運転状態毎に，リアルタイムに適用するセキュリティルールを変更することで異常を早期に発見し，可用性を維持しながら未知のサイバー攻撃にも迅速に対応できる．

1.2.3. セキュリティ面での重要インフラ防護

2009 年から 2010 年にかけて，Stuxnet と呼ばれるマルウェアがイランの核施設に蔓延して実被害をもたらした事案が，重要インフラへの最初のサイバー攻撃として知られている．インターネットに接続していないシステムが USB メモリーを介して感染・発症した点で，当時，大きな衝撃でもあった．

重要インフラにおいては，その「大規模性，複合連動システム化」といった特徴と，「汎用化，オープン化，新技術の適用」といった環境変化が，増大するサイバーリスクを考える点で重要である．

前者の特徴については，数千台のサーバ機器，数万から数十万台の制御機器といったインフラ設備が珍しくなく，1 箇所でもサイバー攻撃が成功すれば影響は広範囲に及ぶおそれがある．このため，構成要素がそもそも大丈夫なのかといった観点から，不正な機器の混入や改変を常時確認し，異常動作を阻止する，真贋判定技術が必要になる．

後者の特徴については，インターネット技術，Linux などのオープンソースソフトウェア（OSS）の採用が進むことで，脆弱性等の情報が得られやすくなっている点から，サイバー攻撃の成立は大前提であり，前述した真贋判定技術がビルトインできない古い機器および，IoT 等の機器やネットワークにおいても，システムの異常を監視可能なボルトオン型の動作監視・解析技術が必要になる．

真贋判定技術（図 1-7）ならびに，動作監視・解析技術（図 1-8）については，当該技術の一部を，総合科学技術・イノベーション会議の戦略的イノベーション創造プログラム（SIP）「重要インフラ等におけるサイバーセキュリティの確保」（管理法人：NEDO）にて，2015 年度から 2019 年度にわたり開発中である[1-7][1-8]．

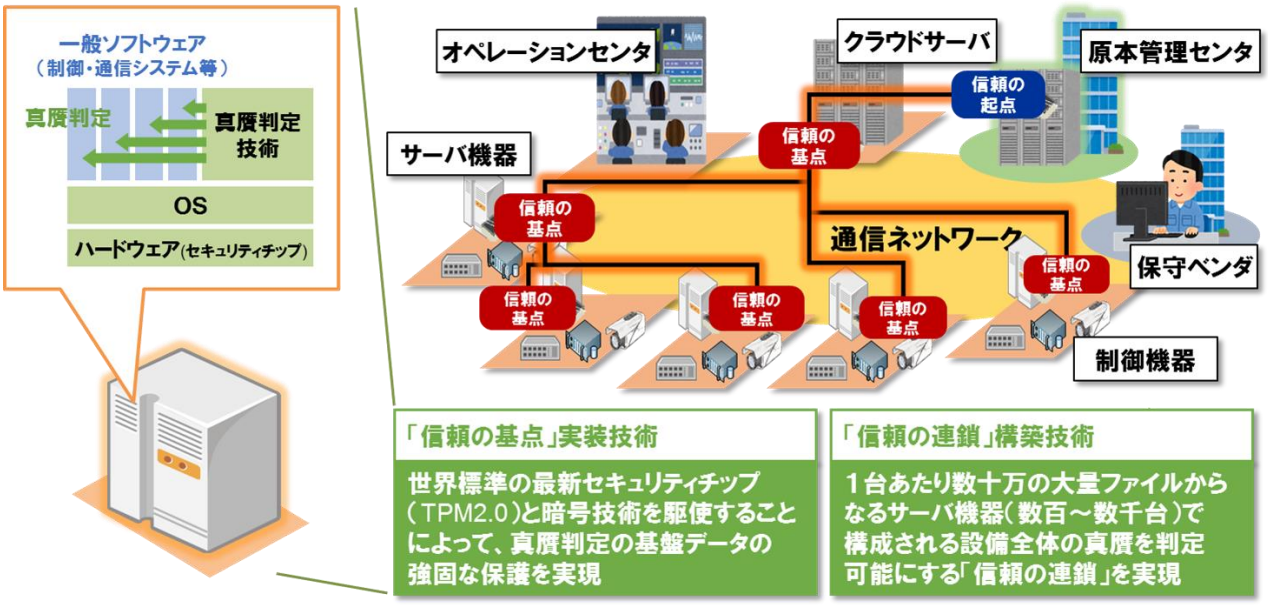


図 1-7.　真贋判定技術

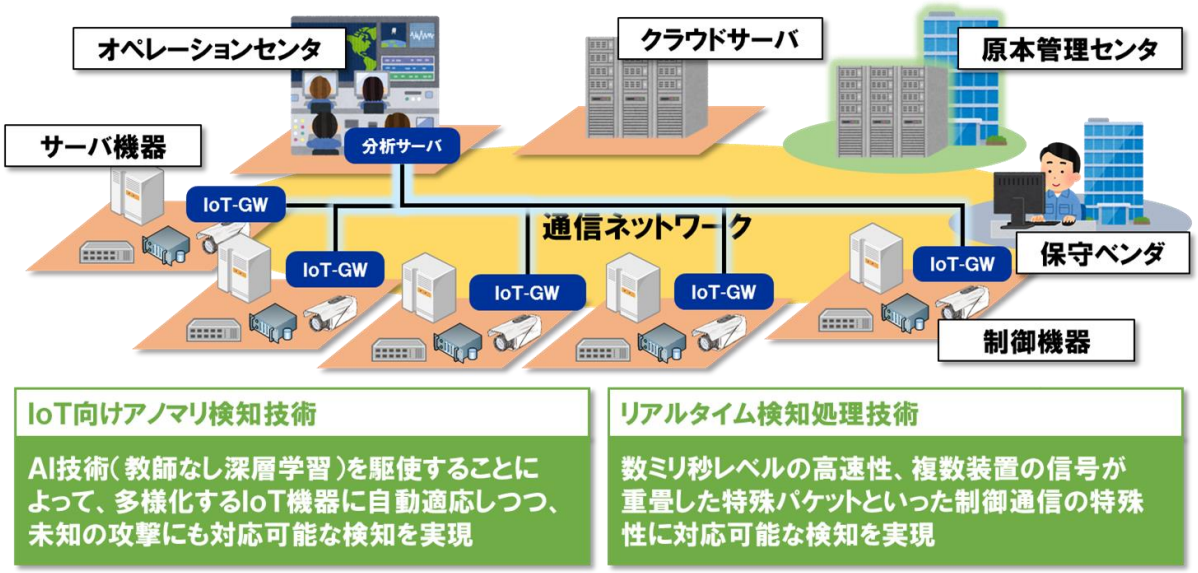


図 1-8.　動作監視・解析技術

また，2020 年の実用化・商用化に向けて開発が進められている新技術”5G”についてもその普及に備えたリスク洗い出しやセキュリティ対策検討が，今後ますます重要になる．具体的には，広帯域，超低遅延，同時接続数の増大といった特徴を備える 5G ネットワーク環境を踏み台とした攻撃能力のアップや，IoT 機器や利用形態の多様化，インフラ依存性の増大に伴う新たなサイバー攻撃，さらには電気通信設備を含むネットワーク機器への攻撃やなりすましが，リスク増になると考えられる．このため，音声とデータの保護に注力した従来ネットワークのセキュリティアーキテクチャに取って代わるべく，5G 環境下における新たなセキュリティアーキテクチャの検討が急がれており，具体的には以下のような観点の考慮が肝要となる．

① NW とサービス，ユーザのハイブリッド認証

② 仮想 NW スライスのセキュリティ管理

③ 大規模 DDoS 攻撃などへの NW 側での対策

④ 無線区間を含むトラヒック監視・アノマリー検知

⑤ プライバシー情報（ID，位置情報，個人的コンテンツ）保護　等

1.3. 本研究の目的と本論文の構成

「IT」「IoT/OT」「重要インフラ」の各領域においてセキュリティの 3 要素を確保するには，暗号技術はもちろんのこと，周辺領域のさまざまな技術をも利用する必要がある．たとえば，双方向マルチメディアサービスはインターネットの普及前から開発され，現代においても盛んに利用されているが，そうしたサービスを高い可用性を持つように構築するためには，接続品質や帯域保証を考慮したネットワーク設計技術が重要となる．さらに，現代のインターネットの時代においては，情報の機密性や完全性を保持する観点から，機械学習，プ

ログラム解析，データ分析のための技術を積極的に採り入れることが重要となってきている．そこで本研究では，「ネットワーク設計技術」「機械学習」「プログラム解析（制御フロー解析など)」「データ分析（特にメモリフォレンジック)」といった周辺領域にある技術に注目し，それらをどのように要素技術として組み合わせてシステム全体のセキュリティを確保するかを実証的に示したい．

以降では先ず，インターネットがブレイクする以前に期待が高まっていた，双方向マルチメディアサービスについて，特に当時のセキュリティにおける可用性重視の観点から，その「システムおよびネットワークの設計手法」について，第 2 章にて提案・評価を行う．次に，永遠のイタチごっこの如く，技術の更なる高度化が持続的に求められている（1.2.1 節にて言及した）IT セキュリティの領域にフォーカスし，その中でも特にこれから開発が嘱望される 3 つのサイバー攻撃対策技術（情報セキュリティ確保のための技術）である，「通信ログ分析に基づくマルウェア感染検知」，「悪性 Web サイトにおける Evasive コード特定」，そして，エンドポイントセキュリティにおける「メモリフォレンジックの高度なスタックトレース」のそれぞれについて，第 3 章から第 5 章にて提案・評価を行う．最後に，第 6 章では，本論文のまとめ，ならびに今後のセキュリティ研究について展望を示す．

第 2 章

可用性を配慮したマルチメディアシステムの設計技術

近代の通信ネットワークの高度化と多様化により，マルチメディアサービスは飛躍的な発展を遂げてきている[2-1]．具体的には，放送，情報家電，教育などの分野でマルチメディア技術を応用した新たなサービスが，これまで数多く生まれてきた．その中で，双方向マルチメディアサービスについては，NTT が，マルチメディア時代に不可欠なネットワーク，ユーザ設備，ソフトウェアという三つの要素の調和した発展と，そこから生まれる新たな利用方法や利用技術（アプリケーション）の創造・開発，そして，これらに対応するネットワークの構築技術・管理技術の確立を目的に，企業，大学，公的研究機関等 127 組織と共同で，1994 年 9 月より 2 年半にわたりマルチメディア通信の共同利用実験として取り組んだものである[2-2]．

本章では，インターネットがブレイクする以前の，千葉県浦安市において約 350 世帯の一般家庭を対象として提供された双方向マルチメディアサービスを題材に，特に当時のセキュリティにおける可用性重視の観点から，システム設計法について提案を行う．

2.1. 研究の背景・経緯

1993 年 9 月に米国政府が発表した NTT（National Information Infrastructure）構想を皮切りに，世界各国で将来の双方向マルチメディアサービスに向けた実験プロジェクトが始まった．これらの実験で観られるように，VOD（Video On Demand）やネットワークゲーム等の双方向マルチメディアサービスを，一般家庭を対象に実現するためには，システムの構成要素として，広帯域のアクセスネットワーク，およびセンタ側にアプリケーションやコンテンツを提供するサーバ，家庭側にサーバと通信処理を行い，TV に映像を表示するための専用端末（セットトップ）が当時必要であった．

このような当時の双方向マルチメディアシステムに対する主な要求条件は以下の通りである.

[1] 双方向通信が可能な高速，広帯域のネットワークを提供する．各種アプリケーションで要求されるさまざまな帯域を提供し，トラヒック特性の影響を受けにくいネットワークを実現するためには，ATM ネットワークが望ましい[2-3].

[2] セットトップは，ユーザによるメンテナンスの容易さやソフトウェアのバージョンアップ等に対する追従性を重視し，ディスクレスの NC（Network Computer）であることが望ましい．この場合，システムカーネルやアプリケーションプログラムは必要時にサーバ側からダウンロードされる.

[3] 一般家庭でリモートコントローラにより TV 感覚で気軽に操作でき，ユーザフレンドリーかつ，ハイエンドな双方向サービスを提供する．具体的なサービスメニューは，VOD 系のアプリケーションとゲーム系のアプリケーションの 2 種類に分けられる.

[4] VOD 系のアプリケーションでは，映像ストリームの転送帯域が保証され，NTSC（National Television System Committee）方式以上の映像品質を満足する．リモートコントローラのボタン操作による応答時間は，2 秒程度とする.

[5] ゲーム系のアプリケーションとして，セットトップ側のみでゲームが完結するスタンドアロン型のものに加え，出会いの場を利用して相手を見つけ，ネットワークを介して対戦ゲームを楽しむネットワーク型のものを提供する．特に，後者においては，不快を感じさせない短い時間でゲームコントローラのボタン操作から，一連の処理及び TV への描画までを終了する.

このような双方向マルチメディアシステムにおいて送受信されるデータは，アプリケーション毎にさまざまであるが，接続品質，帯域保証の観点から，以下の 3 つのタイプに大別される[2-4].

【タイプ 1】接続品質，帯域保証に対する要求が比較的厳しくないアプリケーションプログラム.

【タイプ 2】帯域保証に対する要求は厳しくないが，接続品質が重要視される各種コマンドおよび音声．このタイプについては，予め必要コネクション分の帯域を確保しておく必要がある.

【タイプ 3】接続品質に対する要求は厳しくないが，帯域保証が重要視される映像ストリーム．このタイプについては，一旦コネクションが確立したら，帯域を確保する必要がある.

ネットワークを ATM で実現する場合，サーバとセットトップ間のチャネル設定について，上記タイプ別のデータ毎にそれぞれチャネルを設定する方法と，チャネルは 1 つであるが，分類されたデータ毎にセルに優先度を設定する方法の 2 種類が考えられる．前者は後者に比べて，統計多重効果が得られない分，帯域の使用効率が低くなるデメリットはあるが，トラヒック特性が異なるデータタイプ別に分析が可能になり，ネットワーク設計が容易なこと，サーバ及びセットトップにおける優先度に関わる処理負荷を低減できるメリットがある．以下では，前者のような設定を，データタイプ別チャネル設定法と呼ぶ.

本章では，ATM ネットワークに対し，データタイプ別チャネル設定法を適用する手法において，上述した双方向マルチメディアサービスに対する要求条件を満足し，ネットワークリソースの利用率向上を図り，かつユーザに不快感を与えないサービス品質を実現するシステムインテグレーション並びに設計法について提案する.

2.2. システム設計と課題

一般的な双方向マルチメディアシステムの構成を図 2-1 に示す．システムの構成要素は，セットトップ，光加入者端局装置（SLT: Subscriber Line

Terminal), ATM 交換機, サーバ及びサーバ間通信ネットワーク（LAN）である[2-5]. また, 前項で述べたように, データタイプ別チャネル設定法を適用した場合のセットトップとサーバ間チャネル構成を図 2-2 に示す. ここでは, 負荷分散等の目的で, タイプ１, タイプ２のチャネルに対しては, 各サーバが均等にセットトップ向かいのチャネルを収容する構成を前提としている.

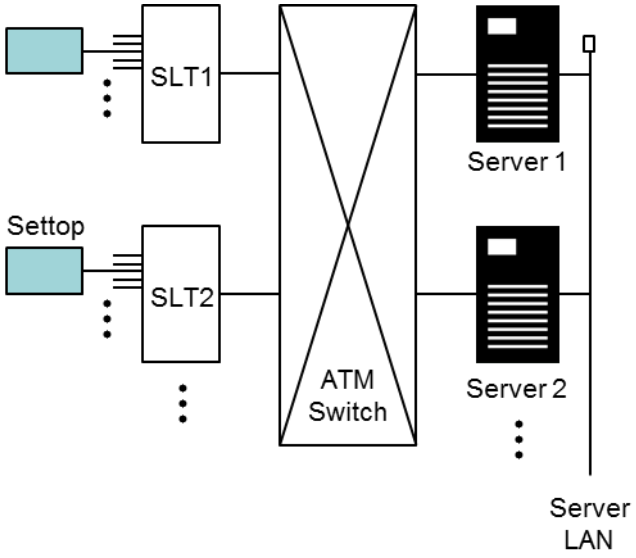


図 2-1. システム構成

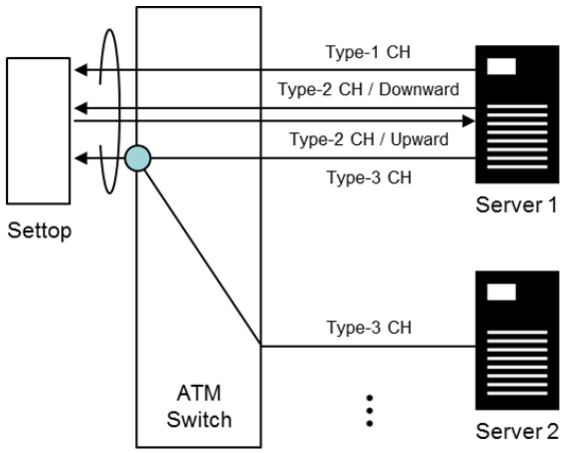


図 2-2. セットトップ・サーバ間のチャネル構成

このようなシステムでは，各タイプのチャネルについて，それぞれ，以下のような課題が挙げられる．

【タイプ 1 チャネルの帯域に関する課題】

図 1 における SLT と ATM 交換機感の物理リンクの帯域は，複数のセットトップで共用される．タイプ 2 チャネル用の帯域を共用セットトップ分，確保しておく必要がある．また一旦，タイプ 3 チャネルのコネクションが確立されると，帯域保証が優先的に行われることから，タイプ 3 チャネルの帯域が共用セットトップ分確保される．従って，この物理リンクにおいて残った帯域が，タイプ 1 チャネルで利用できることになる．このような条件のもとで，ネットワークに対する待ち時間，およびリソースの使用効率を考慮して，物理リンクを共用するセットトップ数の最適値を決めることが課題となる．

【タイプ 2 チャネルにおけるサービス品質関する課題】

ネットワーク型のゲームアプリケーションでは，他のアプリケーションに比べ，タイプ 2 チャネルの使用率が高くなること，レスポンスタイムに対する要求品質が厳しいことから，このタイプのゲームアプリケーションを中心に設計を進めるのがよいと考えられる．ネットワーク型のゲームアプリケーションに対する要求条件は，以下の 2 点である．

(i)　1/60 秒内にセットトップのゲームコントローラから送られるデータの交換と TV への描画が終了すること．
(ii)　音声会話の際の違和感を少なくするため，音声の実時間処理が可能であること．

(i)は家庭用ゲームと同じ品質であり，1/60 秒の間に交換されるデータは 10kbps 程度で，ゲーム空間における各ユーザの位置等をセットトップ間で同期させるための情報やコントローラからのキー入力情報である．また，(ii)については，電話程度の品質を提供することを踏まえ，必要帯域は 64kbps とな

る．これらから，タイプ2チャネルに必要な帯域は，100kbpsのオーダーであり，比較的小さいことから帯域確保上の問題はないと考えられる．

しかしながら，その通信経路上の処理系であり，複数ユーザによる音声会話を実現するサーバについても，上記性能を満足する必要がある．この性能はネットワークインタフェースも含むサーバ側の処理速度に依存しており，主にCPU負荷に左右される．従って，ゲームアプリケーションでは，サービスへの不快感をユーザに与えないよう，サーバにかかるCPU負荷を制御することが課題となる．

【タイプ3チャネルの設計に関する課題】

タイプ3チャネルは他のチャネルに比べて広帯域であるため，ネットワークリソースを効率的に利用すべく，実トラヒック見合いの設計が重要となる．一般的に，ATMネットワークのトラヒック設計は，算出対象，品質規定項目から3つのフェーズに分類される[2-6]．

(i)　対地ごとのトラヒック予測値と呼損率規定値から，必要VC（Virtual Connection）数を算出する．
(ii)　セルレベルの品質規定値を入力とし，必要VP（Virtual Path）帯域を算出する．
(iii) VPを束ね，媒体に収容する．

タイプ3チャネルのように，VC帯域が単一速度，単一品質クラスで，かつ各対地が直通ルートで結ばれている場合，VCコネクションがポアソン到着すると仮定し，アーランB式から必要VC数を算出することができる．しかしながら，ネットワーク設計の第一段階において必須となる対地毎のトラヒック予測，すなわち実験段階にある高速，広帯域の双方向マルチメディアサービスに対するネットワークの需要予測は困難であり，実際にはかなり安全側の設計をせざるを得ない．

そこで，本研究では，サービス開始後にトラヒックデータを測定・分析し，サービス開始前のネットワーク設計を修正することにより，効率的なリソース運用を図ることを考える．このような方法では，修正結果をシステム構成に影響なく反映させることも重要な課題となる．

2.3. 品質のモデル化

本項では，前項で示した3つのタイプのチャネルにおける課題，それぞれを踏まえ，品質のモデル化を考察する．

2.3.1. アプリケーションダウンロード用のチャネルと品質

ある時刻に1台のセットトップがタイプ1チャネルを利用する確率をpとする．同時にk台のセットトップがタイプ1チャネルを利用する確率q(k)は，SLTとATM交換機間の物理リンクを共用するセットトップ数をmとすると，以下の式で表される．

$$q(k) = \frac{m!}{(m-k)!\,k!}\, p^k (1-p)^{m-k} \qquad (1)$$

タイプ1チャネル用に割り当てられた帯域を同時に利用できるセットトップの数をr_0とし，1アプリケーションプログラムをダウンロードする時間をwとすると，k台のセットトップがタイプ1チャネルを使用するときの，1セットトップあたりの待ち時間の平均d(k)は，以下のようになる（但し，r_1は，k÷r_0の商，r_2は，k÷r_0の余り）．

$$d(k) = \frac{w}{2k}(r_1 + 1)(r_0 r_1 + 2r_2) \qquad (2)$$

これらから，タイプ 1 チャネルを利用する際のセットトップのネットワークにおける待ち時間の平均値 W は，次の式で計算される．

$$W = \sum_{k=1}^{m} q(k)\, d(k) \qquad (3)$$

2.3.2. ネットワーク型ゲームアプリケーションのサーバ処理と品質

サーバにおける CPU 負荷は，起動されたアプリケーションプロセスの数に伴い，単調増加する．安定したサービス品質を提供するためには，運用時の起動アプリケーションプロセス数を適切に設定することが重要である．このため，アプリケーションプロセスごとに最大起動数を設定できる機能が必要となる．以下では，先ず CPU 負荷と ATM インタフェース上でのセル棄却率の関係をモデル化し，これをもとにサービス品質の観点から，起動アプリケーションプロセス数の設定値を決めていく．

双方向マルチメディアサービスでは，さまざまなアプリケーションが同一サーバ上で実行されるため，比較的処理が優先されるアプリケーションに対しては，予め適切なCPUリソースを割り当てておくべきである[2-7]．例えば，ネットワーク型のゲームアプリケーションにおける音声合成処理は優先度が高いと考えられ，それに応じてCPUリソースも割り当てられる．そこで本検討では，アプリケーションによる負荷が割り当てられたCPUリソースを越えた場合にセル棄却が発生し，サービス品質が劣化するものとしてモデル化を行った．この場合，セル棄却率bは，サーバに到着するセル数をc，割り当てられたCPUリソースによって処理可能なセル数をc_0とすると，次式で与えられる．

$$\left.\begin{aligned} b &= 0 \quad (c < c_0) \\ b &= \frac{c - c_0}{c} \quad (c \leqq c_0) \end{aligned}\right\} \tag{4}$$

一方，1ユーザが単位時間あたりに発生するセル数をλ_0，1ユーザにサービスを提供するために必要なCPUリソースをs_0とすると，サーバに到着するセル数cとそれを処理するために必要なCPUリソースsの関係は以下の式で求められる．

$$c = \frac{l_0}{s_0} s \tag{5}$$

2.3.3. 実測データに基づく映像ストリームのネットワーク設計

実測された対地トラヒックに対し，呼損率規定値を与え，アーランB式から算出される呼損率がこの規定値を下回る最小のVC数を算出する．呼損率規定値をB，呼量をa，必要VC数をnとすると，以下の式で算出される[2-8]．

$$\min n, \text{ subject to } \frac{\frac{a^n}{n!}}{\sum_{x=0}^{n} \frac{a^x}{x}} \leqq B \qquad (6)$$

上式から算出される必要VC数をn_0，タイプ1，2，3チャネルの帯域をそれぞれ，t_1，t_2，t_3，全セットトップ数をN，全サーバ数をM，先に述べたタイプ1チャネルに関する共用度をC，1ATMインタフェースの帯域をf_0とすると，サーバごとに必要なATMインタフェース数fは，以下の式で算出される．

但し，Gはシステム設計上のパラメータで，算出されたn_0に対するトラヒックが一つのサーバに集中する割合である．各ATMインタフェースへのVC収容では，負荷分散や危険分散の観点から，各タイプのチャネルはf個のATMインタフェースに対して均等に収容されることが重要である．

$$f = \frac{\frac{N}{MC} t_1 + \frac{2N}{M} t_2 + Gn_0 t_3}{f_0} \qquad (7)$$

2.4. 評価と考察

以下では，前項までに検討したモデルを，実際に構築したシステムによって検証する.

2.4.1. アプリケーションダウンロード用のチャネルと品質

式(3)に実システムの仕様（SLT と ATM 交換機間の物理リンク：STM-1，各タイプのチャネル帯域：t_1＝8.0Mbps，t_2＝1.5Mbps，t_3＝4.5Mbps）を適用し，ダウンロードされるアプリケーションプログラムの大きさを 2.0Mbytes（双方向マルチメディアサービスにいて十分な値と考えられる）とすると，1 物理リンクを共用するセットトップ数と待ち時間の平均値の関係は，図 2-3 のようになる．本システムでは，1 物理リンクを共用するセットトップ数は 16 と固定であるため，式(3)の妥当性は厳密には検証できないが，そのセットトップ数が 16 台を超えると急激に待ち時間が大きくなることが分かる．これはタイプ 1 チャネルに割り当てられる帯域が他のタイプのチャネルに圧迫され，タイプ 1 チャネルを同時に利用できるセットトップ数が，それを利用する確率に比べ，小さくなるためと考えられる.

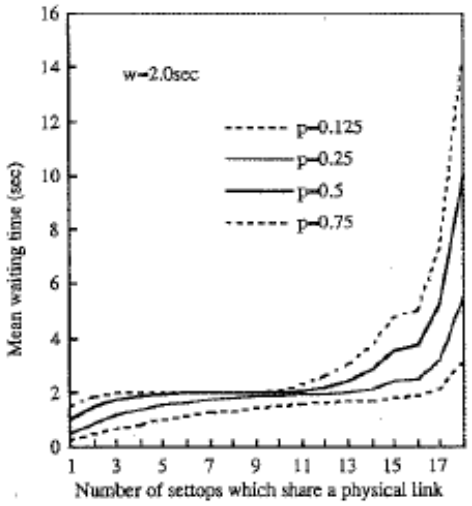


図 2-3. 物理リンクを共用するセットトップ数と待ち時間の平均値

また，1 台のセットトップがタイプ 1 チャネルを利用する確率をパラメータとして式(2)における待ち時間の平均を実測によって算出することができ，その結果を図 2-4 に示す．計測では，1 物理リンクに収容される 16 台すべてのセットトップにおいて映像を視聴している状態で，上記確率に基づいて同時にアプリケーションプログラム（約 400Kbytes）のダウンロードが伴う操作を行い，セットトップ側のレジデントアプリケーションが利用する UNIX 系のシステム時計によって，ダウンロードに要する時間を測定した．この結果から，その傾向はモデルと実測とでほぼ同じことが分かる．

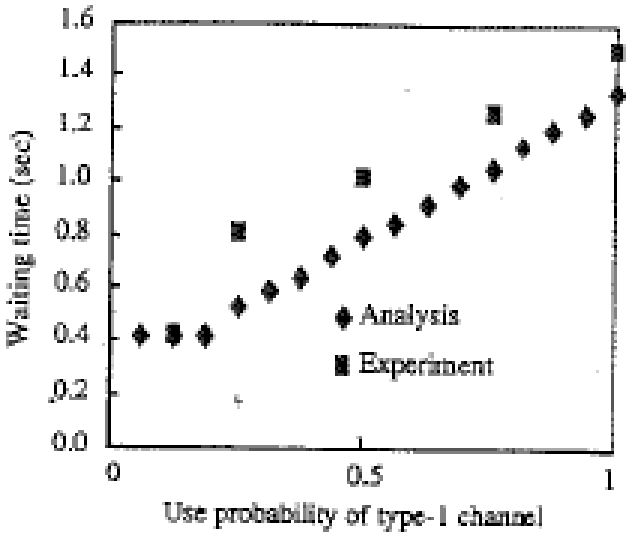


図 2-4. 物理リンクを共用するセットトップ数＝16 の場合の利用率と待ち時間

2.4.2. ネットワーク型ゲームアプリケーションのサーバ処理と品質

ネットワークゲームにおいて音声通信アプリケーションが利用されるケースについて検証を行った．まず，1 ユーザが音声通信アプリケーションを使用する際の CPU 負荷 s_0 を実測し，これと同等の負荷を与えるプログラムを作成した．次に，これを実行することにより，段階的に CPU 負荷を変化させたうえ

で，実際に音声通信アプリケーションを使用し，ATM インタフェース上のセル棄却率を UNIX 系のサーバ OS に付属するアクティビティ監視ツールを用いて計測した．また，式(4)，式(5)において同じ条件を設定し，セル棄却率を求めた（図 2-5）．図 2-5 で与えたパラメータは本システムに基づくものであり，音声通信アプリケーションが使用可能な CPU リソースは全体の 75%，1 ユーザが音声通信アプリケーションにより使用する CPU リソースは 0.68%（実測値）である．

得られた結果より，セル棄却が起こり始める，すなわち CPU 負荷が割り当てられた CPU リソースをわずかに超えた際，モデルよりも実測値においてセル棄却数が小さくなっていることが分かる．従って，この範囲では，利用アプリケーション数をモデルにより安全側に設定することができる．また，それよりも更に CPU 負荷が大きくなると，セル棄却率の実測値がモデルよりも大きい値を示している．これは CPU 負荷が上がるにつれて処理効率が低下するため，棄却されるセル量も大きくなると考えられる．

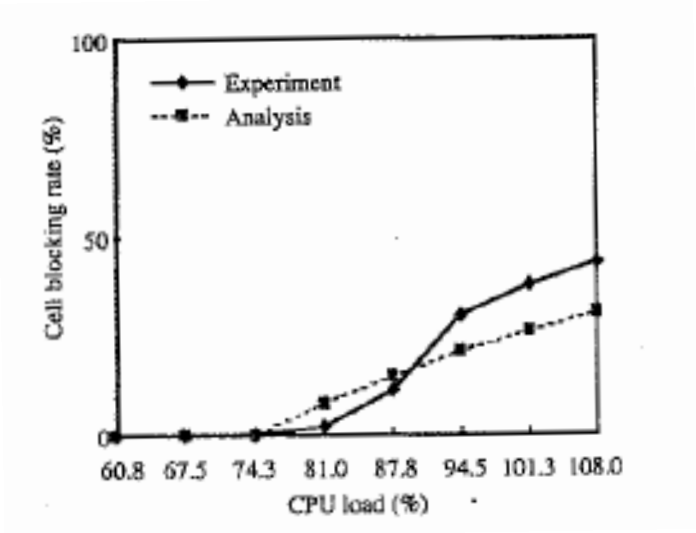


図 2-5. CPU 負荷とセル棄却率

更に，セル棄却率とユーザが感じる劣化について評価を行った．視聴において，CPU 負荷が比較的低く，セル棄却が起こらない状態では若干の固定遅延が感じられた．また，セル棄却が起こる直前で音声に揺らぎのある遅れが感じられるようになり，セル棄却が始まると音の途切れが発生した．本検討では，遅延揺らぎ狭い範囲でしか起こらず，妨害感もわずかであることから，ユーザが感じる劣化の支配的な要因と考えられる音の途切れを主な対象として評価を行った．

被験者は，25 歳から 35 歳の男女で業務系，技術系の会社員を合わせて 14 名であり，主観評価により，MOS（Mean Opinion Score）値を測定した．セットトップ付属のマイクロホンを口元より 15cm 程度離し，市販のヘッドホンを用いて，2 人ペアで会話することにより試験を実施した．評価時の音量は被験者自身が聞き取りやすいよう事前に調整し，評価の最中はこれを変えないようにした．また，普段使っている電話と比較するよう指示を与え，MOS 値を測定した（図 2-6）．

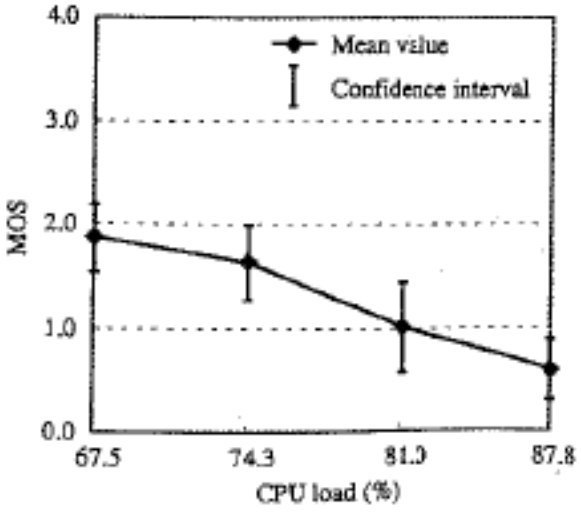


図 2-6. CPU 負荷と会話音声に対する MOS 値

評価結果より，CPU 負荷が大きくなると評価値も下がる傾向が見られる．また，信頼区間を比較することにより，試験条件として設定した CPU 負荷のうち最良のものとセル棄却が発生しているものの間に違いがあることが確認できる．これは，セル棄却による音の途切れが品質に大きな影響を与えているためと考えられる．もともと音声の品質がそれほど高くなかったことから，セル棄却起こった際にはかなり低い評価値となっている．これらより，実際の運用ではセル棄却が起こらない範囲でアプリケーションを使用できるユーザ数を設定することが望ましいと考えられる．

2.4.3. 実システムにおけるネットワーク設計の修正

実際に，本システムを千葉県浦安市における約 350 世帯の一般家庭に利用していただき，サーバに出力されるシステムログから，映像ストリームのコネクション到着間隔およびサービス時間を算出した．測定はサービス利用率が最も高い日曜日の 12 時から 22 時まで行った．これは，本システムが娯楽性の高いシステムであり，日中だけでなく夜間のサービス利用が比較的多いためである．サービス時間の分布ならびに，サービスごとのコネクション到着間隔分布の詳細は，文献[2-9]を参照いただきたい．

サービス時間分布には 3 つの山が観測されたが，これらは本システムにおけるサービスの特徴と考えらえる．映像ストリームを利用するケースは主に 3 種類あり，ひとつは各種アプリケーションのナビゲーション画面における動画像，ひとつはゲームアプリケーションにおけるバックグランドミュージック，ひとつは VOD 系のアプリケーションにおける映画のストリームである．以下では，これらをサービス時間の長さの観点からそれぞれ，short サービス，middle サービス，long サービスと呼ぶことにする．

これらのサービスはトラヒック特性も大きく異なるため，独立に必要 VC 数を算出するが，その結果を表 2-1 に示す（呼損率規定値＝10^{-5}）．映像ストリー

ムは帯域保証されているため，各々の VC 数の単純な加算により安全側の設計を行うことができる．これによると，サーバと ATM 交換機間には計 37 個のタイプ 3 チャネルを設定する必要がある．また，本検討では全アクセス要求が 1 つのサーバに集中する最悪の場合を想定し，各サーバに 37 個のタイプ 3 チャネルが必要であるとした．

表 2-1. サービスごとの呼量と必要 VC 数

Kind of service	Mean service time	Mean arrival time	Offered load	VC
Long service	6,703.2 (sec)	2,704.9 (sec)	2.48	12
Middle service	158.8 (sec)	55.2 (sec)	2.88	13
Short service	22.9 (sec)	10.7 (sec)	2.15	12

図 2-7 に本システムの各サーバ（計 4 台）における ATM インタフェースの帯域使用分布を示す．サービス開始後に修正した帯域に対する実使用帯域の割合が最大でも 56%であることを考慮しても，検討したモデルが安全側の設計になっていることが分かる．

一般に，システム構築後のネットワーク構成の変更が困難な場合が多いが，ここではシステムへの影響が比較的少ない VC 構成の変更のみにより，サービス開始後に設計値の補正が行えることを示した．

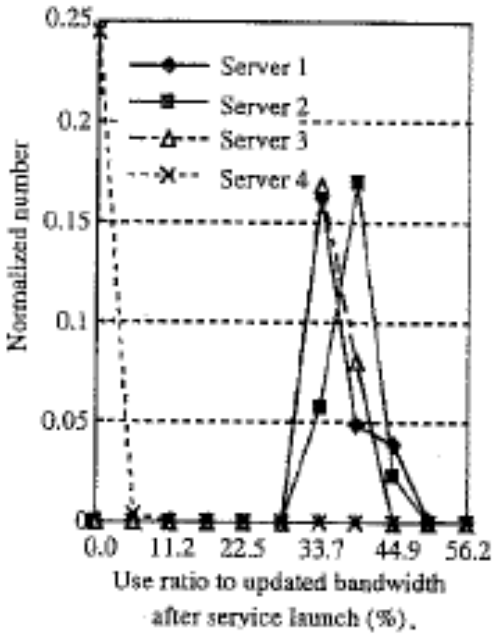


図 2-7. ATM インタフェースの帯域使用分布

2.5. まとめ

一般家庭を対象とした双方向マルチメディアサービスのためのシステム設計法について，特に可用性の観点から，検討を行った．サービスアプリケーションの通信に関する特徴から ATM ネットワークにおける VC を 3 つに分類し，それぞれについて設計上の課題をとらえ，モデル化による分析を進めた．

その結果，(1) 加入者側伝送リンクの利用率向上を図る共用端末数の設定，(2) ユーザに不快感を与えない音声通信アプリケーション起動数の設定，(3) 実トラヒック見合いの必要 VC 数算出とシステム構成に影響を与えない VC 収容法を提案するとともに，実システムによりデータを取得し，評価を行うことにより，モデルの妥当性を検証した．

第 3 章

通信ログ分析に基づく高精度なマルウェア感染検知

以前のサイバー攻撃への対策では，複数のセキュリティ・アプライアンスを導入する，いわゆる多層防御が効果的とされてきたが，昨今は，単に”守る”といった対策だけでは不十分になってきている．これは，近年のサイバー攻撃の手法が巧妙化してきており，各システム，アプライアンス単体，またはそのログだけを確認しても攻撃が起きているかどうかの判断ができなくなっているからである．そこで，多種多様なシステムやアプライアンスのログを一元的に集め，リアルタイムに相関分析等を行うことで，脅威と思われる兆候を見出し，さらなる詳細分析に繋げるための SIEM（Security Information and Event Management）といったコンセプト（またはシステム）が重要視されている．

その一方で，SIEM 構築や運用にかかる手間暇は，その難しさ（効果的な検知手法の確立）や膨大なログの分析に伴う稼働量の大きさ故，課題となっている．本章では，SIEM 等が取り扱うログ分析において，悪性通信ログ判定を行うにあたって，従来手法を上回る高精度なマルウェア感染端末の検知技術について提案する．

3.1. 研究の背景と課題

マルウェア感染端末の検知の一手法として，ファイアウォールやプロキシサーバ等のネットワークの出口の装置にて記録される通信ログを分析し，判定を行う手法がある．通信ログに対し，通信先 FQDN（Fully Qualified Domain Name）や URL をブラックリストと突合したり，マルウェア感染時の通信の振る舞いをルールとして規定し，そのルールに合致するかを探索したりする等の分析・判定がよく行われている．

一方，コンピュータ性能の高まりやマルウェア検体等の過去のデータの十分な蓄積に伴い，機械学習の手法を用いてより効率的かつ高精度な通信ログの分析・判定を可能とする研究が盛んに提案されつつある．一般的な機械学習手法では，対象データから注目すべきいくつかの情報を特徴として抽出，数値化し，その数値データである特徴ベクトルを機械学習アルゴリズムにより学習・推論することが行われる．そのため，機械学習アルゴリズムのみならず，特徴抽出手法もその性能を作用する重要な要素になる．

通信ログ（テキスト）のうち，特に URL や User-Agent 等の特徴抽出手法として有名なのが，Bag of Words (BoW)の手法である．これは，適切な区切り文字や形態素解析等で，テキストを語として分解し，語１つ１つを特徴ベクトルの１つの次元とし，語の出現数等について適切な重みで数値化を行うものである．因みに，セキュリティ知識を生かして，例えば，通信先 URL 中にダッシュ‘－’が多いものはマルウェアである等，ヒューリスティックな手法を特徴抽出として用いるものも提案されている．また，文献[3-1]のような，URL をいったん正規表現等のパターンに変換した後，クラスタリングを実施しており，パターン化によって類似した特徴同士を１つに纏め上げることに成功しているものもある．

しかしながら，しばしば適用される BoW の手法は，ちょっとしたテキストのパターンの変化が頻出する通信ログに対しては，スケールしないという課題がある．例えば，以下の URL において，

http://www.xxx.co.jp/index.html/key1=val1&key2=val2&...

「val1」「val2」は変数の値であり，これらひとつひとつを語として抽出していくため，次元が非常に大きなものになってしまう．

本章では，通信ログにおけるマルウェア感染端末判定において，特に重要な情報源となる通信先の URL からの特徴抽出手法として，データ圧縮アルゴリズムを用い，その圧縮率を特徴として用いる手法を提案する．また，本特徴抽出

手法を教師あり学習に適用することで，BoW といった従来の特徴抽出手法を用いたマルウェア感染端末判定よりも高精度な判定が実現できることを示す．

3.2. データ圧縮による特徴抽出及び分類

Benedetto らは情報エントロピーや Kolmogorov 複雑性等の情報量の指標のひとつとして，データ圧縮アルゴリズムを用いた指標である相対エントロピー(relative entropy)を提案している[2]．同じコードの連続やコードパターンの繰り返し等，何らかの類似パターンが出現するデータについてデータ圧縮アルゴリズムは良く圧縮する点に注目すると，データ A から見たデータ x の情報量の多さは，その圧縮されにくさに関連するという考えに基づいている．

この提案については，分類問題への応用が既にいくつか試みられている[3-2][3-3][3-4][3-5][3-6]．これらの試みは，推論データ x が分類 A，B のいずれかであることを推論するのに，x を各分類の訓練データ A，B のより類似している方に分類するという自然な考えに基づいている．類似度は，相対エントロピーの少なさで測定でき，より相対エントロピーが小さい方に推論データは分類されうる．特に，文献[3-5]は SPAM フィルタへの適用に関するものであり，従来の BoW による特徴抽出と教師あり学習による手法よりも高い精度が得られたと報告している．

ある通信ログが悪性か良性かを分類するイメージについて，図 3-1 に示す．ここでは，悪性および良性の通信ログに対するそれぞれの訓練データの最後尾に検査データ（推論対象）を結合したうえでデータ圧縮を施し，圧縮後のサイズを評価することで分類を行っている．すなわち，悪性系の通信ログを用いた圧縮後のデータサイズが，良性系の通信ログを用いた圧縮後のデータサイズに比べ十分小さければ，推論対象は悪性と判定できる．

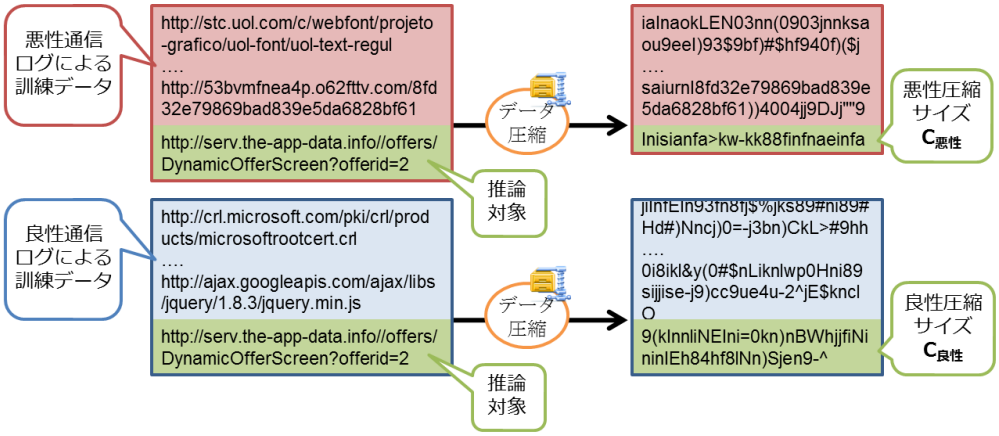


図 3-1. データ圧縮サイズの評価による悪性・良性分類

因みに，データ圧縮アルゴリズムとしては，テキストに対して LZSS (gzip)， LZW (compress)，PMP (rar) が利用できる，との報告がなされている[3-3][3-4][3-5][3-6]．また，楽譜から作曲者を判定する手法においては bizp2 が有効であるとの報告[3-7]がある．

3.3. 提案手法

通信先 URL に対してデータ圧縮アルゴリムを用いた特徴抽出を行い，その特徴ベクトルを用いて教師あり機械学習による分類手法を提案する．圧縮アルゴリズムを用いることにより，URL 文字列中の同一パターンをデータ圧縮アルゴリズムが自動的に効率よく認識する可能性がある．

従来の圧縮による分類手法は，SPAM メールのテキストや Tweet 等，1 データすべてを 1 つの属性と見て，データ圧縮アルゴリズムによる分類を行っていた．マルウェアの通信先 URL については，同種の C&C サーバが複数台あり，状況に応じてその通信先を変更する等の事例が見られることから，少なくとも FQDN と Path 以降の 2 属性に分けて判定したほうがより良く判定できることが

期待される．そこで，複数の属性に対応可能とするため，従来のようにデータ圧縮アルゴリズムで直接分類するのではなく，相対エントロピーを用いた特徴として抽出し，その特徴を改めて教師あり学習を用いて分類する手法を提案する．以下に，その手順を列挙する．

① 訓練データ（悪性の通信ログ，良性の通信ログ）を事例選択する．
② 事例選択した訓練データで，指定された属性（FQDN 等）をデータ圧縮特徴抽出器へ登録する．
③ 登録済みデータ圧縮特徴抽出器を用い，訓練データから特徴抽出を行って属性ごとに悪性圧縮率(Zpos)と良性圧縮率(Zneg)を，それぞれ特徴ベクトルとして得る．
④ 上記特徴ベクトルに変換された訓練データにより，悪性の通信ログに含まれるデータを悪性，良性の通信ログに含まれるデータを良性としてクラス付けを行い，教師あり分類器を訓練する．
⑤ 推論時に，登録済みデータ圧縮特徴抽出器を用いて推論データ(判定対象の通信ログ)から特徴抽出を行い，訓練済みの分類器により推論を行い，スコアを得る．
⑥ 判定閾値以上のスコアとなった通信ログの発信元端末を感染端末と判定する．

特に，手順①の事例選択では，感染端末通信にもしばしば存在する，非感染端末がよくアクセスする URL を悪性の通信ログから削減する．例えば，マルウェア「OutBrowse」の通信先 URL を時系列順に表 3-1 に示す．4 行目は Google 社が提供する API の URL であり，非感染端末からも利用される URL である．これら良性・悪性ともに出現する URL については，データ圧縮での分類にて著しく判定精度を悪化させるであろうことを考慮し，a) 良性の通信ログにも出現する URL を悪性の通信ログより除去，b) 各通信ログ内で重複するものを 1 つのみ残して削減，といった 2 つの操作により事例選択を施す．

表 3-1. マルウェア OutBrowse の通信先 URL

1:	http://installer.apps-track.com/Installer/Flow?pubid=7302&distid=13467&productid=1694&subpubid=-1&campaignid=0&networkid=0&dfb=-1&os=5.1&iev=8.0&ffv=&chromev=&macaddress=&netv=&d1=15693&d2=-1&d3=-1&d4=-1&d5=-1&ds1=&hb=2&systembit=32&vm=1&version=4.0
2:	http://serv.the-app-data.info//offers/DynamicOfferScreen?offerid=2&distid=13467&leadp=1694&countryid=142&sysbit=32&dfb=-1&hb=2&isagg=0&version=4.0&external=0&external=0&
3:	http://cdn.file7desktop.com/ProductS/PlayerStubWrapper1.exe
4:	http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js …(中略)…
5:	http://d.castplatform.com/api/vp/1?clk=gLg_PHWA9aSyioXkt-F4b3J9cI1ybf-t-x7VxWH5dmAWXwln-z (以下略)

また，手順②の訓練データのデータ圧縮特徴抽出器への登録方法は用いるデータ圧縮アルゴリズムにより異なるが，詳細は 3.4 項において説明する．なお，訓練データの形式は URL 属性であれば，表 3-1 のような URL のリストを良性ならびに悪性ごとに取りまとめたものとなる．FQDN 属性や Path 属性でも同様に（事例選択した）良性，悪性ごとの FQDN や Path のリストをそのまま訓練データとする．

手順③のデータ x の悪性圧縮率 Z_pos および，良性圧縮率 Z_neg の定義は以下のとおりである；

$$Z\_pos\ (x) = (C\_pos\ (x) + \gamma) / (L(x) + \gamma)$$

$$Z\_neg\ (x) = (C\_neg\ (x) + \gamma) / (L(x) + \gamma)$$

ただし，C_pos (x)，C_neg (x) はそれぞれ悪性の通信ログおよび良性の通信ログとデータ x の相対エントロピー，L(x)は x のデータサイズ，γはスムージング（機械学習のための調整）パラメータを示す．

3.4. 実験方法

本提案の方式において，どのようなデータ圧縮アルゴリズムおよび教師あり学習の分類器が適しているかについては，BoW のような従来方式との比較とともに興味を引くところである．実験ではまず，判定精度および速度性能の面から，どのデータ圧縮アルゴリズムが本方式に適するかを明らかにした．次にその適したデータ圧縮アルゴリズムを用いて，分類器および特徴抽出に用いる属性を変えつつ，感染端末の判定精度を評価した．また，ブラックリストを用いた判定および，URL の特徴抽出に BoW を用いたサポートベクタマシン（SVM）での判定精度を対照実験として行った．

判定精度の検証では，時系列ホールドアウト検証（フォワードテスト）[3-8]を行い，検査データは訓練データよりも時間的に後のものを用いた．判定精度の指標としては，AUC（Area Under the Curve），pAUC（partial AUC）[3-9]，および，誤判定率（False Positive Rate）FPR ＜ 0.5% となるように判定閾値を調整した際の判定率（True Positive Rate）TPR（FPR＝0.5%）を用いた[8]．因みに，AUC は判定閾値を変化させたときに FPR を横軸に，TPR を縦軸にとったときの曲線下の面積，pAUC は特定の FPR の範囲［p1，p2］での曲面下の面積として与えられる．

pAUC は［p1，p2］＝［0，0.1］で算出されることが多く，本実験でもそれに倣っている．感染端末は非感染端末よりもごく少数であることから，誤判定が少ないことがより重要であり，pAUC および TPR（FPR＝0.5%） は，低誤判定率時の判定率を評価する指標として扱うことができる．

データとしては，マルウェア検体の動的解析により収集した通信ログを悪性ログとし，社内ネットワークのプロキシログを良性ログと見なしたものを用いている．本実験で利用したログの諸元は以下のとおりである．なお，悪性ログについては，「1 検体，1 端末に対応」として評価した．

＜悪性ログ＞

2015 年 2～7 月採取のマルウェア検体の動的解析

⇒検体数(端末数) 71,310 件, ログ件数 7,152,479 件

＜良性ログ＞

2014 年 2～3 月採取の社内 NW プロキシログ

⇒端末数 1,940 件, ログ件数 36,581,398 件

3.4.1. データ圧縮アルゴリズムの選定

特徴選択に用いるデータ圧縮アルゴリズムとして, LZSS (zip, LZ77), LZT (compress の圧縮アルゴリズム LZW, LZ78 の亜種) [3-10], bzip2, LZMA の 4 つを検証した. 検証においては, 時系列ホールドアウト検証を適用し, 悪性ログと良性ログの（時系列的に）前半分のそれぞれ先頭 10 万件を訓練データとして, 同様に（時系列的に）後半分のそれぞれ先頭 10 万件を検証データとした. 実施手順として, 属性は URL を用い, 提案方式の 1～3 までを行い, 良性・悪性のそれぞれの圧縮率から, 以下に示す式によりスコアを算出し, 評価を行った.

Score＝Z neg／Z pos

事例選択済み訓練データの登録については, 各データ圧縮アルゴリズムの特性に応じ, 以下のとおり実施した.

LZSS は, スライド辞書方式による圧縮を行うアルゴリズムである. 圧縮対象のデータ列は, その直前のスライド窓の領域のデータの中から最長一致するデータ部分を検索し, その相対位置とサイズのみを記録することで, データ圧縮

を行う．そのため，スライド窓からはみ出した前方部分の情報は圧縮に加味されないことになる．一般的なLZSSのスライド窓は32kBである．本実験では，訓練データを20kBに分割して登録した．対象のデータxの圧縮率を算出する際は，登録された各分割訓練データそれぞれにデータxを結合してLZSS圧縮し，もっとも圧縮率が小さいものを採用することとした．これにより圧縮に加味されない訓練データを極力無くすことができる．

LZTは，compressやgif形式ファイルで用いられているデータ圧縮アルゴリズムLZWの亜種であり，LZWと同様に動的辞書方式による圧縮を行うアルゴリズムである．逐次，新規のデータ列があれば辞書に追加して行き，圧縮対象のデータは辞書で最長一致したデータ列の辞書No.で代替されることで，データ圧縮を行う．辞書は，Trie木（トライ木と呼ばれる順序付き木の一種）により作成される．本実験では，訓練データをTrie木の辞書に登録する．データxの圧縮率は，この辞書のみを参照してxをデータ圧縮し，算出することとした．辞書No.のbit長が辞書の大きさを決めるが，本実験では24bit長を用いた．LZWとLZTの違いは，前者が辞書満杯となった場合，以降の登録作業を中止してしまうのに対して，後者は利用されていないデータ列を辞書から破棄（LRU: Least Recently Used）して，新規データ列の登録を続行する点である．

bzip2およびLZMAについては，前者がブロックソート，後者がMarkovアルゴリズムを用いてデータ圧縮を行う．LZSSと同様，bizip2はブロック領域，LZMAはスライド辞書領域を必要とするが，その領域はLZSSよりも十分大きい．そこで本実験では，訓練データの登録時はそのままこのデータを保持することとした．データxの圧縮率は，単純にこの訓練データとxを結合して圧縮することで求める．

LZSS，bzip2，LZMAは，Pythonのそれぞれlibz，libzip2，liblzmaを利用することで実施している．LZTについては，Cythonを用いて実装した．

スムージングパラメータは，すべて，$\gamma=10$固定としている．

3.4.2. 分類器および属性選択の比較

分類器はもともとのデータ圧縮分類の定義により，線形でも十分に分類できると予想されるため，線形分類器を中心に選び，以下の 5 種類；

Ridge 回帰（Ridge）

L2 正則化 SVM（サポートベクタマシン）

Fischer の線形判別法（LDA）

ナイーブベイズ(NB)

AdaBoost

を比較検討した．分類器には，Scikit-Learn のライブラリを使用した．

属性は URL および URL を FQDN と Path 以下の部分（以後，単に Path と呼ぶ）に分けたものとし，URL，FQDN，Path の 3 属性から特徴抽出を行い，その属性の組み合わせ（属性選択）により判定率の違いを検証した．

対照実験としてブラックリスト判定，および，BoW 特徴抽出による L2 正則化 SVM での判定を実施した．

ブラックリストは，悪性ログの通信先 URL を FQDN 部，PATH 部，QUERY 部に分けてそれぞれブラックリストとして保持した．QUERY 部は，Value 部分の値をマスクした．さらに，良性ログとブラックリストを突合することで，合致したものはブラックリストから除いた．こうして作成した 3 つのブラックリストのどれか 1 つにでも送信先 URL が合致した通信ログの通信元ホストがマルウェア感染端末である，と判定した．

BoW 特徴抽出では，「/」「?」「&」「=」の 4 文字を区切り文字として，単語を抽出した.

検証は，時系列ホールドアウト検証で実施し，悪性ログと良性ログの前半分すべてを訓練データ，同様に後半分のすべてを検査データとした．実施にあたっては，データ圧縮を行うものは提案方式の手順①～⑥を行い，スコアは同一通信元ホスト中でもっとも高いスコアをそのホストのスコアとし，端末単位での AUC と pAUC 等の指標を算出した.

スムージングパラメータは，すべて，$\gamma = 10$ 固定としている.

3.5. 実験結果

3.5.1. データ圧縮アルゴリズムの選定

データ圧縮アルゴリズム別に測定した判定率を表 3-2 に示す．これらの指標は，通信ログ単位で測定した指標であり，端末単位での指標より大きい（良い）値が出るので注意を要する.

もっとも判定精度が良いのは LZMA であるが，もっとも実行時間がかかっており，1 千万件以上の通信ログ分析に供するには，実行時間面での困難が予想される．LZT は高速であり，判定精度も優れていることから，後の実験の特徴抽出には LZT を用いることとした．bizp2 はまったく判定できなかったといえる.

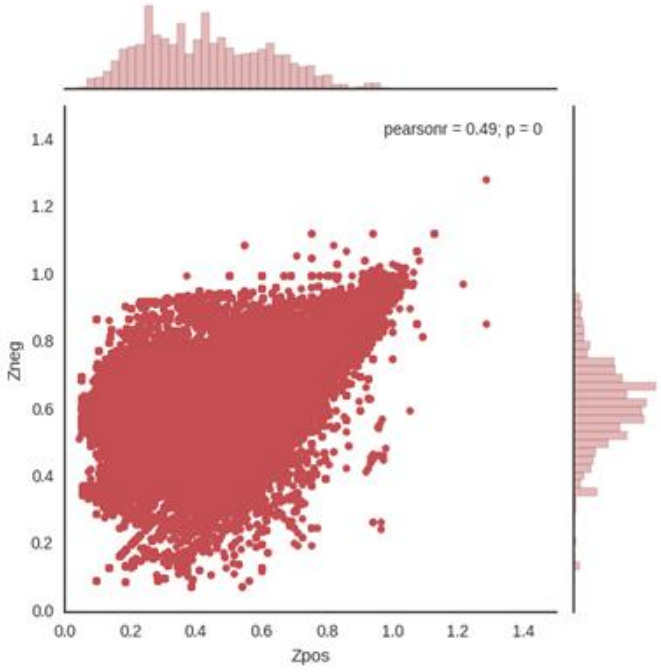
表 3-2. データ圧縮アルゴリズム別判定率(ログ単位，属性 URL)

方式	設定等	実行時間	AUC	pAUC	$TPR_{0.5\%}$
LZSS (gzip)	level 6	16490秒	0.968	0.0797	60.9 %
LZT	辞書長 24bit	20秒	0.973	0.0825	68.1%
bzip2	level 9	68690秒	0.521	0.0057	0.4 %
LZMA	—	98112秒	0.975	0.0828	68.3%

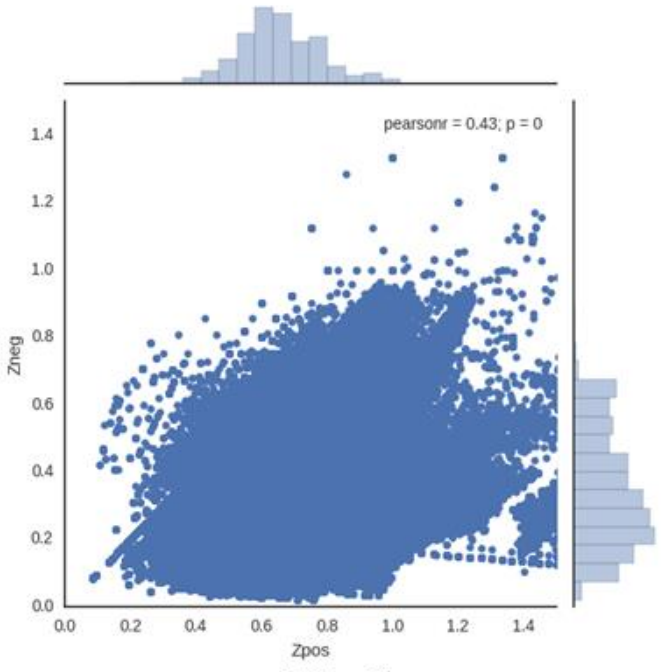
3.5.2. 分類器および属性選択の比較

まず，悪性圧縮率と良性圧縮率によって，悪性と良性を分離可能か検討した．検証データの URL 属性から LZT で特徴抽出を行って，横軸にそのデータの悪性圧縮率，縦軸に良性圧縮率をとり，検査データをプロットした散布図を図 3-2 に示す．悪性ログはグラフの左上に，良性ログはグラフの右下にプロットされているのが見られ，線形推論でも十分に分類可能であることがわかった．

次に，FQDN，Path 属性から特徴抽出（特徴ベクトルは 4 次元）を行い，分類器を変えて判定精度を測定した結果を表 3-3 に示す．L2 正則化 SVM が pAUC＝0.0825，TPR（FPR＝0.5%）＝65.3%と最も良い値を示し，低誤判定率ではよい分類器であるといえる．



a) **悪性ログ**



a) **良性ログ**

図 3-2. URL 良性・悪性圧縮率による検査データ散布

表 3-3. 分類器別判定率

(**ホスト単位，属性**FQDN, Path，**特徴抽出**LZT 24bit)

分類器	AUC	pAUC	$TPR_{0.5\%}$
Ridge α=0.1	0.9268	0.0791	59.4%
SVM C=0.025	0.9306	0.0825	65.3%
LDA	0.9291	0.0784	57.3%
NB	0.8166	0.0602	13.5%
AdaBoost	0.9420	0.0785	58.3%
Blacklist*1	—	—	23.2 %
BoW, SVM*2	0.9030	0.0657	32.0 %

*1 **対照実験**: FPR=0.0%
*2 **対照実験**: URL**を**Bag of Words**で特徴抽出，**C=0.75

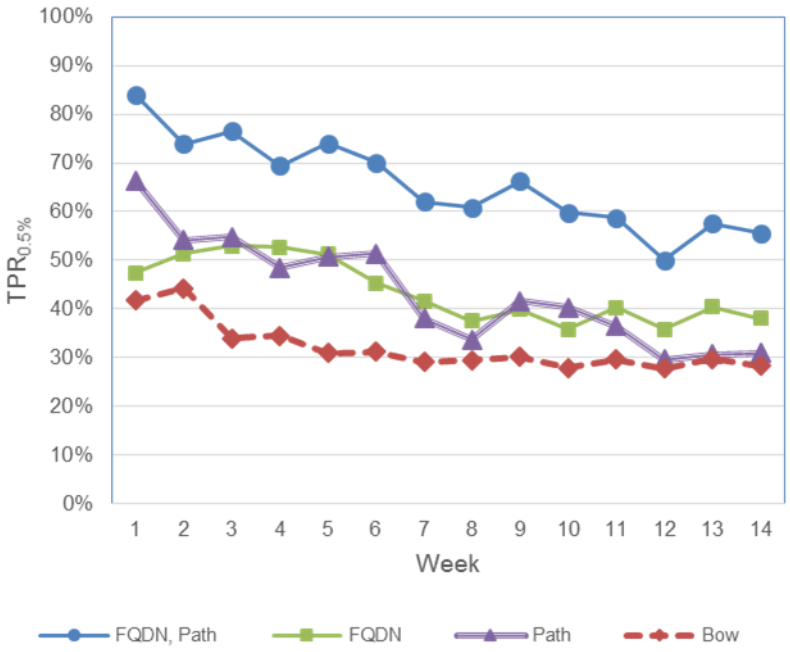
表 3-4 に属性選択を行い，その判定精度を比較した結果を示す．分類器は L2 正則化 SVM を用いた．URL 全体から特徴抽出する場合は，pAUC＝0.0720 に対して，FQDN と Path に分けて特徴抽出した場合は，pAUC＝0.0825 と，2 属性に分けて特徴抽出したほうが良い結果を得られた．

表 3-4. 属性選択別判定率

(**ホスト単位，特徴抽出**LZT 24bit，**分類器**SVM)

属性	AUC	pAUC	$TPR_{0.5\%}$
URL	0.8965	0.0720	41.8%
FQDN	0.8956	0.0631	43.6%
Path	0.7990	0.0562	43.2%
FQDN, Path	0.9306	0.0825	65.3%

また，図 3-3 に，判定率 TPR（FPR＝0.5%）の経時変化を週ごとに追ったグラフを示す．FQDN 及び Path 属性での判定は，第 1 週では判定率 85%であるが，4 週間ごとに約 10%ずつ判定率が落ちていくことがわかる．また，Path 属性が週次低下していくのに対して，FQDN 属性は，8 週目以降には低下が留まる．



(**ホスト単位，特徴抽出**LZT 24bit，**分類器**SVM)

図 3-3. 判定率 TPR（FPR=0.5%）の経時変化

3.6. 考察

なぜ，データ圧縮により適切な特徴抽出を行えるのかを以下に考察する．図 3-4 に URL 属性の悪性圧縮率のヒストグラムを示す．悪性ログのヒストグラムに注目すると，大まかに圧縮率が非常に小さい A，良性ログのピーク値とあまり変わらない圧縮率である B，圧縮率が大きな C の 3 つのピークが存在することがわかる．

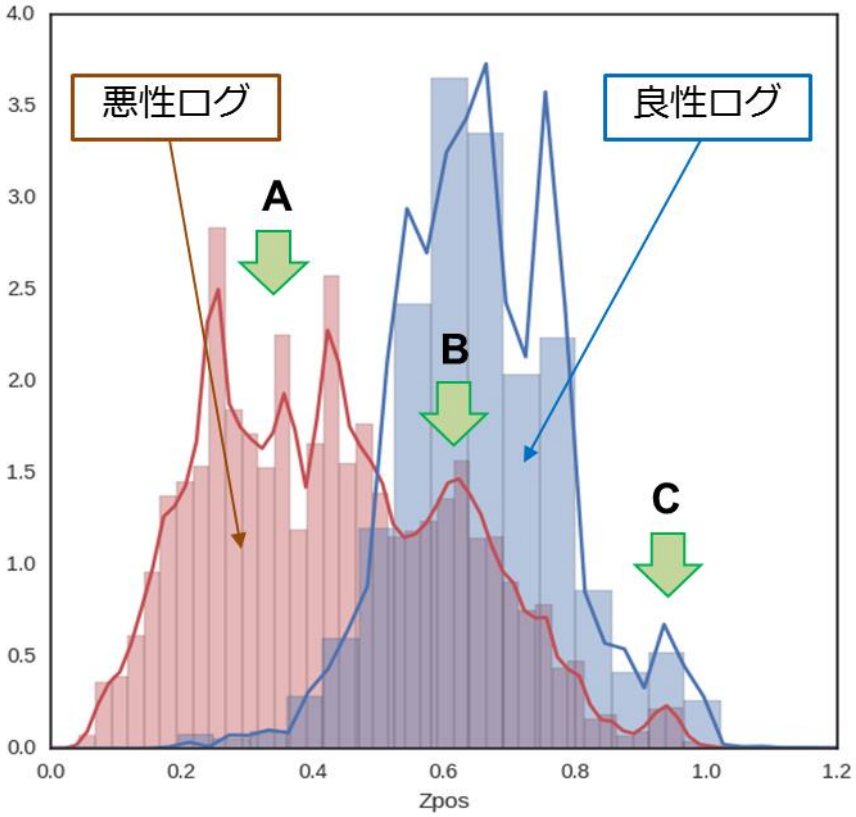


図 3-4. URL 悪性圧縮率によるヒストグラム

A に属する URL として表 3-1 の 1 行目や 2 行目のように，API としてパターン化した URL が存在する．表 3-1 の 2 行目を実際に LZT で圧縮した状態を表 3-5 に示す．表の | はその間のデータ列が圧縮時 1 コードで表されることを示す．未圧縮時に 1,352bit だったデータが，悪性ログとの圧縮を行った場合，220bit にまで圧縮され，かなりのデータ列が 1 コードで表されるのが分かる．Query 部に注目すると，”&distid=1”のように，ほぼ key 名で 1 コードとなっているものや，”countryid=…”ように value も含めて 1 コードとなっているものがある．すなわち，訓練データに存在しなかった key と value の組み合わせは分割し，存在した組み合わせはセットとなるように，データ圧縮が自動的に認識している．このため，API のようなテンプレート的なデータ列は適切に類似性を認識するように思われる．FQDN での顕著な例は，host1.xxx.com,

host2.xxx.com など，ホスト名に連番を含むものが挙げられる．完全一致では異なるFQDNと認識されてしまうが，データ圧縮では，最長データ列一致というデータ圧縮の方針から，|host|2.|xxx.com|のように認識されて類似する(相対エントロピーが少ない)ものとされる．

表 3-5. API 的 URL の圧縮状態

未圧縮: 1352bit: http://serv.the-app-data.info//offers/DynamicOfferScreen?offerid=2&distid=13467&leadp=1694&countryid=142&sysbit=32&dfb=-1&hb=2&isagg=0&version=4.0&external=0&external=0&
良性ログ関連エントロピー: 900bit |http://serv|.the|-ap|p- |data.|info/|/of|fers|/D|yna|mic|Off|erS|cre|en|?o|ffe|rid=2|&dis|tid=1|3467|&le|adp|=16|94&c|ountry|id=14|2&s|ys|bit|=32&|dfb|=-1&|hb=|2&is|agg|=0&ver|sion=|4.0|&ex|ternal|=0&e|xter|nal|=0&|
悪性ログ関連エントロピー: 220bit |http://serv.the-app-data.info//offers/DynamicOfferScreen?offerid=2|&distid=1|3467|&leadp=1694&|countryid=142&sysbit=32&df|b=-1&hb=2|&isagg=0&versio|n=4|.0&e|xternal=0&e|xternal=0&|

*** ||の間が圧縮時に1コードで表されることを示す**

一方でCに属するURLとして，表3-1の5 行目に見られる，符号化・暗号化を伴うURLが存在する．表3-6にその圧縮状態を示す．これは良性・悪性ともに同様に圧縮率が大きくなってしまい，本実験のような線形推論では正しい判定が行えない．しかし，符号化URLは良性・悪性圧縮率ともに大きいという性質を今後活用することは可能であろう．

表 3-6. 符号化 URL の圧縮状態

未圧縮: 4688bit http://d.castplatform.com/api/vp/1?clk=gLg_PHWA9aSyioXkt-F4b3J9cI1ybf-t-x7VxWH5dmAWXwln-z (以下略)
良性ログ関連エントロピー: 4420bit |http://d.|cast|platform|.com/api/|vp|/1|?cl|k=|gLg|_PH|WA9|aS|yio|Xkt|-F4|b3J|9cI|1y|bf-|t-|x7|VxW|H5d|mAW|Xwl|n-|z (以下略)
悪性ログ関連エントロピー: 4980bit A|http://d.|cast|platform|.com/api/v|p/1|?cl|k=|gL|g_|PH|WA|9a|Sy|io|Xk|t-|F4|b3|J9|cI|1yb|f-|t-x|7V|xWH|5d|mA|WX|wl|n-|z

*** ||の間が圧縮時に1コードで表されることを示す**

LZT のデータ圧縮方針である最長データ列一致が，URL の特徴抽出に適していることは，bzip2 ではまったく判定できないことからも推測される．bzip2 は圧縮時にブロックソートを行うが，その際，データ列をばらばらにして並べ替える．このようにデータ列を認識しない圧縮アルゴリズムが URL での判定を行えないことから，URL の文字列が重要な特徴であることを裏付ける．

3.7. まとめ

データ圧縮アルゴリズム LZT を用いた特徴抽出によって，FQDN 部と Path 部の悪性・良性圧縮率を抽出し，L2 正則化 SVM で学習・推論することにより，ブラックリスト判定や BoW による特徴抽出手法よりも優れた判定精度を示した．特に，API 的 URL 等のパターン化した URL，FQDN，Path 間の類似性を認識して特徴抽出を行えるのがデータ圧縮アルゴリズムを用いる利点であると思われる．

一方，提案手法では，符号化 URL を含むログをうまく判定することができず，これは今後の課題と考えられる．

相対エントロピーの算出は，必ずしも悪性・良性両データを必要とするものではなく，どちらか片方のみでも適用可能である．そのため，異常検知手法にも適用できる可能性があり，今後のさらなる検討が期待される．

第 4 章

悪性 Web サイトにおける Evasive コード特定技術

最近のマルウェアには，サンドボックス（外部から受け取ったプログラムを保護された領域で動作させることによって，システムが不正に操作されるのを防ぐセキュリティ機構）による解析を回避する機能を持っているもの（Evasive Malware）もあり，解析機能と解析回避機能のイタチごっこが起きている[4-1]．Evasive という単語（直訳：回避的な）は，「サンドボックスによる解析を回避する」という意味で使われており，プログラム言語等の盲点をついて道理に反した振る舞いを導く，いわゆる Evasive コードが厄介な存在となっている．本章では，巧妙な悪性 Web サイトで活用されている Evasive コードを特定する技術について説明する．

4.1. 研究の背景と課題

セキュリティソフトウェアの開発・販売を生業としているシマンテック社は，年間 100 万を超える悪性 Web サイトからのサイバー攻撃をブロックしているという[4-2]．それらのサイバー攻撃は自動化されており，「エクスプロイトキット」と呼ばれる，PC のアプリケーションやブラウザ等における脆弱性を利用してハッキングを行うためのツールが用いられ，マルウェアのダウンロードやインストールといった不正活動の展開がなされている．

悪性 Web サイトからのサイバー攻撃への対策として，シマンテック社等のセキュリティベンダが主に適用している手法は，ブラックリスト型のブロッキングである．つまり，悪性な URL やスクリプトおよびプログラム（マルウェアやその一種であるエクスプロイト）等のリストである「ブラックリスト」を如何に充実化させるかが，悪性 Web サイトからのサイバー攻撃のブロッキングにおいて肝要となる．

ブラックリストの充実化，つまり悪性な URL やスクリプトおよびプログラム等の収集は，ハニーポットにより行われる．ハニーポット（元来の意味は「蜜の詰まった壷」）は，何らかの有益そうな情報や資源がありそうな場所を，意図的に脆弱性を仕込んだうえで用意し，それにつられた者（サイバー攻撃）を観察（分析）したり，肝心な部分で被害を出さないために目をそらせたり，コンピュータ・フォレンジックを行うための証拠を集めたりする，一種のおとり手法に使われるもの（デコイ・システム）である．

ハニーポットには大きく 2 種類あり，サーバに置く（サイバー攻撃を待ち受ける）タイプのサーバ型と，Web ブラウザ等に機能配備される（悪性 Web サイトをクローリングする）タイプのクライアント型がある．クライアント型のハニーポットは「ハニークライアント」と呼ばれているが，本研究では，ブラックリストの充実化を主な目的とし，その実現を加速化すべく，能動的に（既知の悪性サイトを隈なくクローリングすることで新たな）ブラックリストのエントリーを書き集めることが可能なハニークライアントの高度化技術を追求する．

さらに，ハニークライアントには大きく 2 種類あり，実際の Web ブラウザを用いるタイプの「ハイ・インタラクション型」と，Web ブラウザをエミュレートするタイプの「ロー・インタラクション型」がある．ハイ・インタラクション型のハニークライアントには，実際の Web ブラウザを用いていることから，実行されたサイバー攻撃手法（の一部）を正確に追うことができるというメリットはあるが，当該手法におけるその他の攻撃バリエーションについて分析を行うことはできないというデメリットがある．一方，ロー・インタンラクション型のハニークライアントには，Web ブラウザをエミュレートしていることから，サイバー攻撃手法の攻撃バリエーションについて吟味できるというメリットはあるが，それぞれの攻撃についてひとつひとつのステップを精緻に分析することはできないというデメリットがある．このため，それぞれが相補的な関係にあるこれら 2 種類のハニークライアントを併用することで，悪性 Web サイトの分析ケーパビリティを向上させることができる．具体的には，ロー・イン

タラクション型のハニークライアントを用いて総括的な分析を行った後に，適宜，ハイ・インタラクション型のハニークライアントを用いてより高精度に分析を行うことで効率的なブラックリストの生成が可能となる．

ちなみに，NTT セキュアプラットフォーム研究所においては，ハイ・インタラクション型とロー・インタラクション型のそれぞれについて，世界トップクラスの高性能なハニークライアントを実現している[4-3][4-4]．本研究では，ロー・インタラクション型のハニークライアントによる解析を回避する Evasive コードの特定手法の高度化を狙う．

サイバー攻撃は Web ブラウザの脆弱性を突いてくるが，脆弱性は Web ブラウザの種類やバージョンによって区々である．すなわち，攻撃者は Web ブラウザの種類やバージョンを特定したうえで，当該（サイバー攻撃にとって恰好のターゲットとなる）脆弱性を保有する Web ブラウザに的を絞って，有効的な攻撃を選んで仕掛けてくる．Web ブラウザの種類やバージョンを特定する行為はブラウザ・フィンガープリンティングと呼ばれるが，ブラウザ・フィンガープリンティングに基づく攻撃のエクスプロイト・コードの例を図 4-1 に示す．

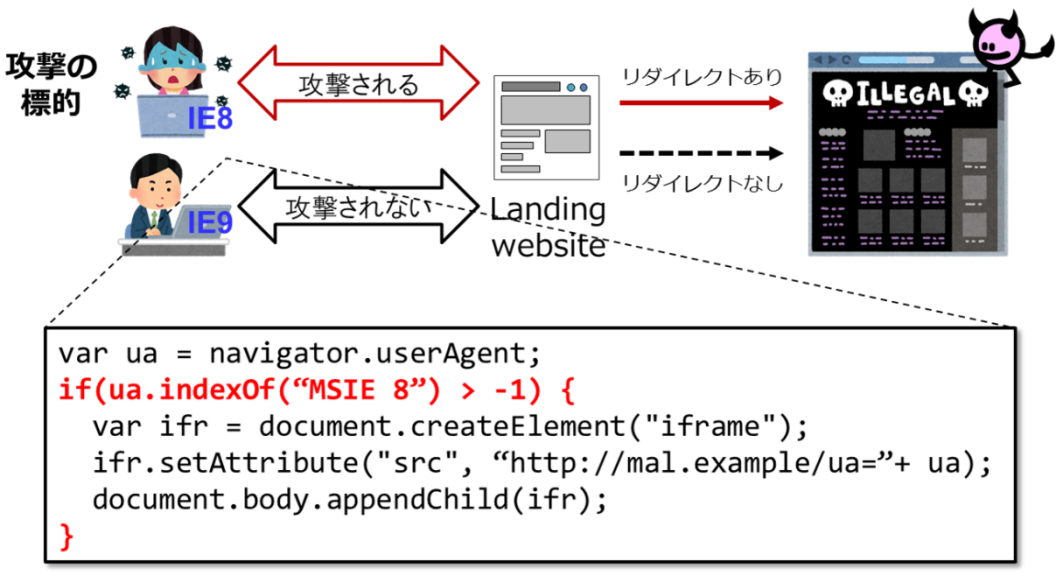


図 4-1. ブラウザ・フィンガープリンティングの悪用例

この例は，悪性 Web サイトにアクセスしたユーザの Web ブラウザが，インターネットエクスプローラーIE（マイクロソフト）のバージョン 8 であれば，悪意のある URL へ誘導（リダイレクト）される，というエクスプロイト・コードである．

Evasive コードとしてはこれまでに，ブラウザ・フィンガープリンティングにおける，Web ブラウザの種類やバージョンの違いによる挙動の違い（以下，ブラウザの癖と呼ぶことにする）を悪用するものが発見されているが，その例を図 4-2 に示す．

```
setTimeout(10);
url = "http://DOMAIN.ru/js/jquery.min.php";
document.write("<script type='text/javascript'
src='"+url+"'></script>");
```

図 4-2. ブラウザの癖に基づくエクスプロイト・コードの例

このエクスプロイト・コードでは，ブラウザの種類やバージョンによって，setTimeout()という関数の扱いが異なることを逆手にとって攻撃を仕掛けている．具体的には，setTimeout 関数の引数は実際，図 4-3 のように，関数またはコードスニペット（プログラミング言語の中で簡単に切り貼りして再利用できる部分またはパターン）と定義されているものの，図 4-2 のように数値（この場合，”10”）を直接代入しても，ブラウザによっては（より最近のブラウザのバージョンでは）実行できてしまう，という現象に基づいている．

つまり，比較的古いバージョンのブラウザでは，setTimeout(10)は実行できずに”Invalid Argument”エラーとなるが，ロー・インタラクション型のハニークライアントも，このようないわゆるブラウザの癖には対応できず，悪意のある URL を抽出できない，という結果に陥ってしまう．

```
var timeoutID = scope.setTimeout(function[, delay, param1, param2,
...]);
var timeoutID = scope.setTimeout(function[, delay]);
var timeoutID = scope.setTimeout(code[, delay]);
```

図 4-3. setTimeout 関数の引数定義

4.2. Evasive コードの特定手法

本研究では，ブラウザの癖に基づくエクスプロイト・コード，すなわち，ロー・インタラクション型のハニークライアントによる解析を回避する Evasive コードの特定手法を追究する．具体的には，以下に示す 3 段階のプロセスを経て Evasive コードの特定を行うこととした（図 4-4）．

① ： ハイ・インタラクション型とロー・インタラクション型のハニークライアントの，リダイレクション先の差異に基づく，Evasive コードが含まれると考えられる Java スクリプトの抽出．

② ： ①で抽出された大多数の Java スクリプトの，想定される Evasive テクニックの観点からの分類（クラシフィケーション）．

③ ： ②で分類された Java スクリプトの手動解析による，悪用された Evasive テクニックの特定．

図 4-4. Evasive コードの特定プロセス

前記① の「Evasive コードを含むと想定される Java スクリプト候補の抽出」にあたっては，まず，以下の 3 種のブラックリストを用いて，（既知の）悪性 Web サイトを，ハイ・インタラクション型（Web ブラウザとしては，マイクロソフトの IE6 と IE8 を使用）とロー・インタラクション型の 2 タイプのハニークライアントによってクローリングを行った.

・公開のブラックリスト

・シマンテック等から調達したブラックリスト

・NTT セキュアプラットフォーム研究所の独自ブラックリスト

2 タイプのハニークライアントのリダイレクション先の差異（リダイレクトされたか否かの違い）から，Evasive コードが含まれると考えられる Java スクリプトを抽出した．具体的には，図 4-5 に示すようにリダイレクショングラフを構築し，ロー・インタラクション型のハニークライアントはリダイレクトされなかったが，ハイ・インタラクション型のハニークライアントがリダイレクトされた悪性 URL に着目し，これにリダイレクションを実行した URL における Java スクリプトを抽出した.

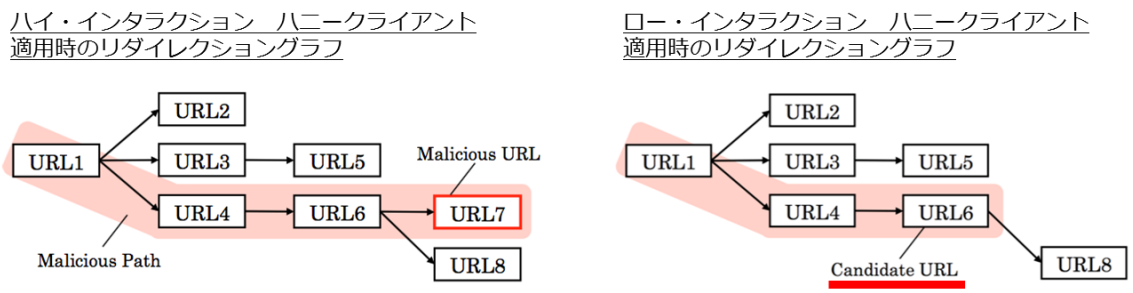


図 4-5. リダイレクショングラフによる Java スクリプトの抽出

前記② の「Evasive テクニックの観点での Java スクリプトのクラシフィケーション」にあたっては，まず，「Web ブラウザの癖を悪用に至る起点，すなわち Evasive コードの所以となるのは，プログラムにおける制御フローの分岐

点」という仮定を置き，図 4-6 に示すように，制御フローに変化をもたらすシーケンスに着目した．実際の当該シーケンスの抽出には，AST（Abstract Syntax Tree）分析の手法を適用した．ちなみに，AST とは，通常の構文木から，言語の意味に関係ない情報を取り除き，意味に関係ある情報のみを取り出した（抽象化した）ツリー状のデータ構造である．

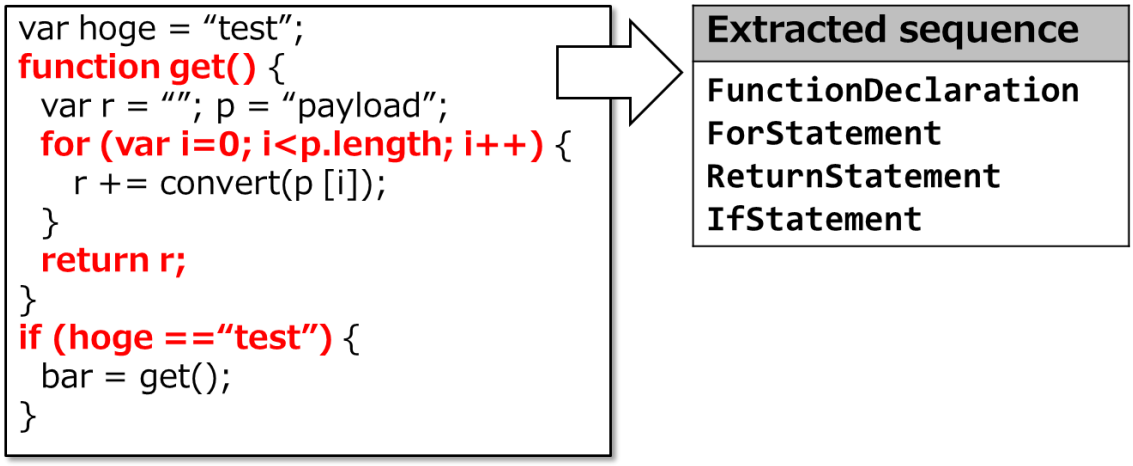


図 4-6. Evasive コードの所以たるシーケンスの抽出

次に，抽出したシーケンスについて，それらの類似度に基づくクラシフィケーションを行った．シーケンスの類似度には，LCS（Longest Common Subsequence），いわゆる最長共通部分列を適用し，以下の計算を行った．

シーケンス類似度＝len(LCS(S1,S2))/max(len(S1),len(S2))

シーケンスのクラシフィケーションには，DBSCAN（Density-Based Spatial Clustering of Applications with Noise）によるコードクラスタリングを適用した．ちなみに，DBSCAN：

「ある空間に点集合が与えられた時に，互いに密接してきっちり詰まっている点をグループにまとめ，低密度領域にある点(その最近接点が遠すぎる点)を外れ値とする手法」

は，科学文献の中でも数多く採用されており，最も一般的なクラスタリングアルゴリズムのひとつである．

前記③の「Evasive テクニックの特定に向けた Java スクリプトの手動解析」にあたっては，図 4-7 に示すような，IE，Chrome，Firefox といった各種ブラウザのデベロッパーツールを用いた．

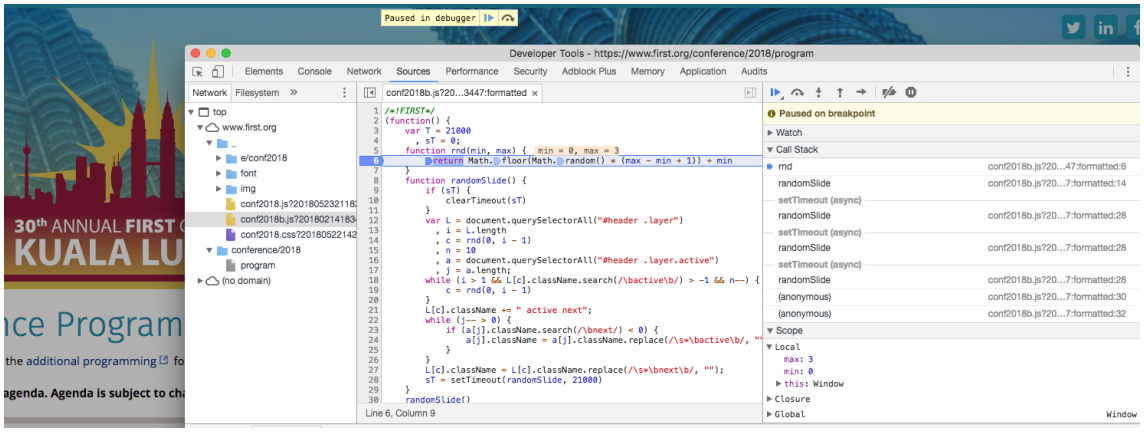


図 4-7. 市中ツールを用いた手動解析による Evasive テクニックの特定

4.3. 発見された Evasive コード

4.2 項で示した一連のプロセス（3 段階による Evasive コードの特定）で活用したデータセットは，NTT セキュアプラットフォーム研究所において，2012 年から 2016 年にわたって収集した HTTP トラフィック，合計 20,272 件である．NTT セキュアプラットフォーム研究所が開発した，ハイ・インタラクション型（Web ブラウザとしては，マイクロソフトの IE6 と IE8 を使用）とロー・インタラクション型の 2 タイプのハニークライアントによってクローリングを行った結果を，表 4-1 に示す．

表 4-1. 2 タイプのハニークライアントによるクローリング結果

HTTPトラヒック（ハニークライアント／URL判定）	アクセス数
合計	20,272
ハイ・インタラクション／良性	459
ロー・インタラクション／悪性	18,497
ハイ・インタラクション／悪性 かつ ロー・インタラクション／良性	**1,166**

リダイレクショングラフの構築によって，ロー・インタラクション型のハニークライアントはリダイレクトされなかったが，ハイ・インタラクション型のハニークライアントがリダイレクトされた HTTP トラフィックは，合計 1,166 件であったが，それらから抽出された，Evasive コードを含むと想定される Java スクリプト候補は，2,410 件であった．

Evasive テクニックの観点での Java スクリプトのクラシフィケーションでは，224 件については DBSCAN 等を通じてクラスタリングできなかったものの，合計 57 個のクラスターに分類できた．クラシフィケーションの詳細を，表 4-2 に示す．

表 4-2. Java スクリプト（シーケンス）の分類結果

クラスターについて	数	
悪性コードを伴うクラスター群	21	合計57
良性コードのクラスター群	36	
ノイズ（クラスタリング不可）	224	

さらに，市中ツールを用いた手動解析を進めた結果，分類されたクラスターのうち，Evasive コードを有するものは 21 個であり，それらは表 4-3 に示すように，5 つの Evasive テクニックに分類できた．

表 4-3. 特定された Evasive テクニック

Evasiveテクニック	Evasiveコード
Use of original object	window.sidebar
Difference in array processing	[“a”,”b”,].length
Difference in string processing	“ \v”==“v”
Difference in setTimeout() processing	setTimeout(10)
Difference in parseInt() processing	parseInt(“0123”)

4.4. ケーススタディ

4.3 項で分類された 5 つの Evasive テクニックのそれぞれに対し，実際の Web ブラウザの振る舞いを検証した．検証に使った Web ブラウザは 4 種類で，それぞれ，IE8，IE9，IE10 と Firefox（Version 50.0.2）である．

＜ケース 1 of 5＞：Firefox 特有のオブジェクト

図 4-8 に示すように，Firefox 以外は「0」をリターン値として返す，という結果であった．

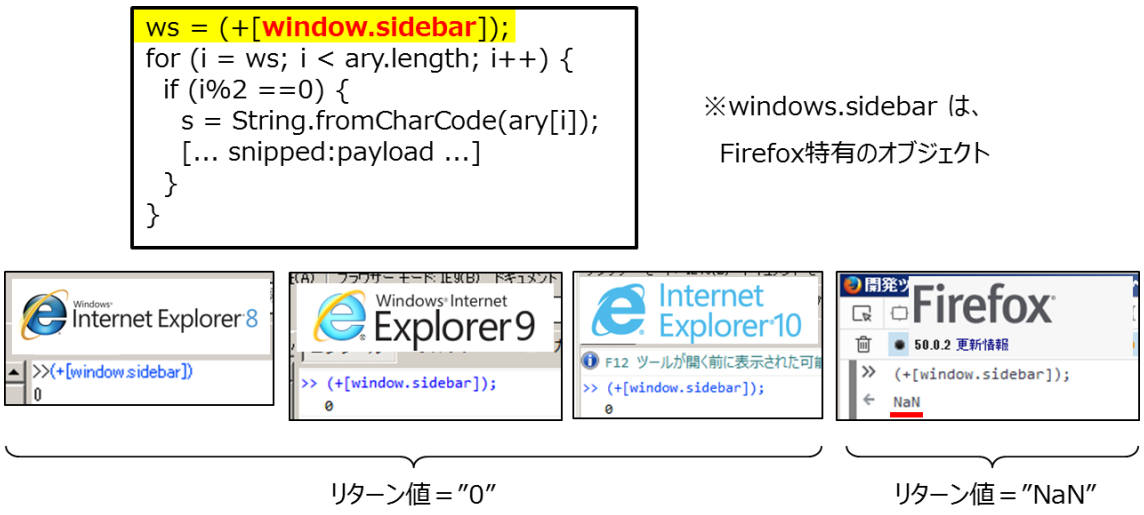


図 4-8.　「Firefox 特有のオブジェクト」に関する検証結果

＜ケース 2 of 5＞：配列における”,”の扱いの差異

図 4-9 に示すように，IE8 とそれ以外（IE9，IE10，Firefox）ではリターン値が異なる，という結果であった．

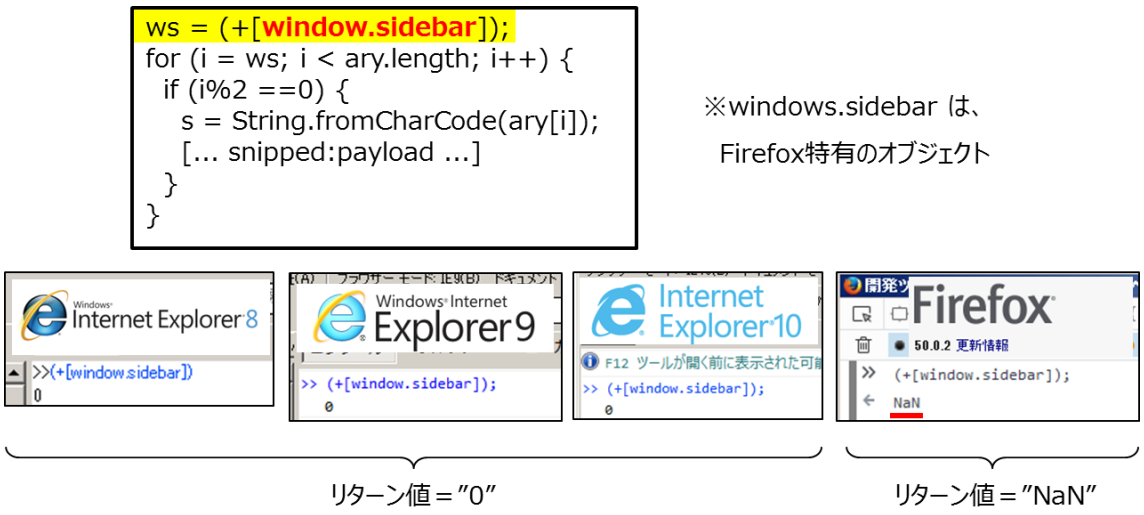


図 4-9.　「配列における”,”の扱いの差異」に関する検証結果

＜ケース 3 of 5＞：垂直タブの解釈

図 4-10 に示すように，バージョン 9 より以前の IE では垂直タブを単なるキャラクター「v」と解釈していた，という結果であった．

```
var t1 = "\v" == "v";
var t2 = document["all"];
var t3 = document["querySelector"];
var b7 = t1 && !t3 && t2;
var b8 = t1 && t2 && t3 && !t4;
var b9 = t2 && !t1 && t4;
t7 = t7 > 0 ? (b7 ? 1 : window["dummy"]) : 1;
t8 = t8 > 0 ? (b8 ? 1 : window["dummy"]) : 1;
t9 = t9 > 0 ? (b9 ? 1 : window["dummy"]) : 1;
[... snipped:redirect/exploit code ...]
```

※垂直タブの解釈が古いブラウザ（IE8）では異なる

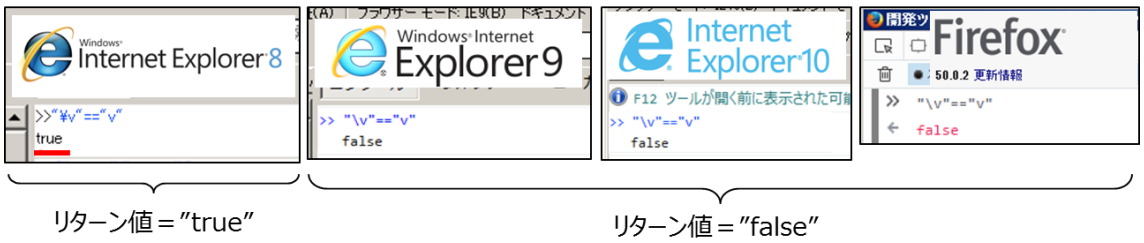


図 4-10. 「垂直タブの解釈」に関する検証結果

＜ケース 4 of 5＞：setTimeout 関数の引数の扱い

図 4-11 に示すように，比較的古いバージョンのブラウザ（IE8 や IE9）では setTimeout(10)は実行できずに”Invalid Argument”エラーとなる，という結果であった．

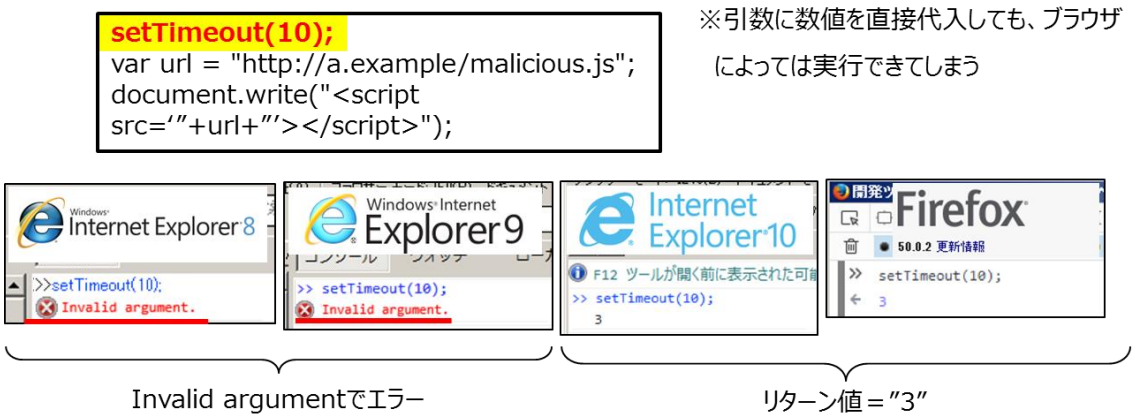


図 4-11.　「setTimeout 関数の引数の扱い」に関する検証結果

＜ケース 5 of 5＞：parseInt 関数の引数（数値）の扱い

図 4-12 に示すように，IE8 では，parseInt 関数の引数の数値は 8 進数として解釈されてしまう，という結果であった．

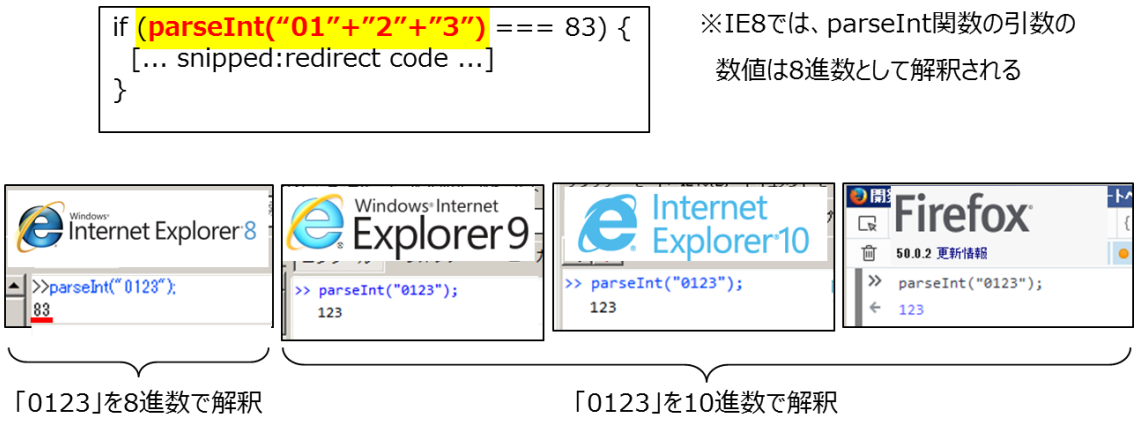


図 4-12.　「parseInt 関数の引数（数値）の扱い」に関する検証結果

4.5. まとめ

本章では，プログラム言語等の盲点をついて道理に反した振る舞いを導く，いわゆる Evasive コードを特定する技術について説明した．Evasive コードは，巧妙な悪性 Web サイトで活用されているが，本研究では特に，ロー・インタラクション型のハニークライアントによる解析を回避する Evasive コードの特定手法について提案している．具体的には，ハイ・インタラクション型とロー・インタラクション型のハニークライアントの，リダイレクション先の差異に基づく，Evasive コードが含まれると考えられる Java スクリプトの抽出を行い，それらを想定される Evasive テクニックの観点からの分類（クラシフィケーション）することで，5 種類の Evasive コードの特定に成功した．

図 4-13 は，それらのうち，「setTimeout 関数の引数の扱い」と「配列における”,”の扱いの差異」に関わる Evasive コードが近年のアタックキャンペーン[4-5][4-6]で悪用されていたことを示す．併せて，「parseInt 関数の引数（数値）の扱い」に関わる Evasive コードは時代遅れということも分かる．

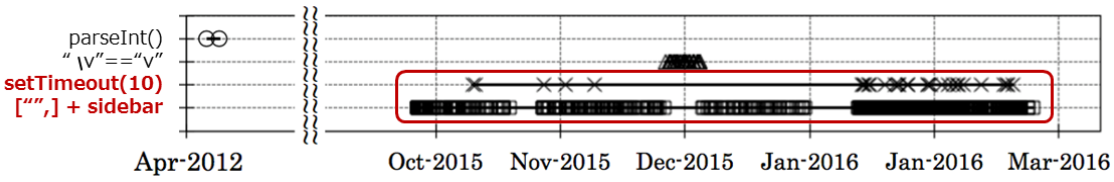


図 4-13. Evasive コードの時系列分析

第 5 章

64 ビット版 Windows のメモリダンプからのスタックトレース構築技術

さまざまなサイバー攻撃において広くマルウェアが使用されているが，昨今の「攻撃の成功」を大前提として対策を実施するうえでは，マルウェア感染に伴う挙動分析が重要になる．このための常套手段としてフォレンジックが有用であるが，中でもプログラム・デバッギングと同様，メモリフォレンジックが高度な分析技術として期待されている．本章では，メモリフォレンジックにおいて重要な役割を果たす，高度なスタックトレースの手法について述べる．

5.1. 研究の背景と課題

従来のスタックトレースにおいては，ベースポインタ（EBP）と呼ばれるいわゆる，「今実行中の関数が使用しているスタック領域の底」であるフレームポインタを活用していた．ところが，最近の Windows x64 オペレーティングシステム（64 ビット版 Windows）では，処理効率化等の理由から，コンパイラがフレームポインタを用いずに関数を生成するようになったことから，フレームポインタを活用したスタックトレースができない状況になっている．

フレームポインタを活用せずにスタックトレースを実施する従来手法としてはスキャンベースの手法があるものの，リターンアドレス（関数の処理終了後の次に実行すべきプログラムコードのアドレス）を誤認識するケースもあり，必ずしも正確なスタックトレースができないという大きな課題（以下では，メイン課題と呼ぶ）がある．

また，マルウェアのような通常のプログラミング作法を必ずしも守っていないであろうプログラムにおいては，そもそも所定のスタック領域を使わないも

のもあり，何らかの方法によりスキャン対象を探さなければならないという課題（以下では，サブ課題 1 と呼ぶ）がある．

さらに，Windows x64 オペレーティングシステムは，WOW64 と呼ばれる，32 ビット版 Windows ベースのアプリケーションを 64 ビット版 Windows でシームレスに実行できるようにする x86 エミュレータを備えている．このような 32 ビット版 Windows ベースのアプリケーションについては，カーネルから直接スタック領域に関する情報が得られないため，スタックトレースができないという課題（以下では，サブ課題 2 と呼ぶ）もある．

5.2. 課題解決のアプローチ（俯瞰）

本項では，前述の課題に対する解決策の流れ（図 5-1 参照）を述べる．

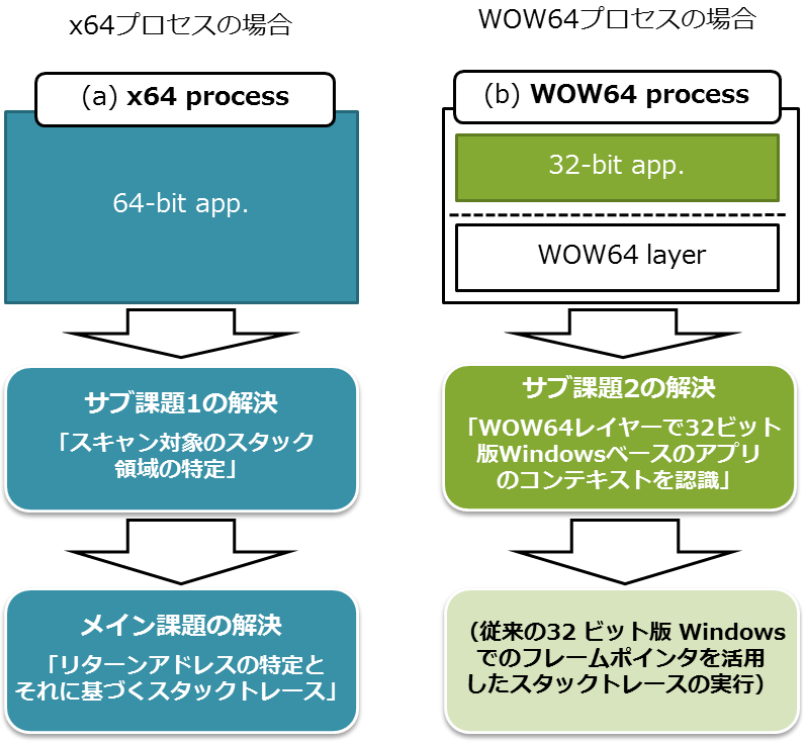


図 5-1. 提案する解決策の実行フロー

まず，メモリフォレンジックの対象とするプロセス毎に解決策を提案する．すなわち，64 ビット版 Windows ベースのプロセスと 32 ビット版 Windows ベースのプロセスの 2 つに分けて解決策を提案する．

そのうえで，64 ビット版 Windows ベースのプロセスのスタックトレースについては，サブ課題 1 の解決，すなわち「スキャン対象のスタック領域の特定」を行い，メイン課題の解決，すなわち「リターンアドレスの特定とそれに基づくスタックトレース」を図る．前者については 5.3 項で，後者については 5.4 項で述べる．5.4 項で詳細を言及するが，リターンアドレスの特定に，例外処理のために用意されているメタデータ（スタック情報）が使える場合と使えない場合があり，それぞれについて対策を述べる．

32 ビット版 Windows ベースのプロセスのスタックトレースについては，サブ課題 2 の解決，すなわち「WOW64 レイヤーで 32 ビット版 Windows ベースのアプリケーションのコンテキスト（一連の処理中で引き継いでいく情報の集合体）の認識」を図る．これが解決できれば，Windows x86 オペレーティングシステム（32 ビット版 Windows）でのフレームポインタを活用したスタックトレースが実行できる．サブ課題 2 の解決については，5.5 項で詳細を言及する．

5.3. Windows x64 におけるスタック領域の特定

本項では，サブ課題 1 の解決，すなわち「スキャン対象のスタック領域の特定」について述べる．

OS は一般に，スレッドのコンテキストを，システムコール等のイベントが発生したタイミングでメモリーにストアしている．Windows においては，ETHREAD と呼ばれるスレッド管理のための構造体が用意されているがその中で，「トラ

ップフレーム（TrapFrame)」というメンバは，イベント発生時点でのレジスタの値が保存されたスタックの内容である．このトラップフレームから実際に使われたスタックが分かるため，これを使うことがサブ課題 1 を解決できる．具体的には，スタックのトップのメモリアドレスである RSP を用いてスタックトレースを進めることができる（図 5-2).

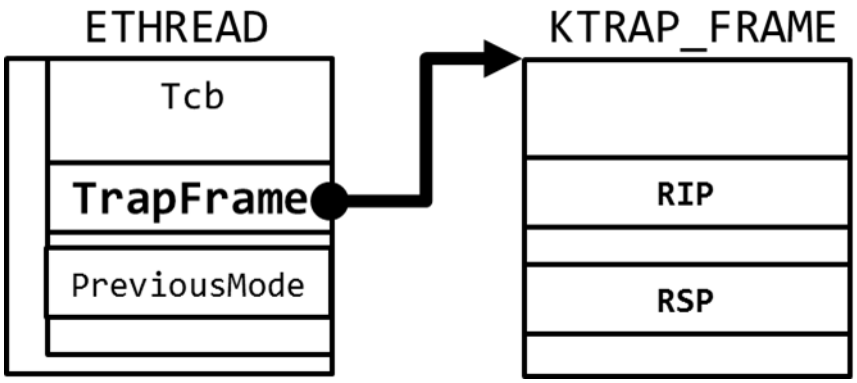


図 5-2. ETHREAD 及びトラップフレーム

5.4. Windows x64 におけるリターンアドレスの特定

本項では，メイン課題の解決，すなわち「リターンアドレスの特定とそれに基づくスタックトレース」について述べる．リターンアドレスの特定においては，例外処理のために用意されているメタデータ（スタック情報）が使える場合と使えない場合があり，以下ではそれぞれについて対策を述べる.

5.4.1. 例外処理用のメタデータが使える場合の特定法

PE（Portable Executable）は，主に 32 ビット版及び 64 ビット版の Windows 上で使用される実行ファイルのファイルフォーマットであるが，64 ビット版の

Windows の実行ファイルは，図 5-3 に示されるような例外処理用のメタデータを有し，これをメモリフォレンジックにおいて活用する．

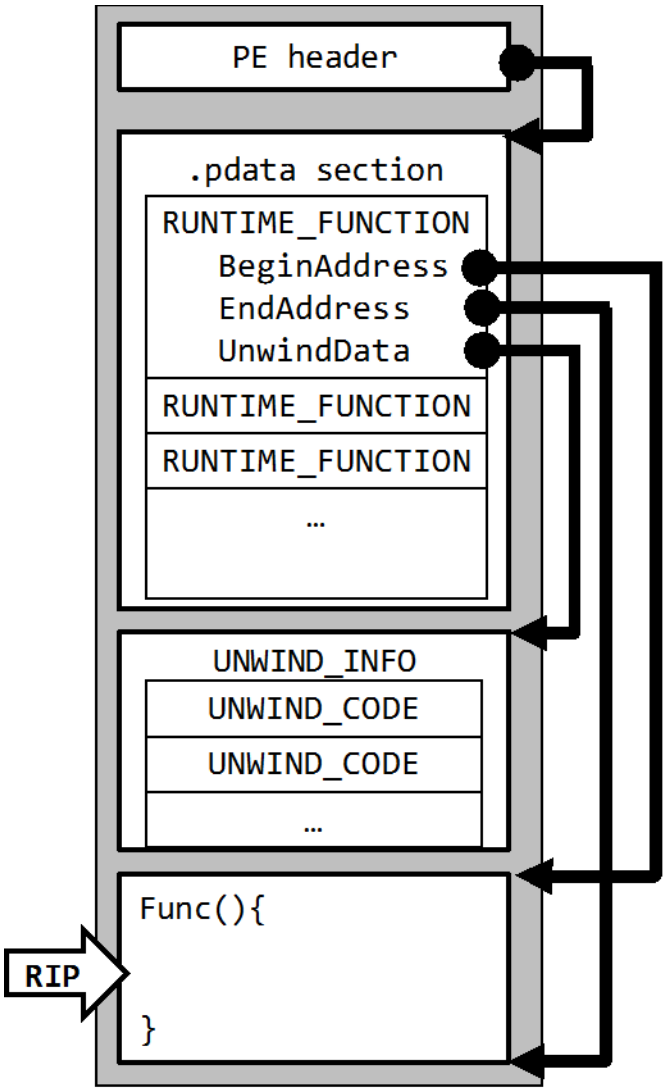


図 5-3. 実行ファイル（Windows x64）のメタデータ

例外処理用のメタデータを使ってスタック・アンワインドをエミュレートし，リターンアドレスを取得する手順の概要を以下に示す．

① VADs（仮想アドレス・デスクリプターズ）のプロセスツリーを参照して，RIP（インストラクションポインタと呼ばれる 64 ビットレジスタ）によって示される領域のベースアドレス（スタックの初期値）を取得後，PE ヘッダーのシグナチャを確認する[5-1].

② .pdata セクションに配置される RUNTIME_FUNCTION 構造体を取得する.

#RUNTIME_FUNCTION 構造体は，スタック領域の割り当てや呼び出す関数に関わるテーブルエントリからなり，具体的には「関数の開始アドレス」「関数の終了アドレス」「アンワインド情報のアドレス」の形式を取る.

③ RUNTIME_FUNCTION 構造体[5-2]の「アンワインド情報のアドレス」によって示される UNWIND_INFO 構造体に含まれるそれぞれの UNWIND_CODE を手掛かりに RSP をアンワインドし，各種レジスタをリストアする.

④ アンワインド情報が他のアンワインド情報にチェーンしている場合は，チェーンの最後まで，前述の処理②および③を繰り返す.

⑤ 全てのアンワインド処理が終了した時点で（RSP の値は RSP’），リターンアドレスをスタックから取り出し，RIP の値にそのリターンアドレスをセットする（RSP の値は RSP”）.

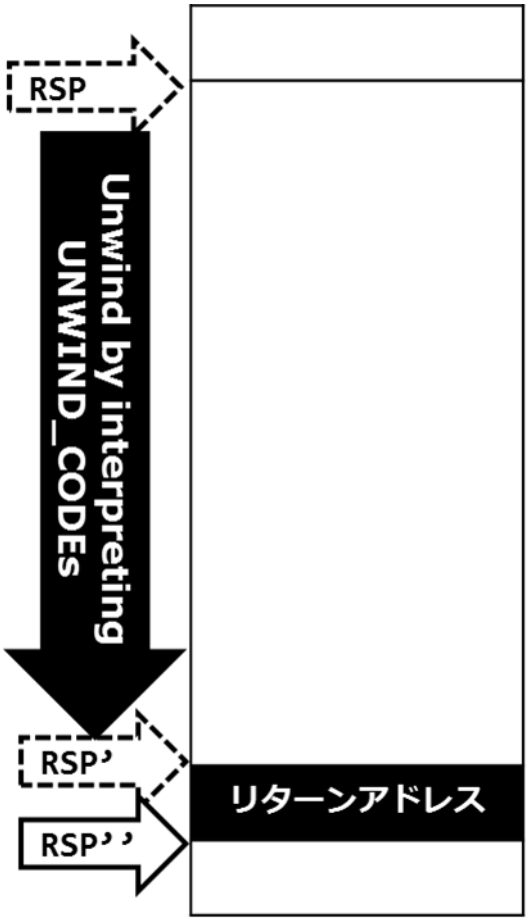


図 5-4. スタック・リワインディングのエミュレーション

5.4.2. 例外処理用のメタデータが使えない場合の特定法

前述の例外処理用のメタデータが使えない場合には，従来のスキャンベースの手法を拡張することで，リターンアドレスの特定を行う．

具体的には，スキャンする過程でリターンアドレスの候補が，関数の処理終了後に次に実行すべきプログラムコードのアドレスかどうか確かめるために，当該プログラムコードを逆アセンブルし，制御フロー分析を行うことで検証を行う．

例えば，図 5-5 のように，スキャンによりリターンアドレスの候補すなわち，ポインタ 1 とポインタ 2 が見つかったとする．それぞれのポインタが指す先のの直前が call 命令であり，それによって呼び出される関数（プログラムコード）について制御フロー分析を行う．関数の実行パスにおいて，現在の RIP に到達する部分があれば，該当する call 命令の直後を指すポインタが正しいリターンアドレスであると言える．すなわち，図 5-5 においては，ポインタ 2 が正しいリターンアドレスと認識される．

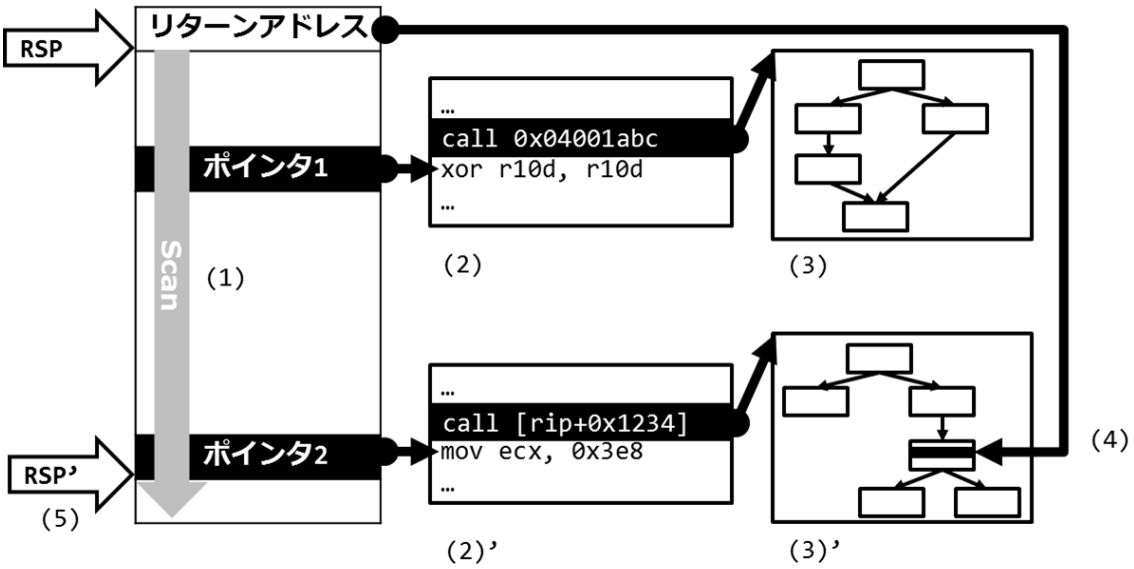


図 5-5. 制御フロー分析に基づくリターンアドレスの検証

5.5. WOW64 レイヤーでのスタックトレース

WOW64 レイヤーで 32 ビット版 Windows ベースのアプリケーションのコンテキスト（一連の処理中で引き継いでいく情報の集合体）の認識ができれば，Windows x86 オペレーティングシステム（32 ビット版 Windows）でのフレームポインタを活用したスタックトレースが実行できる．

実際，32 ビット版 Windows ベースのアプリケーションがシステムコールを呼び出す際に，WOW64 レイヤーはそのシステムコールをエミュレートするが，これは OS のシステムコールの処理とよく似ているものである．

図 5-6 に示すように，WOW64 レイヤーで保持されるユーザコンテキストは，ETHREAD と呼ばれるスレッド管理のための構造体から辿ることができる．具体的には，32 ビット版 Windows ベースのアプリケーションのコンテキストは，WOW64_CONTEXT として，WOW64 レイヤーのスレッドローカルストレージ（TLS）に格納されている．

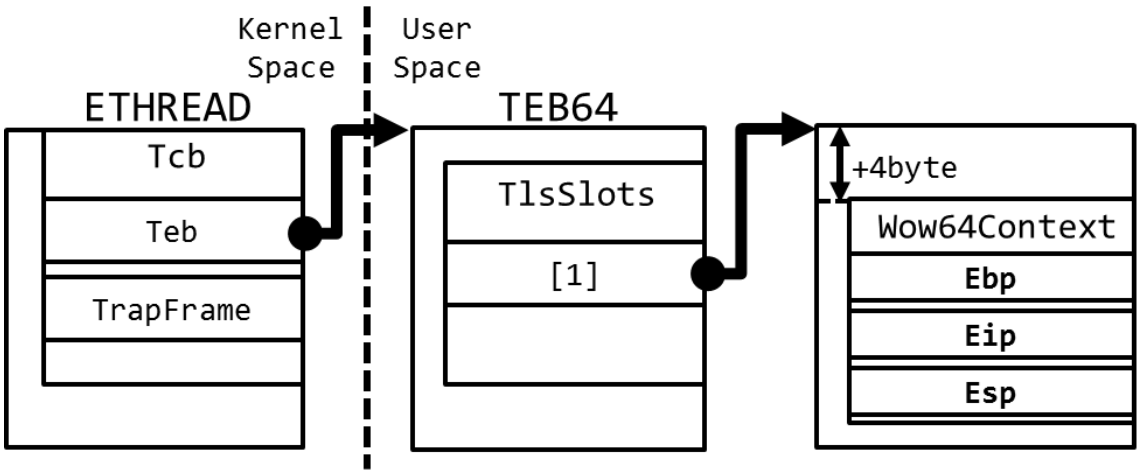


図 5-6. ETHREAD における WOW64_CONTEXT

5.6. 提案手法の評価

高度なフォレンジック及びインシデント・レスポンスのためのフレームワークである Rekall[5-3]のプラグインとして本章で提案した手法を実装し，Microsoft がフリーで提供するカーネルデバッガである WinDbg との比較評価を行った．

その際，KVM（Kernel-based Virtual Machine）上の VM（8 ギガバイトメモリ，Windows7 x64 SP1）から得られるメモリダンプを用いて評価を行った．

具体的には，例外処理用のメタデータを有する notepad.exe という名前の x64 プロセスのユーザスペースにおいてスタックトレースを実施した．これは，5.4.1 節で述べた「例外処理用のメタデータが使える場合の特定法」の検証であり，図 5-7 にその結果を示す．Child-SP は（その行で）呼び出された関数で使われる RSP の値であり，RetAddr がその関数のリターンアドレスを表す．この結果から分かるように，WinDbg で得られたものと提案手法を実装したプラグインから得られたものは同じであることから，提案手法が正確にスタックトレースを実行できたと言える．

WinDbg

```
   Child-SP          RetAddr           Call Site
   (snip)
0  fffff880`023eec20 00000000`77719e6a nt!KiSystemServiceCopyEnd+0x13 (TrapFrame @ fffff880`023eec20)
1  00000000`0016fcf8 00000000`77719e9e USER32!ZwUserGetMessage+0xa
2  00000000`0016fd00 00000000`ff561064 USER32!GetMessageW+0x34
3  00000000`0016fd30 00000000`ff56133c notepad!WinMain+0x182
4  00000000`0016fdb0 00000000`775f59cd notepad!DisplayNonGenuineDlgWorker+0x2da
5  00000000`0016fe70 00000000`7782a561 kernel32!BaseThreadInitThunk+0xd
6  00000000`0016fea0 00000000`00000000 ntdll!RtlUserThreadStart+0x1d
```

実装Plugin

```
   Child-SP          RetAddr           Call Site
a  0x000000000000    0x000077719e6a    - (TrapFrame @ 0xf880023eec20)
b  0x00000016fcf8    0x000077719e9e    user32!NtUserGetMessage+0xa
c  0x00000016fd00    0x0000ff561064    user32!GetMessageW+0x2a
d  0x00000016fd30    0x0000ff56133c    notepad+0x1064
e  0x00000016fdb0    0x0000775f59cd    notepad+0x133c
f  0x00000016fe70    0x00007782a561    kernel32!BaseThreadInitThunk+0xd
g  0x00000016fea0    0x000000000000    ntdll!RtlUserThreadStart+0x21
```

図 5-7. スタックトレース実行の比較結果（例外処理用のメタデータを有する x64 プロセス notepad.exe）

同様に，calc.exe という名前の WOW64 プロセスについてもスタックトレースを実施し，比較を行った．その結果，図 5-8 に示されるように，WinDbg で得られたものと提案手法を実装したプラグインから得られたものは同じであることから，提案手法が正確にスタックトレースを実行できたと言える．

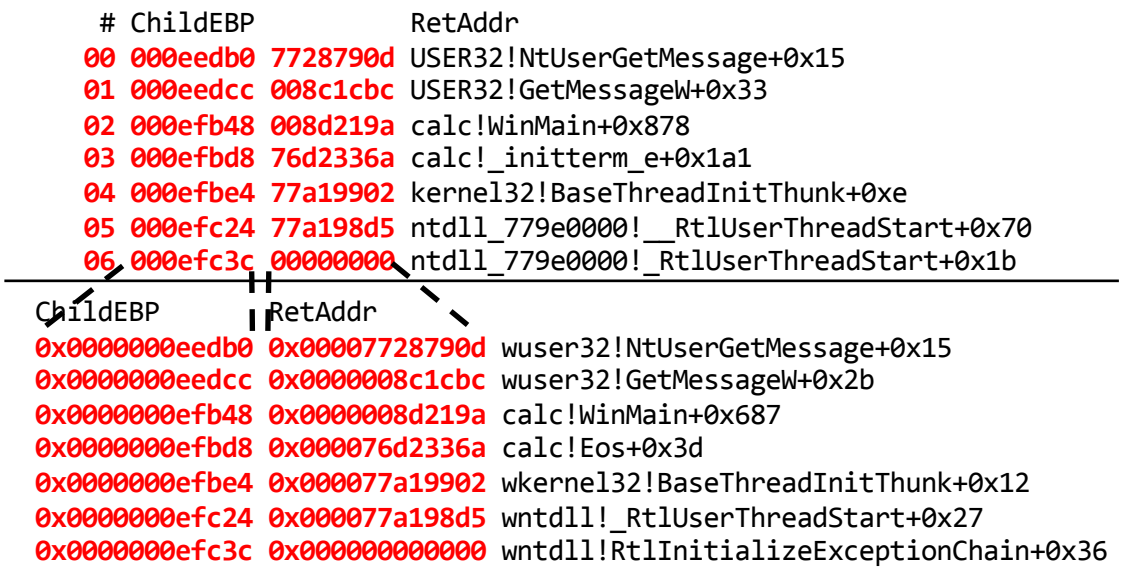


図 5-8. スタックトレース実行の比較結果（例外処理用のメタデータを有する WOW64 プロセス calc.exe）

次に，例外処理用のメタデータを使わない形で，前述の notepad.exe という名前の x64 プロセスについてスタックトレースを実施した．これは，5.4.2 節で述べた「例外処理用のメタデータが使えない場合の特定法」の検証であり，その結果が前述の「例外処理用のメタデータが使える場合の特定法」の検証結果（図 5-7）と提案手法を実装したプラグインから得られたものは同じであったことから，提案手法が正確にスタックトレースを実行できたと言える．

一方，例外処理用のメタデータを使わない形で同様に notepad.exe という名前の x64 プロセスについて，WinDbg を用いてスタックトレースした結果を図 5-9 に示す．

```
Child-SP          RetAddr           Call Site
(snip)
fffff880`023eec20 00000000`77719e6a nt!KiSystemServiceCopyEnd+0x13 (TrapFrame @ (snip))
00000000`0016fcf8 00000000`77719e9e 0x77719e6a
00000000`0016fd00 00000000`00000000 0x77719e9e
```

図 5-9. 例外処理用のメタデータを使わない形での WinDbg によるスタックトレース結果（x64 プロセス notepad.exe）

結果に示されるとおり，WinDbg によって得られたのは，最初の 2 つのエントリーだけであった．

また，例外処理用のメタデータを使わない形で同様に explorer.exe という名前の x64 プロセスについて，従来のスキャンベースの手法を用いてスタックトレースした結果を図 5-10 に示す．

```
SP              RetAddr
0x000007a4f708: 0x07fefd9b1430 (kernelbase!RtlAnsiStringToUnicodeString+0x34), (snip)
0x000007a4f728: 0x07fefe6cd8a2 (ole32!DcomChannelSetHResult+0x3066), (snip)
0x000007a4f7d8: 0x07fefe6c93fa (ole32!CoSetState+0x35a), (snip)
0x000007a4f808: 0x0000776016e3 (kernel32!WaitForMultipleObjectsExImplementation+0xb3), (snip)
0x000007a4f868: 0x07fefeb34981 (shell32!SHCreateDirectoryExW+0xe99), (snip)
0x000007a4f898: 0x000077718f7d (user32!RealMsgWaitForMultipleObjectsEx+0xfd), (snip)
0x000007a4f908: 0x07fefeb34981 (shell32!SHCreateDirectoryExW+0xe99), (snip)
0x000007a4f938: 0x0000777162b2 (user32!MsgWaitForMultipleObjectsEx+0x2e), (snip)
0x000007a4f968: 0x07fefeafe6ca (shell32!SHGetSetSettings+0x1536), (snip)
0x000007a4f978: 0x0000777162e0 (user32!MsgWaitForMultipleObjects+0x20), (snip)
0x000007a4f998: 0x07fefd9b133c (kernelbase!SetEvent+0xc), (snip)
0x000007a4f9b8: 0x07fefeb32ad9 (shell32!SHGetPropertyStoreForWindow+0x1459), (snip)
0x000007a4f9c8: 0x07fefeb2f9ff (shell32!ILIsEqual+0x16f), (snip)
0x000007a4f9f8: 0x07fefe6c9b49 (ole32!CoSetState+0xaa9), (snip)
0x000007a4fa38: 0x07fefeb32d70 (shell32!SHGetPropertyStoreForWindow+0x16f0), (snip)
0x000007a4fab8: 0x07fefeb32cd0 (shell32!SHGetPropertyStoreForWindow+0x1650), (snip)
0x000007a4fae8: 0x07fefe6c2c1d (ole32!CoInitializeEx+0x1ed), (snip)
0x000007a4fb58: 0x07fefeb32de2 (shell32!SHGetPropertyStoreForWindow+0x1762), (snip)
0x000007a4fb88: 0x07fefe033843 (shlwapi!IUnknown_GetWindow+0x68f), (snip)
0x000007a4fbb8: 0x00007782004b (ntdll!RtlpTpWorkCallback+0x16b), (snip)
0x000007a4fc48: 0x000077820012 (ntdll!RtlpTpWorkCallback+0x132), (snip)
0x000007a4fc98: 0x00007781fc62 (ntdll!TpPostTask+0x2e2), (snip)
0x000007a4ff28: 0x0000775f59cd (kernel32!BaseThreadInitThunk+0xd), (snip)
0x000007a4ff58: 0x00007782a561 (ntdll!RtlUserThreadStart+0x21), (snip)
```

図 5-10. 例外処理用のメタデータを使わない形での従来のスキャンベースの手法によるスタックトレース結果（x64 プロセス explorer.exe）

この結果に示されるとおり，従来のスキャンベースの手法によって得られたものには，図中の下線で示している 10 個の誤検知（false positive）が発生した.

5.7. 考察

前項までの結果から分かるとおり，提案した手法は基本的には効果的であると考えるものの，以下に列挙するような，スタックトレースの実施がより複雑でかつ難しくなるケースがあると，実際のマルウェア（Dridex，Zeus P2P，Tinba）の解析を通じて（それぞれ下記①～③のように）考察している.

① アンワインド情報が壊れている，または invalid な PE ヘッダーである場合
② 例えば，UMS（User Mode Scheduler）のような，カーネル側でアプリケーションの実行状態を管理できない場合
③ スレッドを用いずに，API フッキング等のイベントに対してウェイト状態となるような場合

このようなケースを想定し，提案手法をさらに改善するとともに，別のアプローチを指向する必要があると考えている.

また，Windows x64 におけるリターンアドレスの特定について，5.4.2 節で説明した「例外処理用のメタデータが使えない場合の特定法」を以てしても，リターンアドレスの候補をひとつに絞り込めない場合，例えば，脆弱性攻撃等によってインストラクションポインタの制御を奪われ，任意のコードが実行される間接コールのようなケースがある．この問題については，スタックとプログラムコードに対してさらに深い分析を検討する必要があると考えられる.

本研究では，64 ビット版 Windows ベースのプロセスのスタックトレースの課題を，Windows x64 のメモリダンプを使って解決する手法について論じたが，この手法の基本コンセプトは他のプラットフォームにも適用できる．実

際，64 ビット版 Linux や，Windows x86 を対象とした実装も研究されている [5-4].

5.8. まとめ

本章では，Windows x64 のメモリダンプを使ってスレッドのスタックトレースを行うために，スタック・アンワインディングのエミュレーションを適用する手法について提案した．特に，例外処理用のメタデータが使えない場合においては，プログラムコードの制御フロー分析を加味することで，従来のスキャンベースのものより正確にリターンアドレスが特定できる方式を提案した．また，実験による評価を通じて，提案した手法によりスタックトレースが正確にできかつ，実用的であることを示すことができた．

第 6 章

結言

本章を，本論文のまとめとするとともに，AI が人知を超えると言われる Singularity の時代（2045 年問題）に向けて，現在進化を遂げつつある AI のセキュリティに関わる現状の取り組みも加えたうえで，今後の研究について展望を示す.

6.1. まとめ

本稿では先ず，初のコンピューター・ウィルスの出現から現在までにおける環境とセキュリティ上の課題の変遷をレビューしつつ，昨今の具現化する脅威と必要なセキュリティ対策技術について網羅的にまとめた．そのうえで先ず，インターネットがブレイクする以前に期待が高まっていた，双方向マルチメディアサービスについて，特に当時のセキュリティにおける可用性重視の観点から，システムおよびネットワークの設計手法について提案した．次に，永遠のイタチごっこの如く，技術の更なる高度化が持続的に求められる IT セキュリティの領域にフォーカスし，開発が嘱望される 3 つのサイバー攻撃対策技術（情報セキュリティ確保のための技術）について提案した．具体的な成果はそれぞれ，以下の通りである.

① 双方向マルチメディアサービスのためのシステム設計法について，特に可用性の観点から，検討を行った．その結果，(1) 加入者側伝送リンクの利用率向上を図る共用端末数の設定，(2) ユーザに不快感を与えない音声通信アプリケーション起動数の設定，(3) 実トラヒック見合いの必要 VC 数算出とシステム構成に影響を与えない VC 収容を実現した.
② SIEM 等が取り扱うログ分析において，マルウェア感染端末の検知技術の高度化について検討を行った．その結果，データ圧縮アルゴリズム LZT を用いた特徴抽出によって，FQDN 部と Path 部の悪性・良性圧縮率を抽出し，L2

正則化 SVM で学習・推論することで，従来手法を上回る優れた判定精度の検知技術を確立した.

③ プログラム言語等の盲点をついて道理に反した振る舞いを導く，Evasive コードを特定する技術について検討を行った．その結果，ハイ・インタラクション型とロー・インタラクション型のハニークライアントの，リダイレクション先の差異に基づく，Evasive コードが含まれると考えられる Java スクリプトの抽出を行い，それらを想定される Evasive テクニックの観点からの分類することで，5 種類の Evasive コードを特定した.

④ Windows x64 のメモリダンプを使ってスレッドのスタックトレースを行うために，スタック・アンワインディングのエミュレーションを適用する手法について検討した．その結果，例外処理用のメタデータが使えない場合において，プログラムコードの制御フロー分析を加味することで，従来のスキャンベースのものより正確にリターンアドレスが特定できる方式を確立した.

6.2. Singularity (2045 年問題)におけるセキュリティ

Singularity は，カーツワイルの収穫加速の法則とも言われており，2045 年には千ドルのコンピュータの演算能力がおよそ 10 ペタ FLOPS＝人間の脳の 100 億倍にもなり，技術的特異点に至る AI の土台が十分に生まれているだろうとされている．すでに現在においても，AI で自動化されたサイバー攻撃が出現しつつあることから将来，AI による高度なハッキング等が主流になる可能性があり，防御側も AI 等を活用した自動化技術が必須になる．またその際，機械学習への攻撃として，誤認識を誘うような入力の作成，分類器の学習データの汚染，分類器へのクエリから分類器自体を盗む等の攻撃も問題として指摘されている[6-1].

これに対し，バイナリから自動で脆弱性を発見する技術（図 6-1）や，シンボリック実行等を活用して攻撃発動条件を自動抽出する技術（図 6-2）といっ

た要素技術の確立が急務であり，そのうえで，機械学習への攻撃への対策も併せて検討しつつ，自動防御によるサイバー攻撃の無効化を実現していく必要がある．

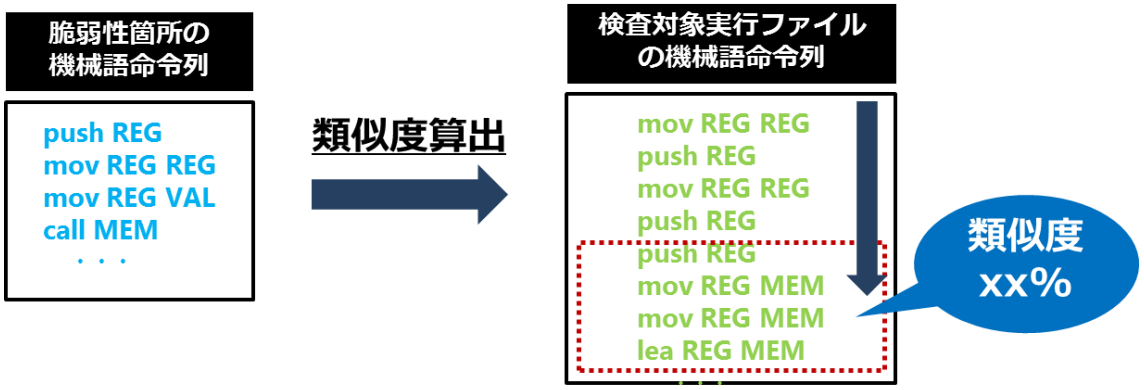


図 6-1. AI ハッキング関連技術　（脆弱ポイント特定）

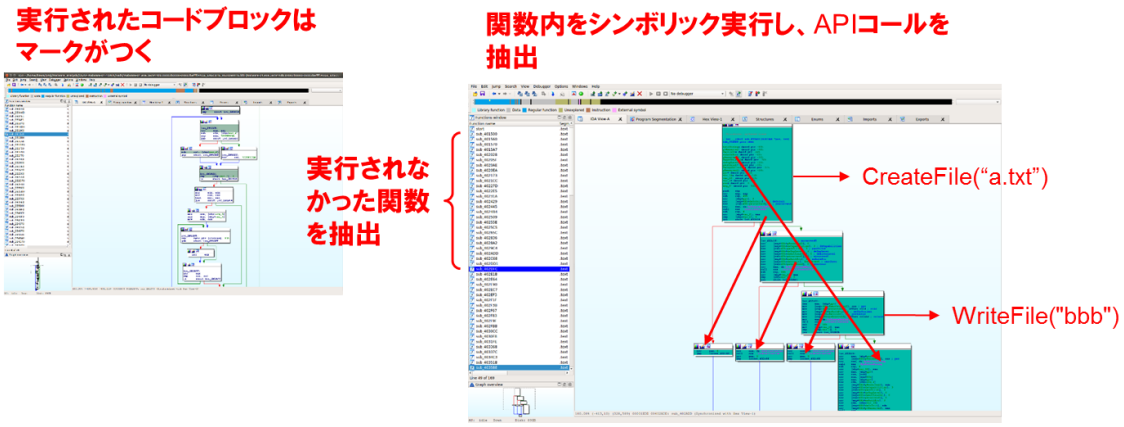


図 6-2. AI ハッキング関連技術　（攻撃プロセスの特定）

さらに，AI の技術的発展により，人間の思考や行動，社会構造の前提が大きく変化・多様化するため，社会規範として機能する法制度の研究も肝要になる．AI を開発から社会実装へスムーズに導くためには，

① AI 適用を前提に人間，社会，産業に与え得る影響とその事実関係を想定
② 現行法を AI の知見に照らし，個別の論点分析
③ 将来の立法政策に向けて AI 時代に対応する法制度を提言

といったステップで取り組みを進める必要があり，国内でも関連動向がいくつか見受けられ[6-2]，理化学研究所・革新知能統合研究センタ（AIP)[6-3]と NTT セキュアプラットフォーム研究所においても，関連共同研究を鋭意展開中である.

このように引き続き，技術と運用そして，法制度といった多面的な視点からセキュリティ課題を解決していく，すなわちセキュリティリスクを低減していくことが肝要であり，永遠に続くいたちごっこの世界を果敢に挑んでいく姿勢が人類にとって不可欠になるであろう.

謝辞

本論文をまとめるにあたり，愛知工業大学・情報科学部　河辺義信教授には，懇切丁寧なご指導，ご鞭撻を賜り，また本論文の審査委員も務めていただきました．ここに謹んで，感謝の意を表します．また，愛知工業大学・情報科学部　中條直也教授および小野木克明教授，愛知工業大学・経営学部　石井成美教授，愛知工業大学・情報科学部　水野忠則教授，には多大なる有益なご指導，ご助言をいただきました．あわせて，心より感謝申し上げます．

本研究は，日本電信電話株式会社・NTT 情報通信研究所ならびに，NTT セキュアプラットフォーム研究所において行われたものであり，当時，本機会を与えて下さった，矢野厚顧問（現住友電気工業株式会社，元 NTT マルチメディアビジネス開発部・担当部長），宮部博史博士（元 NTT 情報通信研究所・グループリーダー）に厚く御礼申し上げます．

また，本研究を進めるにあたり，常日頃からご指導いただいた，徳永裕史博士（元 NTT 情報通信研究所・グループリーダ），千葉工業大学・社会システム科学部学部　谷本茂明教授，法政大学・理工学部　金井敦教授，静岡大学・創造科学技術大学院　西垣正勝教授，NTT セキュアプラットフォーム研究所の関係者のみなさま，に心より感謝いたします．特に，千葉県浦安市での双方向マルチメディア実験においてシステム開発に携わった当時の，米国 SGI 社（Silicon Graphics Incorporation）のみなさま，ならびに NTT ソフトウェア（株）のみなさま，NTT マルチメディアビジネス開発部・第四プロジェクトのみなさまにもあわせて感謝申し上げます．

これまで本論文執筆に向けて，常日頃から研究遂行に関しましてご助言をいただき，また時には励まし続けていただきました，井上友二顧問（現株式会社トヨタ IT 開発センター）に厚く御礼申し上げます．

最後に，日頃よりあらゆる面で支えてくれている妻　悦子をはじめ，家族の全員に深く感謝いたします.

参考文献

[1-1] 初のコンピューター・ウィルス誕生

https://www.nic.ad.jp/timeline/

[1-2] セキュリティを意識した IoT デバイス設計の勘所：サイバー空間の脅威の変遷とその対策から IoT セキュリティを学ぶ, 2017 年 02 月 09 日, 森本純（トレンドマイクロ）

http://monoist.atmarkit.co.jp/mn/articles/1702/09/news004.html

[1-3] 情報セキュリティの 3 要素

https://www.jnsa.org/ikusei/01/02-01.html

[1-4] セキュリティ・バイ・デザイン入門；看板倒れでない設計段階のセキュリティ対策とは, 2016 年 11 月, 独立行政法人情報処理推進機構（ＩＰＡ）, 技術本部 ソフトウェア高信頼化センター（ＳＥＣ）, 金子朋子

https://www.ipa.go.jp/files/000055823.pdf

[1-5] 国立研究開発法人情報通信研究機構法の一部改正案について

http://www.soumu.go.jp/main_content/000536856.pdf

[1-6] 制御システムの安心・安全な運用を実現するサイバーセキュリティ技術「InteRSePT®」の販売を開始

http://www.ntt.co.jp/news2018/1804/180425b.html

[1-7] 安全な重要インフラを実現するアーキテクチャ（2017 年 2 月）

http://www.ntt.co.jp/journal/1702/files/jn20170210.pdf

[1-8] 戦略的イノベーション創造プログラム（SIP）／重要インフラ等におけるサイバーセキュリティの確保

http://www.nedo.go.jp/activities/ZZJP_100109.html

[2-1] 「第 6 章 マルチメディア技術の応用」of マルチメディア, 早稲田大学人間科学部 通信教育課程

http://www.f.waseda.jp/kane/sp_multi/

[2-2] 「マルチメディア通信の共同利用実験最終報告書」の発行について

http://www.ntt.co.jp/news/news97/970530b.html

[2-3] ITU-T, B-ISDN ATM Functional Characteristics: ITU-T I.150, 1992.

[2-4] 川崎 健, 西 哲也, 仲道 耕二, 宗宮 利夫, 江崎 裕, “ATM 網におけるデータ交換サービスの一検討,” 信学技法, SSE95-188, IN95-132, March 1996.

[2-5] W.J.Bolosky, J.S.Barrera, R.P.Draves, R.P.Fitzgerald, G.A.Gibson, M.B.Jones, P.Levi, N.P.Myhrvold, and R.F.Rashid, “The Tiger Video Fileserver,“ NOSSDAV 96, pp.97-104, April 1996.

[2-6] 斎藤 洋, “ATM 網におけるトラヒック課題の研究状況,” 信学誌, vol.75, no.5, pp.496-502, May 1992.

[2-7] 阿部 睦, “端末資源予約方式の一考察,“ 信学技法, CQ96-19, Oct.1996.

[2-8] 藤木 正也, 雁部 顕一, 通信トラヒック理論, 丸善, 1980.
[2-9] 大久保 一彦, 加藤 由花, 鎌田 智之, 平野 剛, “ATM ネットワークを用いたマルチメディアシステムの設計と品質評価に関する一考察,” 信学会論文誌 B-I, Vol.J80-B-I, No.6, pp.3130321, 1997 年 6 月.

[3-1] T. Nelms, R. Perdisci, and M. Ahamad, “ExecScent : mining for new C&C domains in live networks with adaptive control protocol templates”, 22nd USENIX Conf., pp.589-604, Aug. 2013.

[3-2] Benedetto, et.al., "Language trees and zipping", Phys. Rev. Lett., vol.88, 2002.

[3-3] Keogh, et.al., "Towards Parameter-Free DataMining", KDD, pp. 206-215 ,2004.

[3-4] Marton, et.al., "On compression-based text classification", European Conference on Information Retrieval, pp. 300-314 , 2005.

[3-5] Bratko, et.al., “Spam filtering using statistical data compression”, Journal of Machine Learning Research, vol.7, pp.2673-2698, 2006.

[3-6] 西田他, 「データ圧縮によるツイート話題分類」, 日本データベース学会論文誌, Vol.10, No.1, 2011.

[3-7] 足達 他,「圧縮類似度による楽譜からの作曲者の判定」, DEIM2013.

[3-8] 岡野靖, 熊谷充敏, 谷川真樹, 大嶋嘉人, 愛甲健二, 梅橋一充, 村上純一, 「マルウェア静的判定における誤判定を低減させる誤判定グッドウェアを活用した事例選択手法」, 信学技報, Vol.115, No.224, pp.163-170, 2015.

[3-9] L. E. Dodd and M. S. Pepe. “Partial AUC estimation and regression.”, Biometrics, 59(3), pp.614-623, 2003.

[3-10] P. Tischer, “A modified Lempel-Ziv-Welch data compression scheme.”, Aust. Comp. Sci. Commun. 9, 1, pp.262-272, 1987.

[4-1] マルウェア対策, 映像情報メディア関連のセキュリティ〔第４回〕, 井上大介

https://www.jstage.jst.go.jp/article/itej/69/4/69_317/_pdf

[4-2] Symantec Security Response, “Latest Intelligence for April 2017,”

https://www.symantec.com/connect/blogs/latest-intelligence-april-2017

[4-3] M.Akiyama et al., “Client Honeypot Multiplication with High Performance and Precise Detection,” IEICE Trans., Vol.E98.D, No.4, 2015.

[4-4] Y.Takata et al., “MineSpider: Extracting Hidden URLs behind Evasive Drive-by-download Attacks,” IEICE Trans., Vol.E99.D, No.4, 2016.

[4-5] D. Sinegubko, “jQuery.min.php Malware Affects Thousands of Websites”

https://blog.sucuri.net/2015/11/jquery-min-php-malware-affects-thousands-of-websites.html

[4-6] Y. Zhou and W. Xu, “Angler Exploit Kit Continues to Evade Detection: Over 90,000 Websites Compromised”

https://researchcenter.paloaltonetworks.com/2016/01/angler-exploit-kit-continues-to-evade-detection-over-90000-websites-compromised/

[5-1] Dolan-Gavitt B., 2007. The vad tree: a process-eye view of physical memory. Digit. Invest. 4, 62-64.

[5-2] UNWIND_INFO 構造体, Visual Studio 2015

https://msdn.microsoft.com/ja-jp/library/ddssxxy8.aspx

[5-3] Rekall, An advanced forensic and incident response framework.

http://www.rekall-forensic.com/

[5-4] Pshoul, D., 2017. community/dimapshoul at master volatilityfoundation/community github.
https://github.com/volatilityfoundation/community/tree/master/DimaPshoul

[6-1] David Wagner on Adversarial Machine Learning at ACM CCS’17.

https://medium.com/syncedreview/david-wagner-on-adversarial-machine-learning-at-acm-ccs17-f75e729a96d8

[6-2] AI ネットワーク社会推進会議, 総務省

http://www.soumu.go.jp/main_sosiki/kenkyu/ai_network/

[6-3] 革新知能統合研究（AIP）センタについて

https://aip.riken.jp/about-aip/?lang=ja

研究業績

＜学会誌等＞

1. Kazuhiko Ohkubo, "Schema Translation into a Unified Model for Service Operation", GLOBECOM'92, 1992-Dec
2. Hiroshi Arimichi, Kazuhiko Ohkubo and Makoto Jinguji: "High-performance processing of traffic data for service operation systems", 1994 IEEE GLOBECOM. Communications: The Global Bridge, 28 Nov.-2 Dec. 1994
3. Yuzo Fujita, Kazuhiko Ohkubo and Hiroshi Tokunaga: "Requirements for intelligent network service operations", Journal of Network and Systems Management, June 1995, Volume 3, Issue 2, pp 195-216
4. 大久保 一彦, 加藤 由花, 鎌田 智之, 平野 剛: “ATM ネットワークを用いたマルチメディアシステムの設計と品質評価に関する一考察”, 電子情報通信学会論文誌 B-I, Vol.J80-B-I, No.6, pp.313-321, 1997 年 6 月
5. Tohru Nishimura, Kazuhiko Ohkubo and Yasushi Yamanaka: "Implementation Methodology for Customer Trouble Ticketing Interface", Session 9: Network Management, APSITT'99 Technical Program
6. Yuta TAKATA, Mitsuaki AKIYAMA, Takeshi YAGI, Takeo HARIU, Kazuhiko OHKUBO and Shigeki GOTO, "Identifying Evasive Behaviors in Malicious Websites by Analyzing Redirection Differences", IEICE TRANS. INF. & SYST., VOL.E101-D, NO.11 NOVEMBER 2018

7. 畑島隆, 谷本茂明, 金井敦, 冨士仁, **大久保一彦**：“改善型情報セキュリティコンディションマトリクスによる大学生の情報セキュリティ疲れ対策の提案”, 情報処理学会論文誌, Vol.59, No.12, Dec 2018
8. Yuto Otsuki, Yuhei Kawakoya, Makoto Iwamura, Jun Miyoshi and **Kazuhiko Ohkubo**: "Building stack traces from memory dump of Windows x64", DFRWS EU 2018 - Proceedings of the Fifth Annual DFRWS Europe, Digital Investigation 24 (2018), S101-S110
9. T. Hatashima, K. Nagai, A. Kishi, H. Uekusa, S. Tanimoto, A. Kanai, H. Fuji, and **K. Ohkubo**, “Evaluation of the Effectiveness of Risk Assessment and Security Fatigue Visualization Model for Internal E-Crime,” 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), vol.1, pp.707-712, 2018
10. Shigeaki Tanimoto, Takuya Murakami, Phenpimon Wilaratana, Tsutomu Konosu, Takashi Hatashima, Hitoshi Fuji, **Kazuhiko Ohkubo**, Koichi Mizutani and Prajak Chertchom: "Risk Assessment for "Texting While Walking" Based on a User's Viewpoint", The 4th RMUTT Global Business and Economic Conference (RTBEC), pp.139-148, 24-25 May, 2018
11. Shigeaki Tanimoto, Chise Nakamura, Motoi Iwashita, Shinsuke Matsui, Takashi Hatashima, Hitoshi Fuji, **Kazuhiko Ohkubo**, Junichi Egawa and Yohsuke, Kinouchi: "Proposal of Secure Business Model by Visible Light Communication System", 2018 3rd International Conference on Enterprise Architecture and Information Systems (EAIS), pp.817-822, 7-13 July, 2018
12. Shigeaki Tanimoto, Chise Nakamura, Motoi Iwashita, Shinsuke Matsui, Takashi Hatashima, Hitoshi Fuji, **Kazuhiko Ohkubo**, Junichi Egawa and Yohsuke Kinouchi: "Secure Visible Light Communication Business Architecture based on Federation of ID

Management", The 12th International Workshop on Advanced Distributed and Parallel Network Applications (ADPNA-2018), pp.578-589, 5-7 September 2018

13. Kazuhiko Ohkubo: "Cyber-physical Security in an Age of Digital Transformation", Keynote Speech 1, IWIN (International Workshop on Informatics) 2018, In Proc. of IWIN2018, page 58
14. Yasushi Okano, Kazunori Kamiya, Atsutoshi Kumagai, Taishi Nishiyama, Bo Hu, Masaki Tanikawa and Kazuhiko Ohkubo: "Detection of malware-infected host with data compression algorithm", Session 4: Network and Security, IWIN (International Workshop on Informatics) 2018, In Proc. of IWIN2018, pp. 87-94
15. Kazuhiko Ohkubo, Tetsuhisa Oda, Yuki Koizumi, Tetsushi Ohki, Masakatsu Nishigaki, Toru Hasegawa and Yoshinobu Kawabe: "Trust representation under confusion and ignorance", Session 7: Trust and Risk, IWIN (International Workshop on Informatics) 2018, In Proc. of IWIN2018, pp. 195-201
16. Hayato Oba, Shigeaki Tanimoto, Atsushi Kanai, Takashi Hatashima and Kazuhiko Ohkubo: "A Proposal of Improvement of "Return on Security Investment" Model for IT governance", ProMAC2018 (12th International Conference on Project Management)
17. Shogo Maeda, Atushi Kanai, Shigeaki Tanimoto, Takashi Hatashima and Kazuhiko Ohkubo: "A Botnet Detection Method on SDN using Deep Learning", 37th IEEE International Conference on Consumer Electronics (ICCE), 11-13 January 2019
18. Yasushi Okano, Kazunori Kamiya, Atsutoshi Kumagai, Taishi Nishiyama, Bo Hu, Masaki Tanikawa and Kazuhiko Ohkubo: "Compression Algorithm Contribution for Infected-host

Detection", International Journal of Informatics Society (IJIS), Vol.11 (手続き中)

19. **Kazuhiko Ohkubo**: " Cybersecurity Technologies Essential in the Digital Transformation Era", International Journal of Informatics Society (IJIS), Vol.11 (手続き中)

＜その他＞

20. **大久保 一彦** [他]: "INMOS-VO の DB 構造 (IN サービスオペレーションシステム(INMOS-VO)) ", NTT R&D 42(10), p1279-1286, 1993-10
21. **Kazuhiko Ohkubo** and Masaaki Yokoyama: "Data Structure of Traffic Database", IEICE 1991 Spring National Convention, vol. B-661
22. **大久保一彦**, 横山雅明: "トラヒック DB 構成法の検討," 研究開発資料第 05786 号, 1991/09/05
23. **大久保一彦**, 長島雅夫, 山田光博: "サービスオペレーション管理情報の体系化," 研究開発資料第 07447 号, 1992/10/15
24. **大久保一彦**, 有道啓史, 高橋英樹, 森田誠至, 森廣政治: "IN サービスオペレーションシステム・機能設計書 第 2 部:DB 編," 研究開発資料第 09435 号, 1994/03/04
25. **大久保一彦**, 森田誠至, 藤田裕三, 徳永裕史: "カスタマサービス OpS・体系化情報," 研究開発資料第 10079 号, 1994/06/17
26. **大久保一彦**, 山中康史, 岡部恵一, 森隆彦: "NMF 分析書テンプレート," 研究開発資料第 14073 号, 1997/05/23
27. **大久保一彦**, 山中康史: "カスタマトラブルチケット管理システム機能仕様書," 研究開発資料第 14582 号および 14583 号, 1997/11/07
28. **大久保一彦**, 古川浩, 山中康史: "SML-NML 故障管理機能仕様書," 研究開発資料第 16163 号, 1999/03/26
29. Tohru Matsuno, Toru Kawamura, **Kazuhiko Ohkubo**, Hidetsugu Kobayashi, Katsumi Takahashi, and Shigeru Kayaguchi: "Emergence

of New Cyber Attacks and Future Directions in Security R&D," NTT Technical Review, Vol.10, No.10, ntr201210fa1, Oct 2012

30. **大久保一彦**: "これまでのやり方はもう通用しない！IoT 時代に蔓延るサイバー脅威の一歩先行く革新技術," NTT 技術ジャーナル Vol.29 No.4, pp.6-8, Apr 2017

31. **Kazuhiko Ohkubo**: "Research and Development of Security Concerns Relating to Growing Threats and Business Opportunities," NTT Technical Review, Vol.15, No.5, ntr201705fa1, May 2017

32. **Kazuhiko Ohkubo**: "Research and Development of Advanced Security Measures to Protect Customers from Sophisticated and Large-scale Cyberattacks," NTT Technical Review, Vol. 16, No.5, ntr201805fa1, May 2018

【特許】

公開番号	出願日	公開・公表日	出願人・権利者（公報）	発明者または考案者	代理人	発明等の名称
特開2016-009308	2014/6/24	2016/1/18	日本電信電話株式会社;エヌ・ティ・ティ・コミュニケーションズ株式会社	松村　隆宏;大久保　一彦;神谷　和憲;畑田　充弘;奥村　恭弘	特許業務法人酒井国際特許事務所;宮崎　昭夫	マルウェア検出方法、システム、装置、ユーザPC及びプログラム
特開2009-237807	2008/3/26	2009/10/15	日本電信電話株式会社	田村　高廣;大久保　一彦	酒井　宏明;中辻　史郎	脆弱性診断実施装置および診断スケジュール作成プログラム
特開2009-223769	2008/3/18	2009/10/1	日本電信電話株式会社	田村　高廣;大久保　一彦	宮崎　昭夫	電子メール削除装置、電子メール削除方法およびプログラム
特開2009-223768	2008/3/18	2009/10/1	日本電信電話株式会社	田村　高廣;大久保　一彦	宮崎　昭夫	電子メール誤削除防止装置、電子メール誤削除防止方法およびプログラム
特開2003-256669	2002/2/28	2003/9/12	日本電信電話株式会社	大久保　一彦	秋田　収喜;近野　恵一	ユビキタス情報サービスシステム、情報サービス提供方法、ローカルサーバ、ローカルサーバの情報サービス提供方法、プログラム、記録媒体
特開2003-244678	2002/2/15	2003/8/29	日本電信電話株式会社	大久保　一彦	山川　政樹	特定シーンコンテンツ配信システム
特開2003-242404	2002/2/15	2003/8/29	日本電信電話株式会社	大久保　一彦	山川　政樹	販売アシストサービスシステム
特開2003-242155	2002/2/15	2003/8/29	日本電信電話株式会社	大久保　一彦	山川　政樹	フレックスセミナーサービスシステム
特開2003-141051	2001/11/7	2003/5/16	日本電信電話株式会社	斉藤　典明;大戸　健一;大久保　一彦	秋田　収喜;近野　恵一	コミュニケーション適応型メッセージ配信サービスおよびコミュニケーションサーバ
特開2002-320049	2001/4/20	2002/10/31	日本電信電話株式会社	大久保　一彦	三好　秀和;三好　保男	仲介処理システム
特開2002-318983	2001/4/20	2002/10/31	日本電信電話株式会社	大久保　一彦	三好　秀和;三好　保男	取引仲介システム
特開2000-253074	1999/3/1	2000/9/14	日本電信電話株式会社	大久保　一彦	若林　忠;金田　暢之	サービス故障情報提供方法およびサービス故障情報提供システムおよびサービス故障情報提供プログラムを記録した記録媒体
特開2000-236331	1999/2/15	2000/8/29	日本電信電話株式会社	鈴木　俊明;大久保　一彦	伊東　忠彦	サービス故障情報管理システム及び方法、サービスプロバイダ、並びに、サービス故障情報流通プログラムを記録した記録媒体
特開平11-232200	1998/2/16	1999/8/27	日本電信電話株式会社	大久保　一彦	伊東　忠彦	サービス運用情報通知方法及びシステム及びサービス運用情報通知プログラムを格納した記憶媒体
特開平11-066140	1997/8/27	1999/3/9	日本電信電話株式会社	大久保　一彦;豊泉　洋;加藤　由花	伊東　忠彦	ネットワーク自動設計方法及び装置、並びに、ネットワーク自動設計プログラムを格納した記憶媒体
特開平08-202597	1995/1/24	1996/8/9	日本電信電話株式会社	大久保　一彦	三好　秀和;三好　保男	スキーマ変換機構編成方式
特開平08-046617	1994/7/26	1996/2/16	日本電信電話株式会社	大久保　一彦	伊東　忠彦	マルチメディアサービスアクセス方法及びマルチメディアサービスアクセス方式
特開平07-235984	1994/2/22	1995/9/5	日本電信電話株式会社	増井　信彦;大久保　一彦;神宮司　誠	伊東　忠彦	トラヒックデータ編集におけるグループ化方式及びトラヒックデータ編集方法
特開平07-183947	1993/12/24	1995/7/21	日本電信電話株式会社	大久保　一彦;有道　啓史	杉村　暁秀;杉村　興作	トラヒックデータ管理方法
特開平07-044456	1993/8/2	1995/2/14	日本電信電話株式会社	大久保　一彦	森田　寛;小笠原　吉義	時系列データアクセス処理方式
特開平06-222967	1993/1/28	1994/8/12	日本電信電話株式会社	大久保　一彦	伊東　忠彦	データアクセス方式
特開平04-111640	1990/8/31	1992/4/13	日本電信電話株式会社	大久保　一彦;横山　雅明;徳永　裕史	山川　政樹;黒川　弘朗	網データ管理方法

博士論文（第1～5章）と研究業績（項番）の対応

第1章　緒言

19. Kazuhiko Ohkubo: ″ Cybersecurity Technologies Essential in the Digital Transformation Era″, International Journal of Informatics Society (IJIS), Vol.11 (手続き中)

13. Kazuhiko Ohkubo: "Cyber-physical Security in an Age of Digital Transformation", Keynote Speech 1, IWIN (International Workshop on Informatics) 2018, In Proc. of IWIN2018, page 58

第2章　可用性に配慮したマルチメディアシステムの設計技術

4. 大久保 一彦, 加藤 由花, 鎌田 智之, 平野 剛: “ATMネットワークを用いたマルチメディアシステムの設計と品質評価に関する一考察”, 電子情報通信学会論文誌B-I, Vol.J80-B-I, No.6, pp.313-321, 1997年6月

第3章　通信ログ分析に基づく高精度なマルウェア感染検知技術

18. Yasushi Okano, Kazunori Kamiya, Atsutoshi Kumagai, Taishi Nishiyama, Bo Hu, Masaki Tanikawa and Kazuhiko Ohkubo: ″Compression Algorithm Contribution for Infected-host Detection″, International Journal of Informatics Society (IJIS), Vol.11 (手続き中)

14. Yasushi Okano, Kazunori Kamiya, Atsutoshi Kumagai, Taishi Nishiyama, Bo Hu, Masaki Tanikawa and Kazuhiko Ohkubo: "Detection of malware-infected host with data compression algorithm", Session 4: Network and Security, IWIN (International Workshop on Informatics) 2018, In Proc. of IWIN2018, pp. 87-94

第 4 章　悪性 Web サイトにおける Evasive コード特定技術

6. Yuta TAKATA, Mitsuaki AKIYAMA, Takeshi YAGI, Takeo HARIU, Kazuhiko OHKUBO and Shigeki GOTO, "Identifying Evasive Behaviors in Malicious Websites by Analyzing Redirection Differences", IEICE TRANS. INF. & SYST., VOL.E101-D, NO.11 NOVEMBER 2018

第 5 章　64 ビット版 Windows のメモリダンプからのスタックトレース構築技術

8 Yuto Otsuki, Yuhei Kawakoya, Makoto Iwamura, Jun Miyoshi and Kazuhiko Ohkubo: "Building stack traces from memory dump of Windows x64", DFRWS EU 2018 - Proceedings of the Fifth Annual DFRWS Europe, Digital Investigation 24 (2018), S101-S110