博 士 論 文

# 効果的・効率的なインシデントレスポンスの実現技術

Hiroki HADA

羽田 大樹

情報セキュリティ大学院大学

情報セキュリティ研究科

情報セキュリティ専攻

2019 年 3 月

博 士 論 文

# 効果的・効率的なインシデントレスポンスの実現技術

Hiroki HADA

羽田 大樹

情報セキュリティ大学院大学

情報セキュリティ研究科

情報セキュリティ専攻

2019 年 3 月

# 論旨

コンピューターが社会基盤として，さらには企業や組織の戦略を実現するための手段となっている現代において，サイバーセキュリティへの対応は組織が持続的・発展的な活動を行うための重要な課題である．CSIRT は日々発生する組織内の大小様々なセキュリティインシデントをハンドリングすることで本業を支える重要な組織であるが，この業務は適切かつ迅速に行われなければならない．本論文では，組織におけるマルウェア感染に対するリスクの最小化を目指して，CSIRT のインシデントレスポンス業務を“効果的”・“効率的”に行うための 2 つの技術を提案した．

1 つは，Web ブラックリストを活用するための分類の提案である．ブラックリストには Web サイトが悪性と判断された理由やレピュテーションなどが付与されることがあるが，インシデントレスポンスにおいて CSIRT が合理的に判断するための情報を提供しているとはいえない．そこで，合理的な判断を可能とするため「所有者」「コンテンツ」「現在の状態」「最終確認」という 4 項目による悪性 Web サイト情報の分類を提案した．公開されている 38 種類のブラックリストには，この分類に相当する情報がほとんど含まれていなかった．さらに，インシデント判定が求められた 400 件の悪性 Web サイト情報を手動で調査し，外部からの調査によって分類ができることを示した．この分類を活用することで速やかな対応ができることをケーススタディとして紹介し，提案分類が有効に働くことを示した．

もう 1 つは，フォレンジックにおけるマルウェア解析を効率化する手法の提案である．インシデントにおいてマルウェアの亜種が共通的に使用された場合，過去に解析したマルウェアの関数に相当するコードの場所を特定できると速やかに解析が行える．マルウェア解析に適したコード比較アルゴリズムとして，関数における制御フローグラフの編集距離と命令列の最長共通部分列を用いて関数を「検索」する BinGrep を提案した．実際に APT

攻撃で使用されたマルウェアを評価したところ，Emdivi の 11 検体の評価では，インシデントレスポンスにおいて重要であった 27 個の関数の 85%について正解を出力した．また，Emdivi と PlugX のそれぞれ 2 検体の全関数について評価を実施し，マルウェア解析において提案方式が有効であることを示した．

本論文では，CSIRT におけるインシデントレスポンスの行程を分析し，業務を“効果的”・“効率的”に行うための 2 つの技術を提案した．Web ブラックリストに紐づいた分類を用いることで,「トリアージ」における端末特定の要否，調査項目，優先度の判断と「対応実施」における通信遮断の要否の判断がマニュアルに基づいて合理的に行えるようになった．また「事象の分析」において過去に解析したマルウェアの亜種を用いて解析すべき箇所を特定することで通信先や暗号処理の解析が速やかに開始できるようになり，影響範囲が特定できるようになった．この提案により，組織の持続的・発展的な発展に貢献した．

# Abstract

It is an important challenge for the organizations to strengthen cyber security currently where computers are commonly used as a social infrastructure and a means to realize strategies of companies and organizations. CSIRT is an important organization supporting the core business by handling various large and small security incidents in the organization which occur day by day, but these tasks must be handled efficiently and appropriately. In this paper, in order to minimize the risk of malware infections in an organization, we propose two technologies for "effective" and "efficient" incident response for CSIRT.

First, we propose classification for the blacklist of malicious web sites. Blacklist sometimes gives reasons why a website is judged to be malicious or the reputation score, etc. However, it doesn't provide suitable information for reasonable judgment for CSIRT in the incident response. So, we propose malicious web sites blacklist classification which is composed of "Owner", "Content", "Status", and "Update Time" in order to be able to make a reasonable judgement. The 38 public blacklists don't contain much information corresponding to this classification. In addition, we manually investigate 400 malicious web site information for which incident judge is requested and show that they can be classified by external investigation. We introduce several case studies that CSIRT can respond promptly by utilizing this classification and show that proposed classification works effectively.

The other, we propose a method for improving malware analysis efficiency in forensics. If resemble malware is used in some attacks, it is possible to analyze it immediately if the address of the code corresponding to the malware function analyzed in the past can be specified. As the algorithm which is suitable for malware analysis, we propose BinGrep that

"searches" for function using both the edit distance of control flow graph of functions and longest common substrings of instructions. We evaluate the malware used in the real APT attack. In the evaluation of 11 Emdivi samples, we get the correct answer for 85% of the 27 functions which are important in the incident response. In addition, we evaluate all functions of two samples of Emdivi and PlugX, and show that this method is effective in malware analysis.

In this paper, we analyzed the process of incident response in CSIRT and proposed two technologies for effective and efficient incident response. By using this web blacklist classification, it became possible to judge necessity of client identification in the "Triage" process, incident response priority and necessity of block in the "Response" process. Furthermore, by specifying the function to be analyzed using the malware family analyzed previously, it became possible to analyze the process about access and cryptgraphy. These proposals contributed to the organization's sustainable activity.

# 目次

# 図目次

# 表目次

# 用語集

本論文では，使用する用語を以下のとおり定義する．

| No. | 用語 | 説明 |
|---|---|---|
| 1 | 組織（Organization） | 複数人で構成され，同一の目的を達成するために活動を行う団体．企業のような営利目的の団体だけでなく，公共団体といった非営利目的の団体も含む． |
| 2 | CSIRT（Computer Security Incident Response Team） | 組織のセキュリティに責任を持ち，日々の脅威から組織を守ることで組織の持続的な活動を支援する役割を担うチーム． |
| 3 | セキュリティインシデント（Security Incident） | 情報漏えい，不正アクセス被害，マルウェア感染といった，組織におけるセキュリティ上の脅威となる事象． |
| 4 | インシデントレスポンス（Incident Response） | CSIRT の役割のひとつで，発生したインシデントを収束させる営み． |
| 5 | トリアージ（Triage） | インシデントレスポンスにおいて，対応要否の判断や対応の優先度を決定すること． |
| 6 | Web ブラックリスト（Web Blacklist） | Web アクセスの利用に際し，利用者を脅威から守るために使用される悪性なアクセス先を列挙したリスト． |
| 7 | 脅威情報（Threat Intelligence） | 脅威に関する情報の総称．Web ブラックリストも含まれるが，Web 以外の通信先やマルウェア，攻撃者像に関する情報も含む． |
| 8 | マルウェア（Malware） | 悪意のあるソフトウェアの総称．本論文では x86，x64 上の Windows 環境で動作するマルウェアを対象とする． |
| 9 | C2 | マルウェアを制御するための指令を与える主体．C&C（Command and Control）とも呼ばれる． |

| | | |
|---|---|---|
| 10 | マルウェア解析<br>（Malware Analysis） | マルウェアの挙動や特徴を特定する営み. |
| 11 | 静的解析（Static Analysis） | マルウェア解析手法の分類のひとつ．マルウェアを動作させることなく，マルウェアの逆アセンブリを用いてマルウェア解析すること. |
| 12 | コールグラフ（Call Graph） | 実行ファイルの逆アセンブリにおいて，関数をノードとして，関数の呼び出し元と呼び出し先をエッジとして持つ有向グラフ. |
| 13 | 基本ブロック（Basic Block） | 実行ファイルの逆アセンブリにおいて，jmp 等の制御転送命令によるジャンプ先を境界として分割したブロック. |
| 14 | 制御フローグラフ<br>（Control Flow Graph） | 基本ブロックを頂点，制御転送命令によるジャンプを辺で定義した有向グラフ. |

# 第 1 章

# 序論

本章では，本研究を取り巻く背景とその位置付け，および本研究の目的について概説する．また，本論文の構成について説明する．

## 1.1 本研究の背景と目的

コンピューターが社会基盤として，さらには企業や組織の戦略を実現するための手段となっている現代において，サイバーセキュリティへの対応は組織が持続的・発展的な活動を行うための重要な課題である．特に，不特定多数を攻撃対象としたばらまき型のマルウェアスパムやドライブバイダウンロード攻撃だけでなく，特定の組織や人物を狙った APT 攻撃による大規模な感染被害を経験した組織も多い．カスペルスキー社は 2015 年の BlueTermite という攻撃キャンペーンにおいて, 日本国内だけでも 300 以上の組織が深刻な侵害を受けていたと報告している[60]. これらのインシデントでは多数の組織に共通して同じマルウェアの亜種や同様の攻撃手法が用いられていた．このようなインシデントによる被害の発生により，サービスの停止や情報漏洩など組織の活動に大きな影響を与えてしまうだけでなく，メディアでも大きく取り上げられて信頼を失墜してしまうという事件が日常的に発生している．

サイバーセキュリティが経営課題として一般的に認識されるようになった現代においてそのようなリスクに備えるために，日々発生する大小様々な組織内のセキュリティインシデントをハンドリングする CSIRT（Computer Security Incident Response Team）を設置し

ている．CSIRT は事故が発生するという前提に則った上でセキュリティリスクを抑制し，組織が持続的に活動できるようにすることに責任を持つ役割を担う[1]．具体的には，インシデントを未然に防ぐこと，ログ分析を行い発生してしまったインシデントを発見すること，発生してしまったインシデントの原因や影響を調査して再発防止を行うためのインシデントレスポンスを行うことをミッションとする．また，これを実現するための手段としての情報収集や連絡窓口といった業務を担当する．

CSIRT は日々発生するインシデントに対してインシデントレスポンスを行っているが，この中で対応の要否や優先度を決定するトリアージ作業を行いながら，マルウェアの駆除や端末の隔離，再インストール，再発防止といった基本的な対応を速やかに行う．深刻な影響が疑われる場合はさらにフォレンジック調査やマルウェア解析を行い，コストをかけた詳細な解析によって被害範囲や根本原因を正確に特定する必要がある．

インシデントレスポンスにおける対応は一刻の猶予も許されない状況に置かれる．判断に間違いがあった場合は手戻りが発生するだけでなく，防げたはずの被害が発生して取り返しのつかない事態となってしまう可能性もある．また，その判断に時間をかけても同様となる．インシデントレスポンスは確実かつ迅速に行うことが求められるが，一般的にセキュリティ人材の不足や CSIRT 業務に割り当てられるリソースに限りがあるといった事情がある．これらのインシデントレスポンスをいかに“効果的”かつ“効率的”に行うかという課題を抱えている[1]．

本論文では，組織におけるマルウェア感染に対するリスクの最小化を目指して，CSIRT のインシデントレスポンス業務を“効果的”・“効率的”に行うため，2 つの課題に取り組んだ．

1 つは，Web ブラックリストを活用した合理的かつ速やかなインシデントレスポンスの実現である．ブラックリストには Web サイトが悪性と判断された理由やレピュテーションなどが付与されることがあるが，インシデントレスポンスにおいて CSIRT が合理的に判断するための情報を提供しているとはいえない．スキルを持つ担当者がインシデント発見後に改めて脅威情報を調査し，経験に基づいた判断を行う必要がある．そこで，インシデントレスポンスにおいてブラックリストに求められる情報を整理し，あらかじめ付与されるべき属性をブラックリストの分類として定義した．合理的に判断するための情報を事前に

1 これらの業務の一部を SOC（Security Operation Center）などの専門組織に任せる場合もあるが，SOC においても内包する課題に本質的な違いはない．

与えることで，事象を正しく把握して“効果的”な対応を可能とすること，マニュアルに基づいた判断により時間をかけずに“効率的”なトリアージを可能とすることを目指した．

もう 1 つは，フォレンジックにおけるマルウェア解析の効率化の実現である．深刻なインシデントにおける詳細調査は解析作業を伴うため多大な時間とコストが必要となるが，“効果的”な対策が求められる場面においてはこの作業に取り組まなければならない．複数のインシデントにおいてマルウェアの亜種が共通的に使用される場合に，過去に解析したマルウェアの関数に相当するコードの場所を特定できると速やかに解析が行える．被害範囲を調査するために解析したい機能がマルウェアのどこに実装されているか特定することで，“効率的”に影響を調査することができる手法を目指した．

本論文では，この 2 つの課題について取り組み，組織の持続的・発展的な発展に貢献するための技術を提案した．

## 1.2 本論文における貢献

本論文では，CSIRT のインシデントレスポンス業務に着目してこの行程を分析し，インシデントレスポンスを“効果的”・“効率的”に行うための 2 つの技術を提案した．

1 つは，Web ブラックリストを活用するための分類の提案である．ブラックリストには Web サイトが悪性と判断された理由やレピュテーションなどが付与されることがあるが，インシデントレスポンスにおいて CSIRT が合理的に判断するための情報を提供しているとはいえない．事象を正しく把握して“効果的”な対応を可能とすること，マニュアルに基づいた判断により時間をかけずに“効率的”なトリアージを可能とすることを実現するため，ブラックリストに求められる情報を整理し，インシデントレスポンスにおいて合理的な判断を行うにあたって必要な属性情報をブラックリストの分類として提案した．具体的な貢献は以下の 3 点である．

1-1. CSIRT がインシデントレスポンスにおいて合理的な判断を行うための「所有者」「コンテンツ」「現在の状態」「最終更新」という 4 項目による悪性 Web サイト情報の分類を提案した．

1-2. 公開されている 38 種類のブラックリストの仕様について調査し，この分類に相当する情報が十分に含まれていないことを示した．

1-3. 2017 年 9 月 13 日から 2017 年 11 月 23 日までの期間において，サンプリングした 100 組織で実際にインシデント判定を求められた 400 件の悪性 Web サイト情報について詳細な分析を行い，外部からの調査によって分類ができることを示した．また，この分類を活用することで速やかに合理的な判断ができることをケーススタディとして紹介した．

もう 1 つは，フォレンジックにおけるマルウェア解析を効率化する手法の提案である．深刻なインシデントにおいては真に“効果的”な対策が求められるためマルウェア解析による詳細調査を行うが，すでに対応したことのある同様のインシデントで用いられたマルウェアを活用し，被害範囲を特定するために解析したい機能がマルウェアのどこに実装されているか特定する手法を提案した．既存手法ではマルウェアの全体を比較することでこの目的を実現していたが，出力を間違えた場合に有効な情報が得られないという問題があった．提案手法ではマルウェアにおける解析したい機能を指定して検索することで複数の候補を出力し，“効率的”に解析作業に取りかかることを実現した．具体的な貢献は以下の 3 点である．

2-1. 制御フローグラフの編集距離と逆アセンブリの一致率を用いた新しいアルゴリズムにより，過去に解析したマルウェアの関数に相当する関数を確度の高い順に「検索」して出力する BinGrep という方式を提案した．

2-2. 提案手法に対して，既存手法の代表例である BinDiff との比較評価を行った．正常プログラムとして GNU bash と GNU binutils それぞれの古いバージョンと新しいバージョンを比較し，11,049 個の関数のうち 90.3% について正解を出力した．特に，BinDiff が失敗した 1,069 個の関数のうち 65.8%について提案方式が正解を出力し，BinDiff と提案方式を併用することでより多くの関数を特定できることを示した．

2-3. 実際に APT 攻撃で使用された RAT である Emdivi と PlugX について評価を

行った．Emdivi の 11 個の検体を用いて 27 個の関数を評価した結果，85%について正解を出力した. Emdivi と PlugX のそれぞれ 2 検体における全関数に対して評価を行い，Emdivi の 1,018 個の関数のうち 75%，PlugX の 472 個の関数のうち 90%について正解を出力し，提案方式がマルウェア解析においても有効であることを示した．

Web ブラックリストに紐づいた分類を用いることで,「トリアージ」における端末特定の要否，調査項目，優先度の判断と「対応実施」における通信遮断の要否の判断がマニュアルに基づいて合理的に行えるようになった．また「事象の分析」において過去に解析したマルウェアの亜種を用いて解析すべき箇所を特定することで通信先や暗号処理の解析が速やかに開始できるようになり，影響範囲が特定できるようになった．この提案により，CSIRT はインシデントレスポンスを“効果的”･“効率的”に行うことができるようになった．

## 1.3 本論文の構成

本論文は 6 章から構成される．第 2 章では，CSIRT 業務における課題分析を行い，インシデントレスポンスに求められる技術について説明する．第 3 章では，CSIRT のインシデント発見と初動対応を“効果的”・“効率的”に行うための Web ブラックリストの在り方について議論を行い，CSIRT に最適化したブラックリストの分類を提案する．第 4 章では，CSIRT のインシデントレスポンスにおけるフォレンジック調査でマルウェアの静的解析を迅速に行うための関数を特定する手法を提案する．第 5 章では，CSIRT に対する貢献と今後の展望について議論する．最後に，第 6 章にて本論文をまとめる．

# 第 2 章

# CSIRT 業務における課題分析

本章では，一般的に CSIRT が抱える課題を広く調査して整理し，その中で本論文が対象とする課題について示す．

## 2.1 CSIRT の役割と問題意識

CSIRT は組織が持続的・発展的な活動を行うためのセキュリティに関する運用に責任を持つ．本論文ではサイバーセキュリティを議論の対象としているが，場合によっては物理セキュリティについて責任を持つ場合もある[10]．CSIRT はその機能によって，例えば以下のような役割が存在する[3]．

- 組織内 CSIRT
- 国際連携 CSIRT
- コーディネーションセンター

組織内 CSIRT は組織の中に設置され，その組織のシステムやネットワークをサービス対象とした CSIRT である．企業の場合は企業内 CSIRT とも呼ばれる．国際連携 CSIRT は国や地域をサービス対象とした CSIRT であり，その国を代表してインシデントレスポンスのための窓口として活動する組織である．コーディネーションセンターは協力関係にある他の CSIRT をサービス対象とし，CSIRT 間の情報連携や調整を行う組織である．例えば，

JPCERT/CC は国際連携 CSIRT とコーディネーションセンターの役割を兼ねている. また, CSIRT 同士が連携するためのコミュニティとして発足したフォーラムが存在する．世界的なフォーラムとしては FIRST[4]が，アジア太平洋地域を対象としたものには APCERT[4]がある．日本の CSIRT を対象としたものには日本シーサート協議会といったフォーラムがある[6]．本論文では，組織内 CSIRT が行うインシデントレスポンスについて議論を行う．

組織内 CSIRT における問題意識を整理する．カーネギーメロン大学は CSIRT のためのハンドブックを公開している[9]．また，日本シーサート協議会では，問題意識を共有して解決に向けた議論を行っている[2][7][8]. 現場レベルで CSIRT の運用について議論を行う場合，体制や運用プロセスといった組織論に関する問題として議論されていることが多い. 例えば，CSIRT の立ち上げや運営に関する経営層の理解が中々得られず組織横断的な仮想チームによる手弁当での運用を強いられていること，スキルを持つ人材や割り当てられる人員が不足していること，インシデントの発生予測が難しいため予算や時間の確保ができないこと，迅速な対応のための一元窓口やエスカレーションフローなど運用プロセスが最適化されていないこと，ポリシーや手順といった運用を継続的に見直してワークフローを管理する必要があることなどが挙げられている.

これらの問題をより具体的に落とし込んで考えると CSIRT の業務を支援するための技術や仕組みによって解決できる課題としても捉えることができる．脅威情報や脆弱性情報を広く深く収集すること, CSIRT にとって最適化された脅威情報の共有基盤を構築すること, インシデントを早期に発見して影響を判断する手法を確立すること，迅速にシステムを復旧する技術を実現すること，適切な再発防止を実装すること，といった課題が提起されており，解決に向けた様々な技術的な取り組みがある.

## 2.2 CSIRT の課題に対する関連研究

これらの課題に対して様々なアプローチによる解決手法が提案されている．ここでは，特に技術的な解決手法について整理する.

脅威情報を自動的に収集，もしくは評価する研究が行われている[109][52]．公開されている脅威情報は OSINT（Open Source Intelligence）とも呼ばれ，セキュリティ研究者のレ

ポートやブログ，企業が配信するブラックリストといったものが含まれる．収集した脅威情報やインシデントレスポンスの過程で入手した脅威情報は，組織の中で共有されて事前対策や再発防止策として有効に利用されるべきである．そこで，脅威情報を共有するための基盤に関する研究が行われている[11][12].

マルウェアやインシデントを発見するための研究が行われている．マルウェアそのものを発見する手法はアンチウイルスやサンドボックス解析の機能として古くから研究されている[110]．マルウェアそのものを発見できない場合でも，ネットワークからインシデントが発生した後に気づくための手法も提案されている[111]．また，インシデントが発生してしまった際にフォレンジックやマルウェア解析を行うこともあるが，この作業を自動化する研究も行われている[112]．さらに，インシデントを自動的に復旧する研究も行われている[113].

これらは全てシステム的に解決されることが理想ではあるが，実際は従業員や CSIRT のリテラシーやスキルを伴っていなければならない．そこで訓練環境を構築するための研究も行われている[114].

CSIRT の健全な運用には，人・業務プロセスにおける課題と仕組みにおける課題の両面から解決を試みなければならない．“効果的”・“効率的”なインシデントレスポンスを考えた場合に，仕組みだけで解決できることが理想であり，そのような技術が提案されている．ところが，実際は仕組みだけでは全て解決できないことも多く，最後に人による調査や検証が必要となる場合がある．このような工程をサポートする技術はあまり提案されていない．

## 2.3 本論文で取り組む課題

本論文では，CSIRT が行うインシデントレスポンスを“効果的”・“効率的”に実施するための手法を提案する．CSIRT ではインシデントレスポンスを「トリアージ」「事象の分析」「対応計画」「対応実施」というプロセスで行う[34].「トリアージ」「対応実施」においては脅威情報を調査してインシデント前後のログと照合することで対応の優先度や対応要否を決定する.「事象の分析」においては，マルウェア解析によって影響範囲を特定する．こ

こでは，本論文で取り組む 2 つの課題について説明する．

### 2.3.1 脅威情報の調査における課題

インシデント発見工程において, CSIRT は Web ブラックリストに代表される脅威情報を活用して，脅威を発見して初動対応の判断を行う．一般的に，ネットワークが適切に管理されたシステムにおいて従業員は外部に接続できる手段は限られているが，Web アクセスやメールは許可される．そのためマルウェアは Web アクセスやメールを通じて端末に感染し，Web アクセスを通じて外部からコントロールされる場合が多い．組織が業務を行う上でインターネットを利用して Web サイトにアクセスすることは現代では必要不可欠であるが，Web 上には特に多くの脅威が存在するため，悪性 Web サイトのブラックリストを活用して脅威の発見やアクセス制御を実現する．

世界中の攻撃者が取得する悪性ドメインや利用する IP アドレスは変化し続けるため，ブラックリストに登録される悪性 Web サイトは膨大となる．マイクロソフト社は 180 億の Web ページをスキャンしたところ，1000 ページあたり 0.17，すなわち 306 万の侵害ページを発見したと報告している [17]．脅威の全てを網羅している訳ではないが，すでに判明している脅威に対してブラックリストは有効に働くため，ブラックリストに登録する悪性 Web サイトを収集する技術，ブラックリストの精度を評価・向上する技術に関する研究は数多く行われている．本研究では，このような状況において CSIRT が“効果的”・“効率的”に脅威を発見して初動対応を行うために，さらに Web ブラックリストに対して CSIRT に最適なラベルの付与が必要であることを指摘する．

SOC では組織において発生した脅威を正確に伝えるため，インシデントが疑われる事象を発見した際にアナリストが脅威情報の素性を調査し，CSIRT に対して合理的な説明を行っている．一般的に，悪性 Web サイトに関する情報は「ブラックリスト」と一括りに表現されることが多い．Web サイトへのアクセスがブラックリストによって拒否された場合，アクセスした人はその理由についてあまり考える必要はなく，そのサイトへのアクセスを諦めるか，業務を継続するための代替手段を考えれば良い．一方で，CSIRT の視点で考えた場合に，その意味合いは様々となる．インシデントレスポンスを行うことを考えた場合，そのブラックリストがなぜ登録されたのか理由を考える必要がある．例えば，正規サイト

が攻撃者に改ざんされて攻撃コードが埋め込まれた場合，その改ざんサイトがブラックリストに登録されたことでアクセスが遮断されたのであれば，攻撃が未然に防げたためそれ以上の対応は不要であると考えることができる．ところが，アクセスした Web サイトが C2 サーバーであれば，たとえ遮断されていたりサイトが閉鎖したりしていても，すでに感染したマルウェアの活動である可能性が考えられるので，インシデントレスポンスを行う必要がある. これは明確に区別する必要がある. また, 別の事例として 2017 年に WannaCry と呼ばれるランサムウェアが流行したが，特定のハードコーディングされたドメインに Web でアクセスし，応答を受け取ることで動作を停止するキルスイッチと呼ばれる機能を備えていた[18]．キルスイッチへのアクセスは WannaCry の感染端末が存在することを意味するため発見できなければならないが，このアクセスはプロキシサーバーで遮断してはいけない. このような場合も，ブラックリストに登録された意味を明確にする必要がある．

このようにブラックリストを利用する人の立場や求められる対応, 脅威として登録された理由によってその意味合いが変わってくるが，一括りにブラックリストと表現されてしまい様々な情報が区別されずに混在している．特に，CSIRT にとって最適な意味づけは行われていない. 合理的に判断するための情報が事前に与えられれば, 事象を正しく把握して“効果的”な対応が可能となり，作成したマニュアルに基づいて判断することで時間をかけずに“効率的”なトリアージが可能となる．“効果的”・“効率的”な脅威の発見と初動対応の判断を実現するため, 悪性 Web サイト情報に対して CSIRT の役に立つ分類の提案に取り組んだ. この成果を第 3 章にて報告する．

### 2.3.2 事象の分析における課題

インシデントレスポンス工程において，CSIRT はマルウェア感染端末を特定して適切な対処を行うが，重要な情報が含まれる端末や，アクセス権限の高いユーザーが使用する端末が感染した場合は，被害範囲を正確に特定して“効果的”な対策を行うために，フォレンジックなどのより詳細な調査が必要となる場合がある．

例えば，JPCERT/CC によると，BlueTermite といった一連のキャンペーンでは標的型メールによるマルウェア感染から始まり，脆弱性や入手したパスワードを駆使して Active Directory サーバーなどシステムの奥深くまで侵入し，大量の機密情報や個人情報を収集し

ていた[64][65]．この中で，共通的に Emdivi という RAT（Remote Administration Tool）が使用されていた．APT 攻撃のように明確な目的の下で行われる高度かつ執拗な攻撃においてはその被害も甚大であり，侵入経路や影響範囲を特定するため，しばしばフォレンジックを行う．NIST では，フォレンジックを「収集」「検査」「分析」「報告」という 4 つの工程で説明している[66]．インシデントレスポンスにおけるフォレンジックでは，この「検査」工程においてマルウェアの通信先や通信方式などの挙動を解析し，「分析」工程においてプロキシサーバーなどのネットワークログと照合することで影響範囲を調査することがある．

マルウェア解析の手法は主に動的解析と静的解析に分類される[67]．動的解析では，マルウェアを実行して，ファイルやレジストリの操作，ネットワーク通信，API 呼び出しなどを観測する．効率的に解析できる一方，RAT のように攻撃者からの指令を受けて初めて動作するマルウェアを完全に解明する事は難しい．静的解析では，マルウェアの逆アセンブリを取得してコードを読み解く．技術者のスキルと時間を要するため，インシデントレスポンスにおいては特定の挙動に着目して解析することになる．例えば，JPCERT/CC が APT 攻撃のログ分析手法の事例を紹介しているように，プロキシサーバーのログから C2 通信を調査することがある[68]．そのため，接続先のドメインや HTTP リクエストを構築する処理，暗号アルゴリズムや暗号鍵を解析する．JPCERT/CC によると，全ての手口が標的組織ごとに開発されるのではなく，類似の手口や攻撃インフラ（C2 サーバーやマルウェア設置サイト，マルウェアなど）が利用されることも少なくないため，これらの情報が共有されることが望ましいとしている[69]．特に，インシデントにおいてマルウェアの亜種を解析する場合，以前解析したマルウェアの情報が静的解析に利用できることがある．軽微な変更しか行われていないソースコードからコンパイルされたマルウェアの解析においては，難読化が十分でない場合，予備知識がない状況よりも“効率的”に解析が行える．そこで，これから解析するマルウェアにおいて，以前解析したマルウェアの関数に相当するコードを特定する手法の提案に取り組んだ．この成果を第 4 章にて報告する．

# 第 3 章
# CSIRT のための Web ブラックリストの分類の提案

本章では，インシデントレスポンスの「トリアージ」「対応実施」において対応の優先度や対応要否を決定するために必要となる Web ブラックリストの分類について提案する．また，この分類を使用して作成したケーススタディを基に“効果的”・“効率的”にインシデントレスポンスが行えることを示す．

## 3.1 はじめに

CSIRT はインシデントレスポンスにおける「トリアージ」「対応実施」において，脅威情報を調査してインシデント前後のログと照合することで，対応の優先度や対応要否を決定する．脅威情報の調査には時間とスキルを要するが，あらかじめ Web ブラックリストに必要な情報が付与されていれば，この作業を省略できるはずである．つまり，合理的に判断するための情報を事前に与えることで，事象を正しく把握して“効果的”な対応が可能となり，マニュアルに基づいた判断により時間をかけずに“効率的”なトリアージが可能となる．ここでは，Web ブラックリストに付与されるべき CSIRT にとって最適化された情報の定義を提案する．

## 3.2 前提知識

本節では，本論文において必要となる前提知識を示す．

### 3.2.1 ブラックリスト

ブラックリストはアクセスの検知や遮断を目的として悪性 Web サイトの情報を記録したものであり，オープンソースのもの[21][22][23]，製品に組み込まれて提供されるもの[24]，ライセンス契約に基づいて提供されるもの[25]，組織が独自に収集するものがある．ブラックリストに登録する悪性 Web サイトの情報は，ハニーポットやハニークライアント，マルウェア解析，Web サイトからの情報収集，発見者からの申告などによって収集される．

悪性 Web サイトに対するレピュテーションを付与したブラックリストが存在する[26]．また，SNS やショッピングサイトなど，Web サイトのコンテンツのカテゴリで分類しているものがあり，カテゴリの一部として脅威に関する情報が分類されている[24]．

ブラックリストには，URL，ドメイン，IP アドレスが混在して登録される．URL は最も細かい制御であり，同じ Web サイトであってもコンテンツ単位で情報を記録できる．ドメインが記録された場合はそのドメインが提供する Web サイトそのものが悪性と判断される．ドメインは階層構造で定義されるが，悪性情報を含む最大の長さのものが利用される．同じ IP アドレス上に複数のドメインで Web サイトが構成されることもあるが，これらを包括するために IP アドレスを記録する場合がある．

ブラックリスト以外の悪性情報を提供する脅威情報提供サービスがある．機器に設定してログを記録するための情報とは限らなく，データベースとして参照できるサービスや，定期的なレポートとして送られてくるサービスがある[27][28]．ハッシュ値やファイル名，攻撃者のプロファイル情報など，脅威に関連する広い情報が含まれる．

### 3.2.2 攻撃モデル

攻撃者は組織的に活動しており，役割分担がなされている[29]．エクスプロイトキットによるドライブバイダウンロード攻撃では，Web サイトが改ざんされてリダイレクト処理が埋め込まれる．攻撃サイトから攻撃コードがダウンロードされ，攻撃が成功した場合にマ

ルウェア配布サイトからマルウェアをダウンロードして感染する．マルウェアは C2 サーバーから命令を受け取って処理を行う[30]．

また，メールに添付された悪性ファイルを実行してマルウェアに感染する場合がある[31]．文書ファイルのマクロ機能や JavaScript などのスクリプトが添付され，これを実行してマルウェアをダウンロードする場合と，添付された実行ファイルを実行して感染する場合がある．マルウェアに感染すると C2 サーバーから命令を受け取って処理を行うものがある．

### 3.2.3 HTTP と DNS

Web アクセスにおける通信では HTTP と DNS のプロトコルが使用される．DNS によりドメインの名前解決を行い，HTTP によりコンテンツの要求とダウンロードが行われる．どちらもネットワーク層で IP アドレスを用いて通信する．ドメインは DNS のペイロードと HTTP プロトコルの Host ヘッダに現れる．SSL を使用した HTTPS 通信が用いられる場合もある．

### 3.2.4 プロキシサーバーと IPS

プロキシサーバーは DNS による名前解決と HTTP 通信をクライアントの代理で行う．URL，ドメイン，IP アドレスをブラックリストとして登録することで，Web サイトへの通信を制御できる．単純に通信をドロップする，クライアントのアクセスに対してエラー画面を返す，クライアントに対して本当にアクセスしてよいか確認を促す，アクセス自体は許可するがログを記録するといった方法で脅威の防御や発見，監査を行う．HTTPS 通信を中継する際に SSL を一時的に復号して中の HTTP 通信をチェックする機能を備えているシステムもある[32]．

IPS（Intrusion Prevention System）は通信を監視して，シグネチャで定義された特定のパターンに一致したパケットを検知もしくは遮断することができる．DNS の名前解決要求も検査の対象となるため，既知の悪性ドメインの名前解決を検知もしくは遮断するシグネチャを提供する機器がある[33]．

利用者の操作が介在しないマルウェアによるアクセスはバックグラウンドで自動的に行われるが，直接インターネットへアクセスすることは一般的にはファイアウォールで禁止

されているため，このようなアクセスはプロキシサーバーを経由すると想定する．マルウェアの中には端末に設定されたプロキシサーバーを認識せずに直接 DNS サーバーに名前解決を行おうとするものがある．このようなマルウェアについてブラックリストを適用する場合はファイアウォールや DNS サーバーのログを活用する必要がある．本論文ではこのケースについて言及しないが，この場合においても議論は同様となる．

## 3.3 CSIRT におけるブラックリスト利用の課題

「トリアージ」業務において CSIRT は必ずしも正しい判断が下せるとは限らないが，合理的な思考に基づいて対応の要否と優先度を判断する必要がある．

まず，ブラックリストに登録された Web サイトへのアクセスが発生した場合，これが本当にインシデントとして認識すべきかどうか見極める必要がある．例えば，これがマルウェアを配布する Web サイトであった場合，アクセスを遮断していれば脅威は未然に防げたと考えられる．ところが，これが C2 サーバーの場合は，アクセスが遮断されていたとしてもマルウェア感染した端末の存在が疑われるため脅威が残存していると考えられる．ブラックリストにはこのような情報が求められる．

次に，インシデントレスポンスの優先度を決定する．例えば，ブラックリストに記録されたある IP アドレスに対する通信が発生した場合を考える．実はこの IP アドレスに紐づく Web サーバーが 300 サイトあり，そのうちの 1 つだけが改ざん被害を受けていたという場合，このアクセスが危険であるという可能性は低いと判断できる．ところが，ブラックリストがこのような情報を含んでいない場合，優先度が正しく判断できない．

ブラックリストにおけるレピュテーションを数値で表現するものがあるが，その根拠は開示されないため，この値に基づいた判断は合理的とはいえない．ロッキードマーティン社が提案しているサイバーキルチェーンによる攻撃の進行度によって重大さが表現されることがある[115]．ただし，アクセスしたドメインに付与された情報がサイバーキルチェーンにおける初期工程の「配送」段階を示していても，それだけで優先度が低いとは判断し難い．マルウェア名や攻撃者像を提供する脅威情報サービスがある[35]．脅威という視点で情報が分類されているが，必ずしもインシデントレスポンスという視点で十分に活用できる

情報ではない.

「対応実施」においても同様の問題が発生する．何でも遮断するよう設定して，さらにアクセスを遮断するたびに CSIRT がインシデントレスポンスを行うといった業務プロセスを想定することは現実的ではない．すでに感染したマルウェアによる活動であれば，例えアクセスが遮断されていても CSIRT はインシデントレスポンスを行うべきであるし，すでに修正された改ざんサイトへのアクセスが成功している場合は必ずしも CSIRT はインシデントレスポンスを行う必要があるとは限らない．また，共用サービスや侵害された Web サイトを遮断する場合は，業務通信を誤遮断するリスクを覚悟しなければならない.

このように，CSIRT が合理的な思考に基づいて対応の要否や優先度を判断するための情報が不足していることがいえる.

## 3.4 提案

### 3.4.1 スコープ

組織における Web 利用者に対する悪性 Web サイト情報のブラックリストを対象とし，CSIRT の主な業務であるインシデントレスポンスにおいて活用することを想定する．本論文における「悪性 Web サイト情報」とは，Web サービス利用者に対して脅威と考えられるアクセス先の「URL」「ドメイン」「IP アドレス」で定義する.「ドメイン」には FQDN だけでなく, 部分一致で利用する上位のドメインも含まれる. これらはプロキシサーバー, IPS に設定され，該当の通信先に対する HTTP と DNS 通信を検知する.

ワームによる攻撃元やスパムメールの発信源など, 能動的な攻撃は対象としない. また，SNS や娯楽サイトへのアクセスなど，正規の用途が考えられ組織のポリシーによってアクセスの可否判断が変わるものや公序良俗に反するサイトも対象としない．マルウェアのハッシュ値やファイル名，端末においてマルウェアが操作するレジストリなど端末情報，攻撃者が取得した SSL 証明書のハッシュ値も脅威情報として表現されることがあるが，本論文では対象としない[19][20].

本節では CSIRT におけるインシデントレスポンスの判断に役立つ分類を提案するが，その判断自体は組織のポリシーに依存するため規定しない．すでに従来の分類で情報が付与

されている場合，この利用を否定するものではないものとする．

### 3.4.2 提案分類

本項では，ブラックリストにおける悪性 Web サイト情報が CSIRT にとって有益となるために，表 3-1 のとおり分類することを提案する．これは過去 1 年間の実績においてインシデント判定を求められた悪性情報の事例をもとに作成したものである．(1)で Web サイトの所有者で分類する．(2)ではコンテンツの種別について重複選択を許してラベルを付与し，(3)で現在の状態について分類する．さらに，この情報を最後に調査した日時(4)を記録する．インシデントレスポンスでは脅威の種類によって対応方針を決定する．その際に，情報の精度と鮮度を踏まえて，より確からしいインシデントから対応する．脅威の種類は(2)に，精度は(1)，鮮度は(3)(4)にそれぞれ対応する．悪性 Web サイト情報は「URL」「ドメイン」「IP アドレス」のいずれかの形式で登録されるが，たとえ同じ脅威を示しているものであっても，どの形式で登録するかによって分類が異なる場合があることに注意する．

表 3-1　提案する Web ブラックリストの分類

Table 3-1　Proposed classification of web blacklist.

| 項目 | 分類 | |
|---|---|---|
| (1)　所有者 | 1 | 侵害 |
| （重複不可） | 2 | 共有型 |
| | 3 | 攻撃者 |
| | 4 | 不明 |
| (2)　コンテンツ | a | リダイレクト |
| （重複可） | b | エクスプロイト |
| | c | 悪性ファイル配布 |
| | d | C2 |
| | e | キルスイッチ |
| | f | 権威 DNS サーバー |
| | g | DNS シンクホール |
| | h | ソーシャルエンジニアリング |
| | i | 正規コンテンツ |
| | j | 不明・誤登録 |
| (3)　現在の状態 | 1 | 現在も悪性 |
| （重複不可） | 2 | 現在は無害 |
| | 3 | 不明 |
| (4)　最終確認 | 日時 | |

### 3.4.3 CSIRT におけるブラックリストの活用

CSIRT ではインシデントを「トリアージ」「事象の分析」「対応計画」「対応実施」というプロセスで行う[34].「トリアージ」では，検知した全てのインシデントが対応すべきインシデントとは限らないため，CSIRT の資源を有効的に活用するようインシデントレスポンスの要否や優先度を判断する.「対応実施」では，発生したインシデントの収束に加えて再発防止に向けた改善活動を行う.

CSIRT において悪性 Web サイト情報のブラックリストは主に 2 つの用途で使用される.「トリアージ」では，プロキシサーバーや IPS で発見したインシデントレスポンスの優先度を決定するために, ブラックリストに紐づく悪性 Web サイト情報や脅威情報を活用する.「対応実施」では，再発防止や新たな脅威の発見を目的として，プロキシサーバーや IPS に悪性 Web サイト情報をブラックリストとして設定する．その際に，アクセスを遮断すべきかそうでないか，CSIRT がその事象を把握するべきかそうでないか，ということを考慮する.

### 3.4.4 提案分類の詳細

各項目の定義と具体例，CSIRT においてこれを明示的に区別する理由について述べる.

#### (1) 所有者（重複不可）

悪性情報を所有者によって分類する．侵害，共有型，攻撃者，不明の 4 種類に分類され，いずれか 1 つが選択される.

(1)-1 侵害

悪意を持たない一般の Web サイトが攻撃者によって侵害された，もしくは，元々提供しているサービスが悪用された結果，悪性情報として記録されるようになったことを示す．元から存在するページを改ざんして悪性コードを埋め込んだ，新たにファイルとしてマルウェアを設置した場合や，C2 サーバーとしてのプロセスが動作し感染端末に命令を送信するようになった場合などがある.

当該 Web サイトは業務で使用している可能性があるので，脅威情報として記録された場

合であっても，ドメインごと遮断すべきか，悪性と認められたコンテンツだけ遮断すべきか，すでに修正されたと考え遮断しないか，といった対応の判断は CSIRT に委ねられる．また，侵害サイトは一般的に修正，改善されることが期待されるため，ブラックリスト提供者は定期的な見直しによる除外が求められるが，実際はブラックリストから除外されずに残り続けることも多い．そのため，ブラックリストの登録時期が極端に古い場合や対処済みというニュースリリースが公開されるなど，現在は影響がない可能性が高いという判断も期待でき，登録時期を考慮して現在の侵害の可能性を分析することもできる．

不特定多数のユーザーがファイルを公開できるアップローダーといったサービスがあり，ここにマルウェアがアップロードされる場合もある．広告サービスや短縮 URL サービスも悪性サイトへの転送に使用される．元々提供しているサービスを悪用したという意味でここに分類する．

(1)-2 共有型

悪性として記録された情報に対して悪性とは限らない他のリソースが紐づいてしまう場合がある．名前ベースのバーチャルホストでは，同一の IP アドレスに対する HTTP リクエストの中で Host ヘッダに記載された FQDN により応答するコンテンツを識別する．Web サイトを公開するためのディレクトリを提供するサービスでは，同一のドメインに対して複数の Web サイト所有者を URI のパスで識別する．

このような状況において悪性情報に一致するアクセスが発生したとしても，本当に悪性であるリソースにアクセスしたかどうかは不明である．そのためトリアージの優先度は他よりも低くなる．このアクセスを遮断する場合の業務影響の予測は難しく，慎重に実施する必要がある．

ファイル共有アップローダーやブログサービス，匿名通信に使用する Tor ノードの IP アドレス，短縮 URL サービス，ダイナミック DNS サービスで提供される親ドメイン，CDN や IaaS 型クラウドサービスが内部的に割り当てるドメインもこちらに分類される．

(1)-3 攻撃者

攻撃者が自前で調達した Web サイトやドメイン，攻撃者が設置した悪性ファイルの URL

はこちらに分類する．例えば，攻撃者がドメインを取得する際に同一のメールアドレス使いまわすことがあり，whois データベースを参照することでドメインが悪性と判断できる場合がある．

悪意を持たない一般の Web サイトを攻撃者によって模倣された場合，または一見して実在しそうな組織を装ったコンテンツが設置された場合もこちらに分類する．この分類におけるアクセスは問題なく遮断できることが期待できる．

(1)-4 不明

悪性 Web サイト情報を調査してもその素性が全く分からない場合はこちらに分類する．

**(2) コンテンツ（重複可）**

Web サイトから応答されるコンテンツによって分類する．ドライブバイダウンロード攻撃の進行度によって，リダイレクト，エクスプロイト，悪性ファイル配布，C2 という 4 種類に分類する．また，過去の実績から他にもキルスイッチ，権威 DNS サーバー，DNS シンクホール，ソーシャルエンジニアリング，正規コンテンツ，不明・誤登録という 5 種類の分類が事例として必要であることが分かった．複数の役割を持つ場合があるので，適合するもの全てを選択する．

(2)-a リダイレクト

悪性サイトへの転送機能を持つ Web ページで，エクスプロイトキットにおける初期の攻撃段階としてよく使用される．リダイレクト自体は直接的な被害を受けるものではないため，後続のアクセスが失敗していればインシデントレスポンスが不要であると判断しても良い．

(2)-b エクスプロイト

端末の脆弱性を悪用して制御を奪う悪性コードを返す Web ページである．この通信は攻撃であるため，パッチが適用されていないなど，一定の条件を満たした場合に侵害される．

影響を調査するためには，被害端末を特定して端末を調査する必要があるため，区別する必要がある．

(2)-c 悪性ファイル配布

マルウェアを呼び込むためのダウンローダー，マルウェアを生成するドロッパー，マルウェア，アドウェアを返すものを分類する．

(2)-d C2

すでに端末に感染したマルウェアが外部からコマンドを受信するためにアクセスする Web ページである．C&C（Command and Control）とも呼ばれる．HTTP の上に独自のプロトコルを構成するものや，ブログ記事のように一見して Web コンテンツに見えるが実はコマンドとして機能するものも存在する． C2 通信は必ずしも HTTP とは限らず，DNS の TXT レコードを用いた DNS トンネリングによる通信の場合もある．

感染端末の存在が疑われるためアクセスが遮断されていた場合であってもインシデントレスポンスが求められる．一方で，意図的にアクセスした場合には影響がある可能性は低い．ブラウザでアクセスすると，同時に favicon.ico へのアクセスが出る．CSIRT が調査目的や興味本位でアクセスする場合，このような挙動を示す．接続先が C2 サーバーの場合，人手であることが推測されれば影響ないと判断できる場合がある．

(2)-e キルスイッチ

マルウェアがサンドボックスにおける動的解析環境を検知するために行うアクセス先である．

動的解析は存在しないドメインに対しても応答を返す機能を持つものもあるため，キルスイッチからの応答を受信して動作を停止するため，アクセスを遮断してはいけない．

(2)-f 権威 DNS サーバー

ドメインではなく，名前解決を行う権威 DNS サーバーが悪性と判断される分類である．

悪性ドメインは匿名で購入可能なレジストラで取得されるあることがあり，このようなサービスを利用して取得した場合の名前解決は特定の権威DNSサーバーによる名前解決を経由するが，この通信をブラックリストとして検知することがある．

(2)-g DNS シンクホール

テイクダウンされた, もしくは有効期限が切れた悪性サイトのドメインをセキュリティ研究者が取得し，アクセスしても問題がない IP アドレスを応答するように設定したものである．すでに無害化したドメインである．

(2)-h ソーシャルエンジニアリング

機密情報を送信させるフィッシングサイトや偽ショッピングサイト, パソコンのトラブル解決を装って偽のサポートツールのインストールを誘導するサイトなど，ソーシャルエンジニアリングとしての脅威を分類する．

この分類における影響は Web サービス利用者の操作に起因するため，被害者へのヒアリングが必要となる．

(2)-i 正規コンテンツ

侵害されていない一般の Web サイトがブラックリストに登録されることがある．マルウェアがインターネットによる Web アクセスが可能かどうか疎通確認を行う場合や，外部アクセスを行うグローバル IP アドレスを調査するため環境情報表示サイトにアクセスする場合がある．

この通信自体は無害であるため積極的に利用しないが, マルウェアの活動の兆候の可能性として考えることもできる．

(2)-j 不明・誤登録

ブラックリストに登録した根拠が全く不明なものが分類される．明確な根拠がないため，誤登録の可能性も考えられる．この分類は第三者が評価を行う場合のみ使用する．

**(3) 現在の状態（重複不可）**

現在の状態によって分類する．現在も悪性，現在は無害，不明の 3 種類に分類され，いずれか 1 つが選択される．

(3)-1 現在も悪性

悪性情報が登録, もしくは最後に再調査された時点で意図したとおり悪性であったことを示す分類である．

悪性サイトは, 攻撃者が活動を停止した場合や, 所有者が改ざんを修正した場合に無害となる．ただし，攻撃者が活動を再開したり，再発防止策が不十分で再び侵害されたりすることによって再び悪性サイトとなる場合がある．ここに分類された情報は必ずしも正しいとは限らないが，最後に調査された時期と併せた場合にどこまでを正しいと想定するか考えておくために必要な情報である．乱数生成器によって作成した大量のドメインの名前解決を行うことで検知の回避を試みる DGA（Domain Generation Algorithm）マルウェアというものもあるが，大量生成されたドメインのうち実際に名前解決されるものもここに分類される．また，悪性コードを配信してしまう広告があるが，根本的な対策は困難であり今後も悪性挙動を示す可能性があることから，ここに分類する．

(3)-2 現在は無害

現在は無害であると報告されているものを指す．テイクダウンされたサーバーや DNS シンクホール，ドメインの有効期限が切れているものはここに分類される．また，侵害サイトにおいて事象が修正されたと考えられるものはここに分類される．必ずしも適切な再発防止策がなされているとは限らないため，この分類を信用するかどうかは組織のポリシーに委ねられる．

(3)-3 不明

悪性や無害と言える根拠がないものがここに分類される. サービスのポートが応答しないもの，サーバーとしては動作しているが何も応答しないもの，無害なコンテンツを応答す

るものである．一見して無害に分類しても良いように思えるが，アクセス元の IP アドレス，User-Agent や Referer のような HTTP ヘッダなど，特定の条件に一致した場合にのみ悪性コンテンツを応答するものがあるため，不明として分類する．DGA で生成されたドメインで名前解決されないものもここに分類される．この分類における判断は組織のポリシーに委ねられる．

**(4) 最終確認**

最後に確認した日時を記載する項目である．これは現在の状態における評価結果の確からしさを判断するための情報として利用される．

## 3.5 調査・評価

### 3.5.1 公開ブラックリストにおける登録情報

本研究の調査で入手した全 38 種類のブラックリストについて，その登録情報の項目を調査した．結果を表 3-2 に示す．各ブラックリストについて，その項目を，記載あり，一部記載あり，記載なしの 3 つで評価した．また，その詳細を表 3-3 に示す．「5.1 公開ブラックリストにおける登録情報」において，公開されている 38 種類のブラックリストについて調査した．各項目について，○は記載あり，△は一部記載あり，×は記載なしを意味している．

表 3-2 公開ブラックリストにおける登録情報

Table 3-2 Registered information in public blacklists.

| 項目 | 記載あり | 一部あり | 記載なし |
|---|---|---|---|
| (1) 所有者 | 0 | 4 | 34 |
| (2) コンテンツ | 0 | 24 | 14 |
| (3) 現在の状態 | 16 | 0 | 22 |
| (4) 最終確認日 | 22 | 1 | 15 |

表 3-3　公開ブラックリストにおける登録情報の詳細

Table 3-3　Details of registered information in public blacklists.

| No. | ブラックリスト | 所有者 | コンテンツ | 現在の状態 | 最終確認 | 備考（△となった理由） |
|---|---|---|---|---|---|---|
| 1 | Shalla's Blacklists | × | × | × | × | |
| 2 | AVGThreatLabs | × | × | ○ | ○ | |
| 3 | Bambenek Consulting | × | △ | ○ | ○ | C2 ドメインと DGA ドメインが区別される. |
| 4 | BitDefender | × | × | × | × | |
| 5 | CyberCrime | × | △ | × | ○ | マルウェア名で分類される. |
| 6 | security-research | × | × | ○ | ○ | |
| 7 | DNS-BH | × | △ | ○ | ○ | カテゴリの中でフィッシングなどが分類される. |
| 8 | Fortinet | × | △ | × | × | カテゴリの中でフィッシングなどが分類される. |
| 9 | GoogleSafeBrowsing | × | △ | × | ○ | 悪性と判断した簡単な理由が記載されている. |
| 10 | HijackedUrls | × | △ | ○ | ○ | リダイレクトサイトであることが区別される. |
| 11 | Malc0de | × | × | × | △ | 過去 30 日以内に確認されたことが分かる. |
| 12 | MalwareDomainList | × | △ | × | ○ | マルウェアの種類で分類される. |
| 13 | OpenPhish | × | × | ○ | × | |
| 14 | PhishTank | × | × | ○ | ○ | |
| 15 | Quttera | × | × | ○ | ○ | |
| 16 | Ransomware Tracker | × | △ | ○ | ○ | ランサムウェアに限定されているが, Distribution Sites, Botnet C&Cs, Payment |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | Sites が区別される. |
| 17 | SCUMWARE | × | △ | × | ○ | マルウェア名で分類される. |
| 18 | Spam404 | × | △ | × | ○ | カテゴリの中でフィッシングなどが分類される. |
| 19 | ThreatCrowd | △ | △ | × | × | マルウェア名で分類される．Passive DNS 情報を活用してドメイン，IP アドレスなどの要素をグラフ表示する機能を持ち，これを確認することで共有サーバーの存在が判明する場合がある. |
| 20 | ThreatLog | × | △ | ○ | ○ | カテゴリの中でフィッシングなどが分類される. |
| 21 | URLVir | × | × | ○ | ○ | |
| 22 | VXVault | × | × | × | ○ | |
| 23 | WebSecurityGuard | × | △ | × | × | カテゴリの中でフィッシングなどが分類される．不特定のユーザーによるコンテンツの評価と「Yes/No」といったレピュテーションを含む. |
| 24 | ZeroCERT | △ | △ | × | ○ | Web サイトの状態を表す「State code」の 1 項目として「compromised」という情報を含む．カテゴリの中でフィッシングやマルウェア名などが付与される場合がある. |
| 25 | ZeuS Tracker | △ | △ | ○ | ○ | ZeuS に限定されているが，「Level」において「Hacked webserver」「Free hosting service」という情報を含む. |
| 26 | Artists Against 419 | × | × | ○ | ○ | |
| 27 | CLEAN-MX | × | △ | ○ | ○ | マルウェア名で分類される. |
| 28 | Certly Guard | × | △ | × | × | カテゴリの中でフィッシングなどが分 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | | 類される. |
| 29 | hpHosts File | △ | △ | × | × | カテゴリの中に「hijack sites」という情報を含む. |
| 30 | Malware Patrol | × | × | × | × | |
| 31 | MalwareURL List | × | △ | × | × | マルウェアの種類で分類される. |
| 32 | Site Safety Center | × | △ | × | × | カテゴリの中でフィッシングなどが分類される. |
| 33 | Safe Web | × | △ | × | × | カテゴリの中でフィッシングなどが分類される. |
| 34 | Web Inspector | × | △ | ○ | ○ | カテゴリの中でフィッシングやマルウェアの種類などが分類される. |
| 35 | VirusDesk | × | △ | × | × | 悪性と判断した簡単な理由が記載されている. |
| 36 | Gred | × | △ | × | × | カテゴリの中でフィッシングなどが分類される. |
| 37 | SiteCheck | × | × | ○ | ○ | |
| 38 | MESD blacklists | × | × | × | × | |

(1) 所有者に関する情報は多くのブラックリストにおいて記載がされていなかったが，4 つのブラックリストにおいては一部記載がされていた．ZeroCERT では Web サイトの状態を表す「State code」の 1 項目として「compromised」という情報が区別されていた[36]．ZeuS Tracker では「Level」という項目に「Hacked webserver」「Free hosting service」という情報が区別されていた[37]．ThreatCrowd では直接的な項目を含まないが，Passive DNS 情報を活用してドメイン，IP アドレスなどの要素をグラフ表示する機能を持ち，これを確認することで共有サーバーの存在が判明する場合がある．いずれのブラックリストにおいても，独立した項目として所有者の有無が区別されていたものではなく，カテゴリの分類の 1 つとして侵害サイトや共有サーバーが選択できるという状態であった[38]．

(2) コンテンツに関する情報は 6 割以上のブラックリストにおいて一部記載されていた．HijackedUrls は悪性サイトにリダイレクトする Web サイトの情報を収集しており，リダイレクト先の URL が記載されていた[39]．Ransomware Tracker はランサムウェアに関する Web サイトの情報を収集しており，「Distribution Sites」「Botnet C&Cs」「Payment Sites」が区別されていた[40]．WebSecurityGuard では，不特定のユーザーによるコンテンツの評価と「Yes/No」といったレピュテーションが記載されていた[41]．その他のブラックリストにおいてもエクスプロイトと悪性ファイル配布が区別されているものが多かったが，いずれも分類項目は一部に留まっていた．

(3) 現在の状態については約 4 割，(4) 最終確認は約 6 割のブラックリストが記載していた．Malc0de では 30 日以内に登録されたことが分かるが，具体的な日付に関する情報がないため一部記載ありに集計した[42]．

### 3.5.2 インシデント判定が求められた悪性情報の分析

実際にインシデント判定が求められた 400 件の悪性情報を対象として，手動で調査して分類した．この分類は SOC サービスを提供する中で実務として行われたものであり，サンプリングした 100 組織において，2017 年 9 月 13 日から 2017 年 11 月 23 日までの期間に検知した悪性情報を調査したものである．インシデント判定が求められた事象の検知元を表 3-4 に，インシデント判定の結果，対応が必要と判断された 31 件の事象に関連するマルウェアを表 3-5 に示す．

表 3-4　インシデント判定が求められた悪性情報の入手元

Table 3-4　Devices that detected malicious activity to judge incident.

| 検知機器 | 検知手法 | 件数 |
|---|---|---|
| Proxy | ブラックリストとの照合 | 82 |
| Firewall | ブラックリストとの照合 | 11 |
| IPS | シグネチャ検知 | 207 |
| Sandbox | 動的解析の通信先 | 10 |
| SIEM | 継続的な通信 | 54 |
| （上記全ての相関分析） | DGA マルウェアの名前解決 | 12 |
| | その他 | 24 |

表 3-5　インシデントに関連するマルウェア

Table 3-5　Malware related to the incident.

| マルウェア名 | 件数 |
|---|---|
| Locky | 6 |
| Ramnit | 2 |
| Hancitor | 2 |
| Loki | 1 |
| Daserf | 1 |
| WannaCry | 1 |
| Ursnif | 1 |
| 不明 | 17 |

調査は以下の手法で行った. 悪性情報は通信が発生した時間と調査を行った時間の差が大きいと情報が不正確となるため，実通信がブラックリストにマッチした時点で速やかに調査を開始した.

- インターネット上の情報を検索する

検索エンジンで悪性情報を検索する. 悪性情報に関する調査結果は公開されていることも多い．悪性情報そのものだけでなく，これをホスティングしている事業者について調査することで Web サイトの所有者について分かる場合もある.

- whois 情報や名前解決結果を確認する

whois 情報やドメインの名前解決，ドメインの逆引きから Web サイトの所有者について調査を行う．Passive DNS を活用すると，過去の時点での名前解決結果や，IP アドレスに紐づく他のドメインの情報が得られ，これだけで悪性と判断できる場合や，IaaS 型クラウドサービスで多くの利用者が紐づいていて悪性と言える根拠が少ないことが分かる場合もある.

- 脅威情報データベースを確認する

レピュテーションや脅威情報データベースの登録情報，アンチウイルスの検知名，Web サイトのカテゴリを参照する.

- 実際にアクセスする

動的解析サンドボックスから実際に Web サイトにアクセスを行い，応答されるコンテンツや悪性挙動の有無を調査する．何度もアクセスすることで攻撃者に調査であることが気づかれ挙動を変えるものがあるため，大量の IP アドレスを保有し 1 分毎に送信元がユニークに変わる環境を使用した．ただし，水飲み場型攻撃でアクセス元などの条件が一致した場合のみ攻撃を受ける場合や，滅多に攻撃が発動しない場合に無害に見えてしまうことがある．必要に応じて Web Archive を利用し，ブラックリストに登録された当時の Web サイトを調査する.

- パケットキャプチャを確認する

ブラックリストにマッチした実通信のパケットキャプチャが取得できている場合は活用する．水飲み場型攻撃のように特定の環境でしか発動しない脅威も調査できる可能性があ

る．

- マルウェアや悪性スクリプトを解析する

必要に応じてマルウェア解析や Web サイトに埋め込まれた悪性スクリプトの解析を行う．マルウェアの接続先や悪性スクリプトのリダイレクト先が判明する．

ブラックリストは，製品の一部として提供されるもの，インテリジェンスサービスとして提供されるもの，無償で公開されているもの，組織の通信ログから独自に収集したものを使用している．この悪性情報を分類したところ，この内訳は表 3-6 に示すとおりとなった．

表 3-6　評価結果

Table 3-6　Evaluation result.

| | 項目 | URL | ドメイン | IP | 合計 |
|---|---|---|---|---|---|
| | | 35 | 348 | 17 | 400 |
| (1) | 1 | 16 | 132 | 0 | 148 |
| | 2 | 2 | 18 | 17 | 37 |
| | 3 | 5 | 80 | 0 | 85 |
| | 4 | 12 | 118 | 0 | 130 |
| (2) | a | 2 | 9 | 0 | 11 |
| | b | 0 | 4 | 0 | 4 |
| | c | 16 | 105 | 2 | 123 |
| | d | 6 | 41 | 0 | 47 |
| | e | 0 | 1 | 0 | 1 |
| | f | 0 | 0 | 0 | 1 |
| | g | 0 | 6 | 1 | 7 |
| | h | 4 | 88 | 1 | 93 |
| | i | 5 | 2 | 0 | 7 |
| | j | 3 | 103 | 13 | 119 |
| (3) | 1 | 10 | 78 | 1 | 89 |
| | 2 | 15 | 95 | 4 | 114 |
| | 3 | 10 | 175 | 12 | 197 |

この結果において，「(1) 所有者」の 67.5%，「(2) コンテンツ」の 70.2%，「(3) 現在の状態」の 50.7% が「不明」以外に分類され，ブラックリスト作成者が本定義に従って分類することが一定の割合で可能であることが示せた．今回はインシデント判定が求められてから第三者の立場として悪性情報を評価しているが，ブラックリスト作成者本人であればリアルタイムかつ登録した理由が明確であるため，より高い割合となる可能性がある．

今回の調査において，ブラックリストで検知した脅威情報で登録理由が不明なものは「(2)-j 不明・誤登録」「(3)-3 不明」に分類したが，ブラックリストの登録者であれば理由が分かると考えられる．

### 3.5.3 CSIRT における効果

本分類は，CSIRT の「トリアージ」におけるインシデントレスポンスの要否や優先度の決定と，「対応実施」におけるプロキシサーバーや IPS へのブラックリスト設定の判断のために使用することができる．この分類に紐づく根拠に基づいた合理的な判断により“効果的”な対応と，策定したマニュアルに従った迅速な判断により“効率的”な対応を実現することができる．マニュアルは組織のポリシーに従って策定する必要があるが，一例として 3.7 節でケーススタディを紹介し，これを基に“効果的”・“効率的”な対応が行えることを示す．ここでは比較的シンプルな基準でインシデントレスポンスを行う CSIRT を想定している．

例えば，このマニュアルを想定して 400 件のインシデントに対応した場合，281 件（70.3%）のインシデントは「不明・誤登録」以外に分類され，CSIRT に対して論理的な思考に基づいた明確な対応方針を即座に与えられる．

今回は 400 件の悪性情報を全て調査する必要があったため「最終確認」は評価時点での日時となったが，実際はブラックリスト作成時刻となるため情報が古くなる可能性がある．ドライブバイダウンロード攻撃における悪性サイトの 80%は生存期間が 170 日以下であると報告されている[44]．これをどのように捉えるかは組織のポリシーに従うところであるが，「リダイレクト」「エクスプロイト」「悪性ファイル配布」「C2」に関して悪性と判断された場合，半年間は悪性と考えた方が良いと捉えることができる．ただし「C2」に関しては，所有者が「攻撃者」の場合は感染端末が疑われるため最終確認に関係なく対応すべきである．

「ソーシャルエンジニアリング」に関して，JPCERT/CC は国内外へのフィッシングサイトの通知を行っているが，97%は 10 営業日以内に対処されたと報告しており，生存期間は非常に短いことが分かる[45]．ただし，実際にアクセスして比較的容易に確かめられるため，最終確認に頼らなくても良い．

「キルスイッチ」「DNS シンクホール」「正規コンテンツ」は一般的に変化しないため，運用上は最終確認に関係なく現在の情報として活用する．逆に「権威 DNS サーバー」は匿名レジストラで取得したドメインなど，これ自体が必ずしも悪性であることを示すものではないため，最終確認に関係なくその都度調査を行う必要がある．

## 3.6 関連研究

本節では，ブラックリストの分類に関連する標準技術，関連研究について示す．

### 3.6.1 標準技術

サイバー攻撃活動における脅威を構造的に表現するために策定された STIX という記述形式がある[43]．バージョン 1 では脅威情報を Campaigns, Threat_Actors, TTPs, Indicators, Observables, Incidents, Courses_Of_Action, Exploit_Targets といった構成で表現するが，「(1) 所有者」において侵害サイトであることを明確に示す項目がない．「(3) 現在の状態」を示すものとしては「Campaigns: Status」が近いが，これは攻撃者の活動を示している．すでに活動が終了したキャンペーンであっても Web サイトにマルウェアが残っている場合があるため，CSIRT の活動に対して直接的には結びつかない．また，STIX において脅威はサイバーキルチェーンの 7 段階モデルで表現されることがあるが，攻撃の初期段階だからといってインシデントレスポンスが不要と考えることもできない．

脅威情報を共有するための基盤として開発された MISP[2]というソフトウェアがある[46]．脅威情報に対してラベルを付与することができるが，どのようにラベルを定義して脅威を表現すべきかといった規定や主張はない．ただし，Taxonomy (分類法) という機能があり，デフォルトで用意されている 30 種類を含む様々な既存のフレームワークをインポートして

2 Malware Information Sharing Platform.

利用することができ，この中に脅威の種別や状態を表現するものがある．SANS はマルウェアの種類を分類している[61]．ENISA[3]は物理的な脅威や法的リスクを含めたビジネス上の脅威を表現している[47]．eCSIRT はマルウェアの種類や侵入方法などを定義している[48]．FIRST や CIRCL[4]はインシデントの種別を定義している[49][50]．NATO は情報の信頼度を定義している[62]．また，出典元の記載がないが adversary というフレームワークを利用しており，インフラの所有者やその状態について定義している．これは「(1) 所有者」と「(3) 現在の状態」に近いが，悪性 Web サイト情報に対紐づく内容ではなく，攻撃者が利用するインフラについて紐づく情報である．

### 3.6.2 関連研究

ブラックリストを評価する研究として，Kührer らはブラックリストを収集するシステムを構築し，49 種類のブラックリストを提供するサイトから 41 万件以上のユニークな URL を収集してその共通度合いや IP アドレスを用いた関係性を報告している[51]．さらに，マルウェアブラックリストの有効性を評価するため，非使用（パーク）ドメインとシンクホールという分類で 19 種類のブラックリストを分析している[52]．Sheng らは，191 個のフィッシング URL を調査し，フィッシングサイトがブラックリストに登録されるまでの時間を調査している[53]．これらは CSIRT にとって有用かどうかという視点では調査していない．

マルウェア関連ドメインを検出する研究として，Antonakakis らは TLD サーバーおよび大規模な機関で動作するシステムを構築し，IP レピュテーション等と組み合わせることでマルウェア関連ドメインを早期に検出する機能を提案している[54]．さらに，クラスタリングを用いて C2 通信に用いられる DGA ドメインを検出する手法を提案している[55]．Bilge らは，パッシブ DNS を用いることにより判明した悪性ドメインを調査している[56]．Rahbarinia らは，無効化したドメインを発見してシンクホール，パーキング，NX-Domain Rewriting の 3 種類に分類する SinkMiner というシステムを提案している[57]．

Sinha らは，スパムメールの送信元ブラックリストを調査するため，レピュテーションベースによる複数のブラックリストの有効性を調査している[58]．Rossow らは，スパムメー

3 European Network and Information Security Agency.
4 Computer Incident Response Center Luxembourg.

ルの送信元の IP アドレスの評価において，ブラックリストとホワイトリストの両方に登録されたものについて調査している[59]．Thomas らは，ソーシャルネットワーク等の Web サービスに投稿されるコメントに含まれる URL がスパムかどうかを判定するシステムを提案している[60]．

## 3.7 ケーススタディ

本提案における分類を用いたインシデントレスポンスの例をケーススタディとして示す．表 3-7 はユーザー企業においてブラックリストに付与されたラベルから比較的シンプルな基準でインシデントレスポンスの判断を行いたい CSIRT を想定した場合の対応判断マニュアルを例として示したものである．このマニュアルにおいて「対応する」という文言は，トリアージにおいては端末や Web アクセス利用者を特定してヒアリングやウイルススキャン，再インストールなどの処置を行うことを意味し，対応実施においてはプロキシサーバー等に遮断設定を行うことを意味する．

実際に発生し得る代表的なインシデントを想定し，このマニュアルに従って合理的な判断が行えることをケーススタディとして示す．この例では便宜上，最初のアラートを検知した日時を「2017 年 11 月 23 日 12:00」として説明する．

(1) 被害の可能性が低い例

被害の可能性が低く，一般的には対応が不要であると考えられるインシデントレスポンスの例を 2 つ示す．

(1-1) DNS 名前解決要求を IPS で検知したことをトリガーにトリアージを開始した．このドメインは「コンテンツ：リダイレクト」「現在の状態：現在も悪性」「所有者：侵害」「最終確認：2015 年 6 月 8 日 15:38」に分類されている．CSIRT はプロキシサーバーのログからこの悪性ドメインにアクセスした利用者の IP アドレスを特定したが，この IP アドレスからはそれ以降のアクセスは行われていなかった．CSIRT が調べたところ，このサイトは日本の企業が公開している公式サイトであったが，

登録された日時が 2 年以上前と古く，すでに修正されているためリダイレクトはされなかったと考えられる．利用者は改ざんサイトへのアクセスは行ったものの，攻撃サイトへのリダイレクトは行われず特に被害は受けなかったためクローズした．

(1-2) プロキシサーバーで悪性ドメインへの通信を検知した．このドメインは「コンテンツ：C2」「現在の状態：悪性」「所有者：侵害」「最終確認：2017 年 5 月 27 日 20:19」に分類されている．CSIRT がこのドメインについて調査を行ったところ，このサイトは誰でも無料でアカウントが作成できるブログサイトであったが，このサイトに関する追加の脅威情報は特に見当たらなかった．このサイトのどこかに悪性コンテンツが埋め込まれている，もしくは特定のブログ記事にコマンドを埋め込んでボットを操作するマルウェアに感染している可能性があるが，通常のブログ記事を閲覧しただけの状況も十分に考えられ，単純にこのドメインにアクセスしたという事実だけで影響があると判断するには材料が乏しい．またこの前後に関連するアクセスは存在しなかったため，インシデントレスポンスとしてはクローズして良いと判断した．

(2) 被害が疑われる例

被害がほぼ確実に発生している，もしくは被害を受けている可能性があり，何らかの対応が必要であると考えられるインシデントレスポンスの例を 3 つ示す．

(2-1) プロキシサーバーで悪性 URL へのアクセスを検知してトリアージを開始した．この悪性 URL は「コンテンツ：悪性ファイル配布」「現在の状態：現在も悪性」「所有者：不明」「最終確認：2017 年 11 月 9 日 6:35」に分類されている．この悪性ファイルはダウンロードが成功しており端末上で動作している可能性があるため，利用者を特定してマルウェア感染を調査する．

(2-2) DNS 名前解決要求を IPS で検知したことをトリガーにインシデントレスポンスを開始した．この悪性ドメインの分類は「コンテンツ：C2，DNS シンクホール」「現在の状態：現在は無害」「所有者：攻撃者」「最終確認：2017 年 8 月 19 日」であった．また，このサーバーに対して 5 分毎に定期的にアクセスを行い始めた．通信先は現在

C2 としての役割を果たしていないものの，感染端末が潜んでいることが強く疑われたためインシデントレスポンスを開始した．

(2-3) DNS 名前解決要求を IPS で検知したことをトリガーにインシデントレスポンスを開始した．この悪性ドメインの分類は「コンテンツ：正規コンテンツ」「現在の状態：現在は無害」「所有者：不明」「最終確認：2016 年 4 月 8 日 19:03」であった．CSIRT が調べたところ，これはアクセスすると自分の IP アドレスを表示してくれる無害なサイトであった．ところが，前後のプロキシサーバーのログを調べてみると幾つかのドメインに対するアクセスが含まれていたため，脅威情報を検索エンジンで調べた．これは C2 であることが分かったため，すでにマルウェアに感染していると考えてインシデントレスポンスを行った．IPS が検知したドメインは通常の用途も考えられこれ自体では悪性と言えるものではないが，マルウェアが活動する際に自分の環境を調べるために利用していたサイトであったためブラックリストとして登録されていた．

表 3-7　CSIRT におけるインシデントレスポンス判断マニュアルの一例

Table 3-7　An example of incident response manual in CSIRT.

| コンテンツ | 現在の状態 | 所有者 | 「トリアージ」における対応（○対応する, △調査結果次第で対応する，－対応しない） | 「対応実施」における対応（○対応する, △調査結果次第で対応する，－対応しない） |
|---|---|---|---|---|
| リダイレクト | 現在も悪性・不明 | 攻撃者<br>侵害・不明<br>共有型 | △プロキシサーバーで前後の通信を確認する. 関連する通信がある場合はその通信先に対して調査を行い, 脅威情報が存在する場合はその分類に従って対応する. | ○対応する.<br>△業務利用を確認する. 影響が想定されなければ対応する.<br>△トリアージにおける追加調査の結果に従い対応する. |
| | 現在は無害 | 攻撃者<br>侵害・不明<br>共有型 | —対応しない. | ○対応する.<br>—対応しない. |
| エクスプロイト | 現在も悪性・不明 | 攻撃者<br>侵害・不明<br>共有型 | △プロキシサーバーで前後の通信を確認する. 関連する通信がある場合はその通信先に対して調査を行い, 脅威情報が存在する場合はその分類に従って対応する. | ○対応する.<br>△トリアージにおける追加調査の結果に従い対応する. |
| | 現在は無害 | 攻撃者<br>侵害・不明<br>共有型 | —対応しない. | ○対応する.<br>—対応しない. |
| 悪性ファイル配布 | 現在も悪性・不明 | 攻撃者<br>侵害・不明<br>共有型 | ○対応する.<br>△プロキシサーバーで前後の通 | ○対応する.<br>△トリアージにおける追加調査 |

| | | | | |
|---|---|---|---|---|
| | | | 信を確認する. 関連する通信がある場合はその通信先に対して調査を行い, 脅威情報が存在する場合はその分類に従って対応する. | の結果に従い対応する. |
| | 現在は無害 | 攻撃者 | —対応しない. | ○対応する. |
| | | 侵害・不明 | | |
| | | 共有型 | | —対応しない. |
| C2 | 現在も悪性・不明 | 攻撃者 | ○対応する. | ○対応する. |
| | | 侵害・不明 | | |
| | | 共有型 | △脅威情報に対する現時点の追加調査を行う. 該当する脅威情報があった場合は対応する. | △トリアージにおける追加調査の結果に従い対応する. |
| | 現在は無害 | 攻撃者 | ○対応する. | ○対応する. |
| | | 侵害・不明 | | |
| | | 共有型 | △脅威情報に対する現時点の追加調査を行う. 該当する脅威情報があった場合は対応する. | △トリアージにおける追加調査の結果に従い対応する. |
| ソーシャルエンジニアリング | 現在も悪性・不明 | 攻撃者 | △プロキシサーバーで POST 通信の有無などを調査する. 情報送信が疑われる場合は対応する. | ○対応する.<br>△業務利用を確認する. 影響が想定されなければ対応する. |
| | | 侵害・不明 | | |
| | | 共有型 | △プロキシサーバーで POST 通信の有無などを調査する. 可能であればどのような場合に脅威となるか詳細を調査する. 情報送信 | △トリアージにおける追加調査の結果に従い対応する. |

| | | | | |
|---|---|---|---|---|
| | | | が疑われる場合は対応する. | |
| | 現在は無害 | 攻撃者侵害・不明<br>共有型 | —対応しない. | ○対応する.<br>△業務利用を確認する. 影響が想定されなければ対応する.<br>—対応しない. |
| キルスイッチ | 任意 | 任意 | ○対応する. | ○プロキシサーバー等で通信を確保する. |
| 権威 DNS サーバー | 任意 | 任意 | △脅威情報に対する現時点の追加調査を行う. 該当する脅威情報があった場合はその分類に従って対応する. | △トリアージにおける追加調査の結果に従い対応する. |
| DNS シンクホール | 任意 | 任意 | △プロキシサーバーで前後の通信を確認する. 関連する通信がある場合はその通信先に対して調査を行い, 脅威情報が存在する場合はその分類に従って対応する. | ○対応する. |
| 正規コンテンツ | 任意 | 任意 | △プロキシサーバーで前後の通信を確認する. 関連する通信がある場合はその通信先に対して調査を行い, 脅威情報が存在する場合はその分類に従って対応する. | △トリアージにおける追加調査の結果に従い対応する. |
| 不明・誤登録 | 任意 | 任意 | △調査を継続し, 個別に判断する. | △トリアージにおける追加調査の結果に従い対応する. |

## 3.8 小括

本章では，Web サイトの脅威に対して CSIRT が抱えるブラックリストの課題を示し，合理的に判断するための情報を事前に与えることで，事象を正しく把握して“効果的”な対応を可能とすること，マニュアルに基づいた判断により時間をかけずに“効率的”なトリアージを可能とすることを示した．CSIRT にとって有用な 4 項目による分類を提案し，公開されている 38 種類のブラックリストにおいてこの分類に相当する情報がほとんど含まれていないことを示した．また，100 組織において実際にインシデント判定を求められた 400 件の悪性情報について手動で調査を行い，提案する分類に妥当性があることを示した．結果として，ブラックリスト作成者が CSIRT のために提供すべき情報を示すことができた．

今後はより多くのデータで分類の網羅性を検証することができる．特に，新しい脅威や今回の評価には存在しなかった想定しない事例が出現し，「コンテンツ」において適切な分類がない事態が考えられる．現状では「不明」に分類することになっており，数が少ないうちはそれでも有効に機能すると考えているが，将来的には分類を見直す必要が出てくる可能性もある．また，実データを用いた評価において悪性 Web サイト情報の分類を手動で行ったが，実際の運用ではコストにつながるため，分類作業を自動で行う技術が求められる． CSIRT における実際のインシデントレスポンスの負荷の軽減を評価し，またヒアリング等を実施することで，CSIRT にとってより適切な分類を考えることができる．

悪性ファイルの中でも広告を表示するアドウェアや仮想通貨を採掘するマイニングツールなどは Potentially Unwanted Program（PUP）などと呼ばれる．PUP はほとんどマルウェアと同等の挙動を示すものから，広告を表示するだけのものまで様々であるが，組織のポリシーによって CSIRT のトリアージにおける優先度を下げられるものがあるため，さらなる分類を検討することができる．

侵害サイトは何らかの脆弱性があると考えられるため，複数の攻撃者に侵害される場合がある．そのため，ブラックリスト作成者が付与したラベルは脅威のひとつに過ぎず，別の脅威を含むかもしれない．このような場合を考慮するフレームワークも検討できる．

# 第 4 章
# 制御フローグラフの比較を用いた関数の検索によるマルウェア解析の効率化

本章では，インシデントレスポンスの「事象の分析」において，影響範囲を迅速に特定するためのマルウェアの静的解析で必要となる関数の検索手法について提案する．

## 4.1 はじめに

深刻なインシデントにおけるインシデントレスポンスの「事象の分析」では，マルウェア解析による詳細調査が求められる場合があり，“効果的”な対策が求められる場面においてはこの作業に取り組まなければならない．別のインシデントレスポンスで解析したマルウェアの情報がある場合，予備知識がない状況よりも効率的に解析が行えることがある．マルウェアを静的解析する場合に，これから解析するマルウェアから以前解析したマルウェアの関数に相当するコードを特定することで，“効率的”に解析作業に取りかかる手法を提案する．

## 4.2 マルウェア解析において比較に求められる要件

マルウェア静的解析はコストが高いため，緊急を要するインシデントレスポンスにおいて

は，通信や暗号に関する処理など特定の挙動に着目して解析を行う．これを実現するために，ソフトウェアのセキュリティパッチによる変更内容を解析するため，2 つのプログラムの逆アセンブリを入力として「比較」を行い，相当する関数を対応づける研究が行われている[75][76]．代表的な実装として，BinDiff というソフトウェアがある[77]．BinDiff は逆アセンブリ中に含まれる全ての関数について対応付けを試みるため，プログラムの変更内容を抽出する目的において実力を発揮する．これをマルウェア解析に適用した場合にも目的のコードを特定する上で一定の成果が得られるが，後戻りを行わずに局所的な最適解を選択し続ける貪欲アルゴリズムで探索を行うため，連鎖的に対応付けを間違えてしまう場合があり，さらに間違えた場合は解析者が利用できる情報がないという課題があった．

2 つのプログラム全体の対応や変更点を抽出する必要はなく，目的とする関数が対応付けられれば十分である．図 4-1 のプログラム 1，2 における関数間の呼び出し関係を有向グラフで表現したコールグラフにおいて，BinDiff はプログラムを比較して対応する関数を出力するが，プログラム 1 の関数 C のように誤って対応付けする場合がある．関数 C を解析したい場合，関数 C に相当する可能性の高い関数から順に出力する方式が求められる．

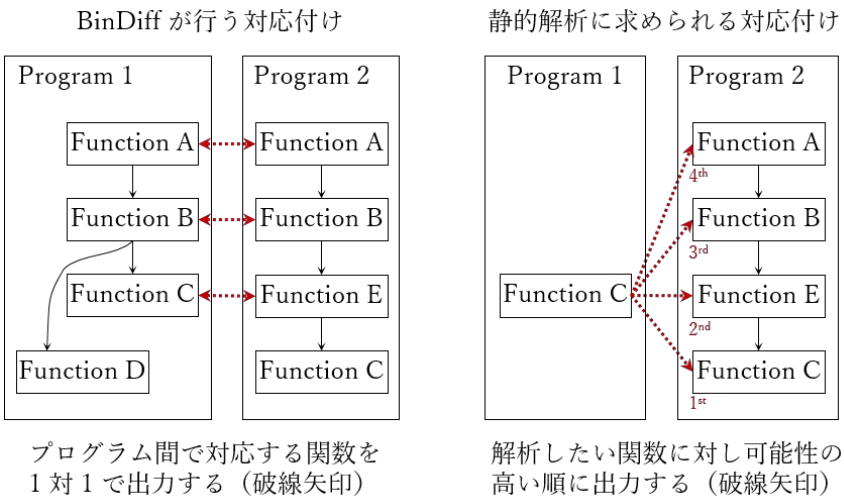


図 4-1　BinDiff の対応付けと静的解析で求められる要件

Figure 4-1　BinDiff matching method and its requirement in static analysis.

そこで, 本章では制御フローグラフの編集距離と逆アセンブリの一致率を用いた新しいア

ルゴリズムにより，過去に解析したマルウェアの関数に相当する関数を確度の高い順に「検索」して出力する BinGrep という方式を提案する．マルウェア解析者はこの中から正解を探して速やかに解析作業に取りかかることができる．

## 4.3 関連研究

本節では，インシデントレスポンスにおけるマルウェアの静的解析を効率化するため，コードを特定することを目的とした関連研究を示す．

### 4.3.1 暗号アルゴリズムと実行コードの特定

プログラムのソースコードが入手できない場合に，逆アセンブルされた実行命令列からリバースエンジニアリングでプログラムの仕様を調査する．その中でも，プログラムの中で使用された暗号アルゴリズムと暗号処理に関わる命令列の箇所を特定する研究が行われている．

Gröbert らは，プログラムを動作させて実行命令列を取得し，複数の経験則を組み合わせることで暗号アルゴリズムを含む関数の場所を特定し，平文，暗号文，暗号鍵の組み合わせを取得して既存アルゴリズムと比較することで暗号アルゴリズムを特定する手法を提案している[70]．Calvet らは，難読化されたプログラムにおいても暗号アルゴリズムの特定を可能とするため，命令単位の実装に依存しない入出力パラメーターの特定手法を提案している[71]．

### 4.3.2 プログラム間における関数の対応と差分の特定

リバースエンジニアリングにおいてソフトウェアのセキュリティパッチによる変更内容を解析するため，2 つの実行ファイルを入力として，プログラム間で対応する箇所や差分を特定する研究が行われている．

バイナリファイルにおけるバイト単位の値を単純に比較し，一致や差分を特定する実装がある[72][73][74]．ただし，コンパイラによって命令順序が入れ替わる，異なるレジスタが割り当てられる，異なる命令を使用するなど，同一のソースコードであってもコンパイ

ラや最適化オプションによって出力される実行コードは多様である．これらを全て差分と判断してしまうため，パッチ解析を目的とするリバースエンジニアリングにおいては実用的でない．

そこで，2 つの逆アセンブルされた命令列を入力として，意味上の一致もしくは差分を特定する研究が行われている．Flake は，逆アセンブリを関数単位で分割して有向グラフで表現したコールグラフと，さらに関数を制御命令単位で分割して有向グラフで表現した制御フローグラフを定義し，コールグラフのマッチング問題を解く経験則的なアルゴリズムを構築することで，2 つのプログラムの一致と差分を特定する手法を提案している [75]．Dullien らはこの手法を拡張し，Property 関数を用いて比較範囲を適切に限定する事で精度を向上させる手法を提案し，Microsoft Windows のセキュリティ更新プログラムの修正箇所を特定している [76]．また，このアルゴリズムを BinDiff というソフトウェアで実装している [77]．Bourquin らは，BinDiff が対応付けできなかった残りの関数の集合に対して，さらに拡張割り当て問題のアルゴリズムを利用して対応づけを行う BinSlayer という手法を提案している [78]．Gao らは，最大共通部分グラフを取得する近似アルゴリズムを用いて，コールグラフと制御フローグラフそれぞれにおいて，一致強度をパラメーターに関数やベーシックブロックの対応を発見する BinHunt という手法を提案し，gzip と tar のパッチによる差分を抽出している [79]．Ming らは，BinHunt において関数境界が正しく取得できない状況を想定し，関数単位ではなくベーシックブロック単位で比較する iBinHunt という手法を提案し，thttpd と gzip におけるパッチの修正箇所を特定している [80]．その他にも，グラフを利用してプログラムの対応と差分を出力する実装が存在する [81][82][83][84]．

### 4.3.3 再利用されたコードの特定

コードの盗用や使いまわしの特定，フォレンジックの効率化のため，コンパイルされたプログラムを対象として再利用されたコードを特定する研究が行われている．

LeDoux らは，関数に含まれる実行コードを抽象化した上でそれぞれの関数を複数のハッシュ値で表現して関数を特定する FuncTracker という手法を提案している [85]．ただし，ハッシュ値を利用するため，命令や命令順序の変化に対しては同一の関数と見なせないという課題があった．Ruttenberg らは，2 回のクラスタリングを行うことで，関数単位では

なくソフトウェアコンポーネント単位の再利用を特定する手法を提案している[86].

また，コンパイルされたプログラムから，コードの使い回しによって伝搬的に発生したコードクローン脆弱性を特定する研究が行われている．Pewny らは，過去に発見された脆弱性の機械語命令列を正規化してから式木で表現してこれをシグネチャとし，検査対象の命令列から生成された式木との編集距離を算出することで脆弱性を発見する手法を提案している[87]．中島らは，機械語命令列の正規化を行った上で，局所的な類似度を算出する独自の文字列検索アルゴリズムを適用しコードクローン脆弱性を発見する手法を提案している[88].

### 4.3.4 マルウェアの分類

プログラムにおいて関数やコードを特定する手法を応用し，マルウェアを分類する研究が行われている．大量のマルウェアを比較する必要があるため，プログラムの特徴を抽象化して計算を高速化している．

機械語命令列の特徴を利用してマルウェアを分類する手法が提案されている．岩村らは，逆アセンブリにおける機械語命令列の最長共通部分列 (LCS) の長さを用いてマルウェアの類似度を定義する手法を提案している[89]．Karim らは n-gram を拡張した n-perm という機械語命令列の順序の変化に強い指標を利用した分類手法を提案している[90]．Gheorghescu は，ベーシックブロック単位で命令列の編集距離を算出して，これを類似度として分類する手法を提案している[91].

グラフを利用してマルウェアを分類する手法も提案されている．岩本らは，共通のソースコードから作成されたマルウェアは API 関数の呼び出し順序も変わらない事に着目し，制御フローグラフから API 推移グラフを構築してマルウェアを分類する手法を提案している[92]．Hu らは，2 つのコールグラフ間の類似度を求めるための近似アルゴリズムと，複数のマルウェアと比較するためのインデックス構造を提案し，大量のマルウェアを分類する SMIT というマルウェア管理システムを構築している[93]．Kinable らは，コールグラフ間の編集距離を求める近似アルゴリズムを用いてマルウェアを比較し，k-medoid と DBSCAN というクラスタリング手法を用いて評価を行っている[94].

### 4.3.5 関連研究における課題

インシデントレスポンスを目的としたマルウェア解析を効率化するため，解析すべき実行コードを特定する関連研究について示した．4.3.1 項の暗号アルゴリズムと実行コードを特定する手法は，単体のマルウェアを用いて適用できる一方で，暗号のように特定の処理だけを対象としたものである．4.3.2 項のプログラム間における関数の対応と差分を特定する手法は，リバースエンジニアリングの中でも主にパッチの修正箇所の特定を目的としており，2 つのプログラム間における関数の対応と差分を特定する手法である．2 つのプログラムにおける関数を 1 対 1 で対応するが，誤って対応付けされた関数に関しては解析者にとって無意味な出力である．4.3.3 項の再利用されたコードを特定する手法は，一般的な開発者によって開発されたプログラムを想定している．コンパイラやコンパイルオプション等の違いによる実行コードの変化を考慮しているが，マルウェアのように実行コードが変化しやすいプログラムには適していない．4.3.4 項のマルウェアの分類は，マルウェアの類似度を求める手法であり実行コードの特定は行っていない．

## 4.4 BinDiff のマルウェア解析への活用と課題

マルウェアの亜種を解析するにあたって, 以前解析した処理に相当するコードが特定できると，速やかに作業に取りかかることができる．4.3.2 項で取り上げた BinDiff は，2 つのプログラムの「比較」の機能を有するが，マルウェア解析において目的コードを特定する方式としては課題がある．本節では，BinDiff のアルゴリズム自体にその課題が内在することを示す．

BinDiff は逆アセンブラの IDA Pro[100]が出力した逆アセンブリを受け取る．高レベル言語で開発された構造化プログラムは，一般的にコールグラフと制御フローグラフというグラフの入れ子で表現することができる．コールグラフはプログラムに含まれる関数を頂点，関数の呼び出しを辺で定義した有向グラフである．1 つの関数に含まれる逆アセンブリは，制御転送命令（jmp 系命令）が現れる箇所で分割され，この単位の逆アセンブリを基本ブロックと呼ぶ．制御フローグラフは基本ブロックを頂点，制御転送命令によるジャンプを辺で定義した有向グラフである．また，制御フローグラフ，すなわちコールグラフの各

頂点に対して（α, β, γ）=（基本ブロックの数，基本ブロック間の辺の数，関数呼び出しの数）という 3 次元の特徴量を与える．

比較される 2 つのプログラム A, B における関数の集合，すなわちコールグラフの頂点の集合を $G_A=\{a_1, \cdots, a_n\}$, $G_B=\{b_1, \cdots, b_m\}$ とし，その部分集合を $S_A \subseteq G_A$，$S_B \subseteq G_B$ とする．BinDiff は Selector と Property という関数を用いる．Selector ε は $a_i \in S_A$ と $S_B$ を入力として，$a_i$ に一致する $S_B$ の単独の要素を返す関数である．ただし，候補が複数存在する場合は何も返さない．「一致する」とは様々なアイデアが考えられるが，例えば（α,β,γ）が一致する，命令列から生成した CRC32 が一致する，関数名が一致する，など複数の Selector を定義して利用する．Property π は $a_i \in S_A$ と $S_A$ もしくは $b_j \in S_B$ と $S_B$ を入力として，$a_i$ や $b_j$ と関係の深い頂点の集合 $S'_A$ や $S'_B$ を出力する．例えば，頂点に対する呼び出し元となる頂点の集合を返す関数，頂点から呼び出す先の頂点の集合を返す関数がある．

アルゴリズム 4-1 の initialMatches で最初の対応付けを行う．プログラム A と B における関数の集合 $S_A$，$S_B$ を入力として，全ての頂点 $a_i \in S_A$ に対し全ての Selector を適用し，その出力 $b_j$ に対して対応 $\{a_i \mapsto b_j\}$ をリスト M の中に記録する．対応付けられた頂点は順次 $S_A$，$S_B$ の中から除外する．

アルゴリズム 4-2 の propagateMatches は既に対応付けされた頂点のリスト M をもとに，対応付けの範囲を拡大する．全ての対応付け $\{a_i \mapsto b_j\} \in M$ に対して $a_i$, $b_j$ に Property を適用して関連する頂点の集合を取得し，再びこの範囲で initialMatches と同様に全ての Selector を適用して対応をリスト M に記録する．

BinDiff は 2 つのプログラムの逆アセンブリを入力としてそれぞれコールグラフと 3 次元の特徴量を取得し，アルゴリズム 4-3 の binDiff を実行する．この中で，initialMatches を実行して初期の対応付けを求め，停止するまで繰り返し propagateMatches を実行する．

アルゴリズム 4-1　initialMatches

Algorithm 4-1　initialMatches

| | |
|---|---|
| **In** | $S_A$: プログラム A に含まれる関数の集合 |
| | $S_B$: プログラム B に含まれる関数の集合 |
| **Out** | M: プログラム A と B の間で対応する関数のリスト |
| | $S_A$: 入力の $S_A$ に対し対応付けが残った関数の集合 |
| | $S_B$: 入力の $S_B$ に対し対応付けが残った関数の集合 |
| **1** | `Function initialMatches(`$S_A$`, `$S_B$`)` |
| **2** | $M \leftarrow \emptyset$ |
| **3** | **foreach** $a_i \in S_A$ **do** |
| **4** | **foreach Selector** $\varepsilon$ **do** |
| **5** | **if** $(a_i, b_j) \leftarrow \varepsilon(a_i, S_B)$ **then** |
| **6** | $M \leftarrow M \cup \{a_i \mapsto b_j\}$ |
| **7** | $S_A \leftarrow S_A \setminus \{a_i\}$ |
| **8** | $S_B \leftarrow S_B \setminus \{b_j\}$ |
| **9** | **break** |
| **10** | **return** $(M, S_A, S_B)$ |

アルゴリズム 4-2 propagateMatches

Algorithm 4-2 propagateMatches

| | |
|---|---|
| **In** | M: プログラム A と B の間で対応する関数のリスト |
| | $S_A$: プログラム A に含まれる関数の集合 |
| | $S_B$: プログラム B に含まれる関数の集合 |
| **Out** | M: 入力の M に対し対応付けを拡大した関数のリスト |
| | $S_A$: 入力の $S_A$ に対し対応付けが残った関数の集合 |
| | $S_B$: 入力の $S_B$ に対し対応付けが残った関数の集合 |
| **1** | **Function** **propagateMatches**(M, $S_A$, $S_B$) |
| **2** | **foreach** $\{a_i \mapsto b_j\} \in M$ **do** |
| **3** | **foreach** **Property** $\pi$ **do** |
| **4** | $S'_A \leftarrow \pi(a_i, S_A)$; |
| **5** | $S'_B \leftarrow \pi(b_j, S_B)$; |
| **6** | **if** $S'_A \neq \emptyset \wedge S'_B \neq \emptyset$ **then** |
| **7** | **foreach** $a'_i \in S'_A$ **do** |
| **8** | **foreach** **Selector** $\varepsilon$ **do** |
| **9** | **if** $(a'_i, b'_j) \leftarrow \varepsilon(a'_i, S'_B)$ **then** |
| **10** | $M \leftarrow M \cup \{a'_i \mapsto b'_j\}$ |
| **11** | $S'_A \leftarrow S'_A\{a'_i\}$ |
| **12** | $S'_B \leftarrow S'_B\{b'_j\}$ |
| **13** | $S_A \leftarrow S_A ¥ \{a'_i\}$ |
| **14** | $S_B \leftarrow S_B ¥ \{b'_j\}$ |
| **15** | **break** |
| **16** | **return** (M, $S_A$, $S_B$) |

アルゴリズム 4-3　binDiff

Algorithm 4-3　binDiff

| | |
|---|---|
| **In** | $G_A$: プログラム A における関数の集合 |
| | $G_B$: プログラム B における関数の集合 |
| **Out** | M: 対応付けされた関数のリスト |
| | $S_A$: プログラム A にのみ含まれる関数の集合 |
| | $S_B$: プログラム B にのみ含まれる関数の集合 |
| **1** | `Function binDiff(`$G_A$`, `$G_B$`)` |
| **2** | $S_A \leftarrow G_A$ |
| **3** | $S_B \leftarrow G_B$ |
| **4** | $M' \leftarrow \emptyset$ |
| **5** | $(M, S_A, S_B) \leftarrow$ `initialMatches(`$S_A$`, `$S_B$`)` |
| **6** | `while` $M' \neq M$ `do` |
| **7** | $M' \leftarrow M$ |
| **8** | $(M, S_A, S_B) \leftarrow$ `propagateMatches(`$M, S_A, S_B$`)` |
| **9** | `return` $(M, S_A, S_B)$ |

BinDiff のアルゴリズムについて，図 4-2 のコールグラフを持つプログラムを例に，自身のプログラムと比較することで実際の動作を示す．頂点のラベルはノードの番号と特徴量（α, β, γ）である．この例で使用する Selector は特徴量が一致するものを選ぶ関数とし，propagateMatches は呼び出し元，呼び出し先の頂点の集合を返す 2 つの関数とする．最初に InitialMatches を実行すると, Selector によって頂点 1, 5 がユニークに対応付けられる．次に 1 回目の propagateMatches において，Property を頂点 1 に適用して呼び出し先の頂点 {2, 3, 4} を取得してこの中で再度 Selector を実行すると，特徴量がユニークである頂点 2 を対応付けられる．また，頂点 5 の呼び出し元，呼び出し先の頂点 {4, 6} の中でそれぞれ Selector を実行すると，頂点 4 と頂点 6 の両方が対応付けられる．propagateMatches の 2 回目の実行では，同様に頂点 3 を対応付ける．

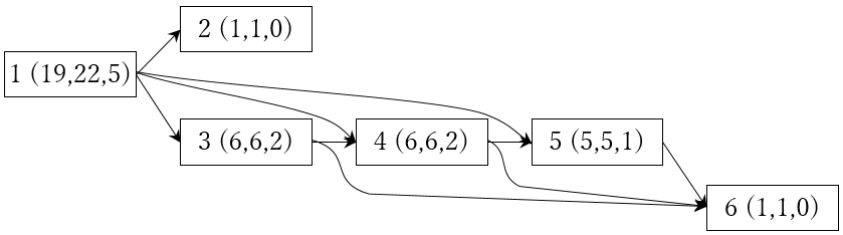


図 4-2　BinDiff アルゴリズムを実行するコールグラフ

Figure 4-2　Callgraph example for BinDiff algorithm.

BinDiff は，2 つのプログラムにおいて同じ関数から呼び出された関数は同一である可能性が高いという前提に基づき，関数の呼び出し関係に着目しながら関数の対応付けを行う．逆アセンブリ中に含まれる全ての関数について対応付けを試みるため，プログラムの変更内容を抽出する目的において実力を発揮する．BinDiff をマルウェア解析に適用した場合にも一定の成果が得られるが，途中の対応付けが正しいと仮定して関数の呼び出し関係を辿りながら処理を続ける貪欲アルゴリズムであるため連鎖的に判断を誤りやすい．さらに，それぞれの関数に対して対応する関数を 1 つしか出力しないため，出力が誤っていた場合は何も情報が残らないという課題がある．これらの理由により，マルウェアのように意図的に解析を妨害する機能が実装されたプログラムにおいては，BinDiff では目的のコードを検出する上で十分な情報が得られない場合がある．

## 4.5　提案方式 BinGrep

### 4.5.1　要件

本節では，インシデントレスポンスにおけるマルウェア解析に適したコード比較方式（提案方式）に求められる要件について示す．インシデントレスポンスにおけるマルウェアの静的解析では特定の挙動に着目して解析を行う．まずは過去に解析したマルウェアとこれから解析するマルウェア亜種を BinDiff に適用して全ての関数について対応付けを行い，解析したい関数に相当するコードを探す．BinDiff の出力を確認して正解であると判断すれば解析者はそのままアセンブリを読み解く．解析者が誤った出力であると判断した場合に提案方式を適用する．解析者は，提案方式に対して過去に解析したマルウェアの逆アセンブリ，解析したい処理を行う関数，マルウェア亜種の逆アセンブリを入力して，マルウェア亜種における関数を「検索」する．

提案方式では，相当する可能性のある関数の候補を確度の高い順に複数出力することで，1 位の出力が誤っていたとしてもマルウェア解析者が下位の出力から正解を探して解析に取りかかれる必要がある．この際，複数の検索結果に対してどれが求めるコードであるかマルウェア解析者が速やかに判断できるよう，逆アセンブリを並べて比較できるインターフェースとなることも考慮する．

マルウェアの静的解析は市販の PC を使用することが一般的であり，解析作業に支障のない待ち時間で処理が完了することを想定する．

提案方式では BinDiff を代表とする従来研究と同様に，逆アセンブルにより関数構造が正しく復元できていることを前提とする．プログラムによっては関数名がシンボルとして含まれ関数の比較に利用できることがあるが，マルウェアの場合は通常シンボルが削除されるため，外部関数名などの最低限のシンボルしか利用できない状況を想定する．

### 4.5.2　定義

提案方式では，関数が似ているか判断するために有向グラフの編集距離を利用したアルゴリズムを提案する．頂点にラベルが付与されたグラフにおける編集操作を，頂点と辺の

挿入，頂点と辺の削除，頂点のラベルの置換という操作で定義する．一方のグラフをもう一方のグラフに変換するのに必要となる最小の編集操作の数を有向グラフの編集距離と定義する．

有向グラフの中でも，根と呼ばれる頂点が存在し，連結で閉路がないものを木と呼び，頂点がラベルを持つ場合はラベル付き木と定義する．

### 4.5.3 グラフの比較と編集距離

プログラムの比較研究において，しばしばグラフの一致問題を解決する必要がある．2 つのグラフに対して共通する最大の部分グラフを求める最大共通部分グラフ（MCS）問題は NP-Hard であることが知られている[95]．

また，グラフの類似性を定義する指標として，2 つのグラフの共通部分グラフを求める代わりに，グラフ間の編集距離を利用することができる．一般的なグラフに対して編集距離を求める問題は NP-Hard であることが知られている．パターン認識の分野では大きいグラフに対して計算する必要があるため，効率的に近似解を求めるアルゴリズムが研究されている[96]．

ラベル付き木に対する編集距離としては, Zhang の計算量 $n^4$ のアルゴリズム[97]と Klein の計算量 $n^3 \log n$ のアルゴリズム[98]が知られている．ただし，最適なアルゴリズムは入力するグラフに依存する[99]．

### 4.5.4 提案アルゴリズム

本節では提案方式におけるアルゴリズムについて説明する. 4.5.1 項の要件を満たすため，関数間の距離を用いることで，相当する可能性の高い複数の関数を順番に出力できるアルゴリズムを構築する．提案アルゴリズムはアルゴリズム 4-4 の constructCFT とアルゴリズム 4-5 の binGrep で構成される．

制御フローグラフ G を頂点と辺の集合 $G = (V, E)$ と定義し, 頂点を $V = \{V_1, V_2, ..., V_n\}$, 辺を $E = \{(V_i, V_j), ...\}$ (i, j は頂点 V のインデックス）と定義する．constructCFT は，グラフ間における編集距離を効率的に計算するため，制御フローグラフ G に対して近似するラベル付き木を出力する関数である．JmpTo(G, i) は $V_i$ に対する関数呼び出し先の頂点の集

合を返す関数, DelJmp(G, i, j）は G の辺から（$V_i$, $V_j$）を削除する関数とする. SetLabel(G, i）は G の頂点 $V_i$ にラベルを与える関数とする．基本ブロック $V_i$ において call 命令で外部関数を呼び出している場合，この外部関数名を昇順に連結した文字列をラベルとする．例えば,「call memset」と「call printf」を実行する基本ブロックにおけるラベルは“memset printf”となる．外部関数呼び出しが存在しない場合は共通のラベル“NULL”を与える．constructCFT は再帰関数として深さ優先探索により制御フローグラフ G を探索し，記憶領域 m に探索済みの頂点を追加する．ただし，探索済みの頂点を発見した場合は G から辺を削除する.

binGrep は, プログラム A において検索元となる関数 $G_A$ とプログラム B における関数の集合 $G_B^*$を入力として，$G_A$ と $G_B^*$の全ての要素と比較し，順位の高い順に出力する関数である. constructCFT を使用してグラフを初期化した上で EditDistance($G_A$, $G_B$）を用いて 2 つの制御フローグラフ間における編集距離を計算し, さらに LCS($G_A$, $G_B$）を用いて関数 $G_A$, $G_B$ に含まれる命令列のオペコードに対して最長共通部分列を計算し，これらの結果を 2 次元ベクトルとして r に格納する．最後に Sort(r）で第 1 要素を昇順に，同着の場合は第 2 要素を降順にソートして出力する.

アルゴリズム 4-4　constructCFT

Algorithm 4-4　constructCFT

| | |
|---|---|
| **In** | G: 制御フローグラフ<br>i: G の頂点（基本ブロック）のインデックス<br>m: 訪問済みであることを記録するための記憶領域 |
| **Out** | G: 制御フローグラフに近似するラベル付き木 |

```
1  function constructCFT(G, i, m)
2    foreach j ∈ JmpTo(G, i) do
3      if j ∈ m do
4        G ← DelJmp(G, i, j)
5      else
6        m ← m ∪ {j}
7        G ← SetLabel(G, j)
8        G ← constructCFT(G, j, m))
9    return G
```

アルゴリズム 4-5　binGrep

Algorithm 4-5　binGrep

| | |
|---|---|
| **In** | $G_A$: プログラム A における検索元となる関数<br>$G_B^*$: プログラム B の関数 $G_B$ の集合 |
| **Out** | r': 検索結果のリスト |

```
1   function binGrep(G_A, G_B*)
2     r ← ∅
3     SetLabel(G_A, 0)
4     constructCFT(G_A, 0, ∅)
5     foreach G_B ∈ G_B* do
6       SetLabel(G_B, 0)
7   constructCFT(G_B, 0, ∅)
8       r ← r ∪ (EditDistance(G_A, G_B), LCS(G_A, G_B))
9     r' ← Sort(r)
10    return r'
```

### 4.5.5 実装

IDA Pro が生成した逆アセンブリを含むデータベース（IDB ファイル）を使用し, IDA Pro の API を利用するスクリプトとして提案アルゴリズムを実装した．ラベル付き木における編集距離の計算は, Zhang のアルゴリズムを実装した Python ライブラリ[101]をベースに, Python を C 言語拡張する Cython[102]に移植して高速化したものを使用した．

提案アルゴリズムの出力例を示す．図 4-3 は CUI 上で実行した結果である．検索元のプログラムの IDB ファイル，検索元の関数名，検索先のプログラムの IDB ファイルを引数として指定して実行する．IDA Pro のコンソール（Output window）にも同じ結果を出力することができる．結果はデフォルトで 10 位まで出力するが，オプションによって上限なく出力することも可能である．検索結果は編集距離（`Dist`）が小さい順に並び，さらに同着の関数は命令の一致率（`Sim`）が大きい順に並ぶ．最小の編集距離を持つ関数名の右には「*」印が付与される．

マルウェア解析者は関数の検索に関して BinDiff と提案手法を併用することを想定している．ここで実際の利用イメージを説明する．マルウェア解析者はまず BinDiff を用いて全ての関数の対応付けを行う. 検索したい関数 `f` が含まれた解析済みマルウェアを `A` として，`A` を解析対象マルウェア `B` と比較する．ここで関数 `f` に対応したマルウェア `B` の関数 `g` を確認し，正解であれば検索は完了となる．不正解であった場合は，関数 `f` とマルウェア `B` を提案手法で検索し, 図 4-4 に示すように出力された関数を上位から順番に確認する．左側のウインドウは検索元プログラム，右側のウインドウは検索先プログラムを IDA Pro で開いた画面であり, 検索先プログラムの下部にある Output Window 上に実行結果を表示する．表示された関数名をダブルクリックすると上部に該当する関数の逆アセンブリが表示されるため，解析者は速やかに関数の内容を確認できる．

```
$ bingrep.py Emdivi_01.idb sub_405954 Emdivi_02.idb
Src Program  is ‘Emdivi_01’
Src Function is ‘sub_405954’
Dest Program is ‘Emdivi_02’
-------------------------------------------------
No. Dist   Sim:    Dst Function
  1    0   100:      sub_405D1B *
  2    0    99:      sub_410BE1 *
  3    4    70:      sub_40BF28
  4   13    16:       __alldvrm
  5   15    15:        __alldiv
  6   15    13:        __allrem
  7   15    12:         _memset
  8   15     8:         _strcmp
  9   16    19:      sub_40DB78
  9   16    19:      sub_4035BF
```

図 4-3　提案アルゴリズムの実行例（CUI）

Figure 4-3　Execution example of proposal method (CUI).

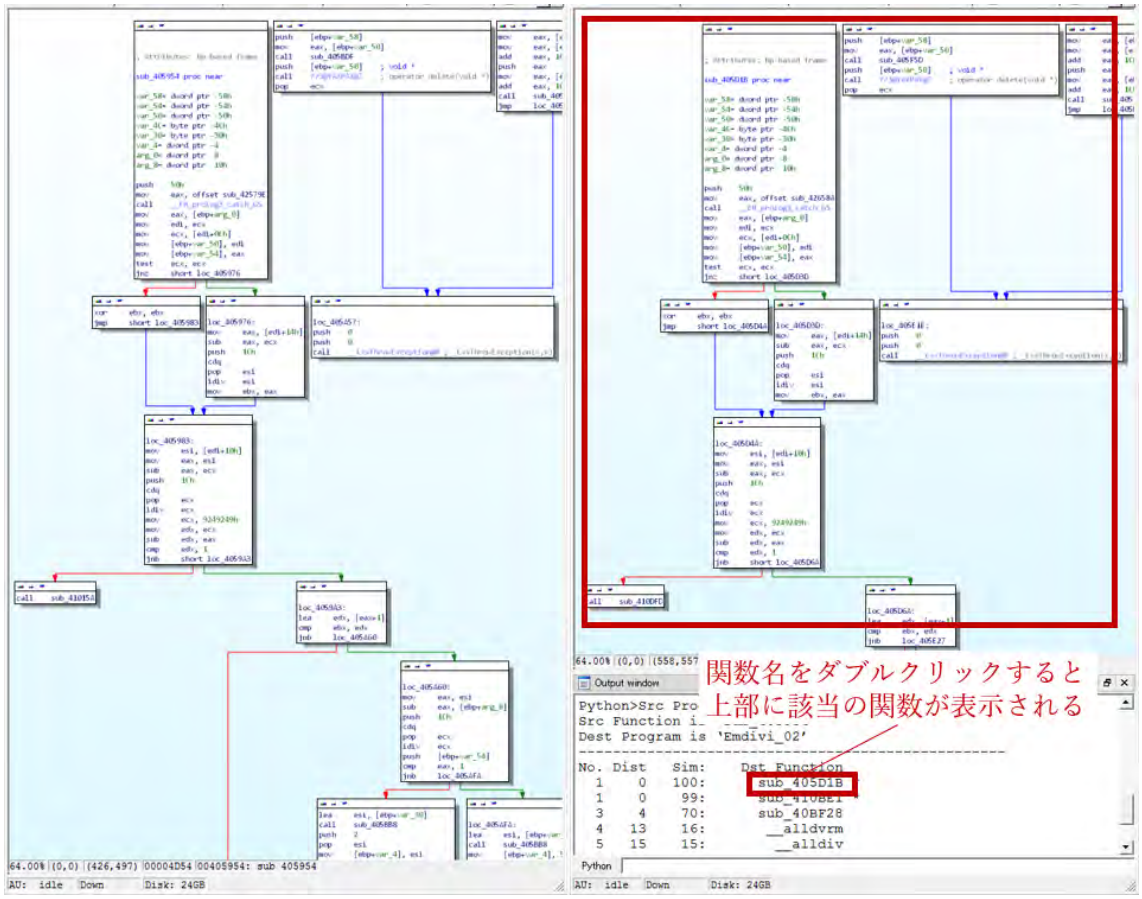


図 4-4　提案アルゴリズムの実行例（GUI）

Figure 4-4　Execution example of proposal method (GUI).

## 4.6　評価

提案方式（BinGrep）の有効性について，正規のプログラムとマルウェア検体を対象に評価を行った．正規のプログラムでは 2 つのバージョンの GNU bash と binutils において全ての関数を検索し BinDiff の結果と比較した．マルウェア検体においては，6 つの Emdivi を使用し, インシデントレスポンスで重要となった 3 つの関数について検索を行い, BinDiff の結果と比較した．さらに，Emdivi と PlugX のそれぞれ 2 つの検体を使用し，全ての関数を検索して BinDiff の結果と比較した．

マルウェア解析者は一般的な PC を使用することを想定した. 評価に用いた PC の仕様は，CPU Intel Core M-5Y71 1.20GHz，メモリー容量 8GB，SSD 256GB，Windows 8 64bit である．

### 4.6.1　GNU bash による評価

GNU bash の 2 つのバージョンのプログラムを用いて提案方式の評価 1 を行った．使用したバージョンは表 4-1 のとおりである．

2 つのプログラムに対して Linux の strip コマンドでシンボル情報を削除した上で，bash 4.0 でシンボルが削除された 381 個の関数に対して提案方式を用いて bash 4.3.30 の関数を検索した．同じシンボル情報を持つ関数を正解と定義し，BinDiff が正解を出力できるかどうか，提案方式における検索結果の上位 10 位以内に正解が出力されるかどうか，それぞれ評価を行った．

提案方式と BinDiff の比較結果を表 4-2 に示す．BinDiff は 345 個の関数を正解し，提案方式は 343 個の関数を正解した．全体としてほぼ同精度の精度を達成した．要件として，BinDiff が不正解であった関数に対する提案方式の適用を想定しており，BinDiff が不正解となった 36 個の関数のうち 29 個（80.6%）に対して正解したため，BinDiff と組み合わせることで全体の正解範囲が広がり，提案方式の有用性が示せた．

また，提案方式において正解が検索結果に表示された順位の統計を図 4-5 に示す．335 個の関数は 1 位として出力され，38 個の関数が 10 位以下として出力された．

表 4-1　比較に使用した bash のバージョン

Table 4-1　bash version.

| No. | 評価プログラム | 公開時期 |
|---|---|---|
| 1 | bash 4.0 | 2009 年 2 月 20 日 |
| 2 | bash 4.3.30 | 2014 年 11 月 7 日 |

表 4-2　bash における提案方式と BinDiff の比較評価

Table 4-2　Experiment result of bash comparison.

| | | BinDiff | | 合計 |
|---|---|---|---|---|
| | | 正解 | 不正解 | |
| 提案方式 | 正解 | 314 | 29 | 343 |
| | 不正解 | 31 | 7 | 38 |
| 合計 | | 345 | 36 | 381 |

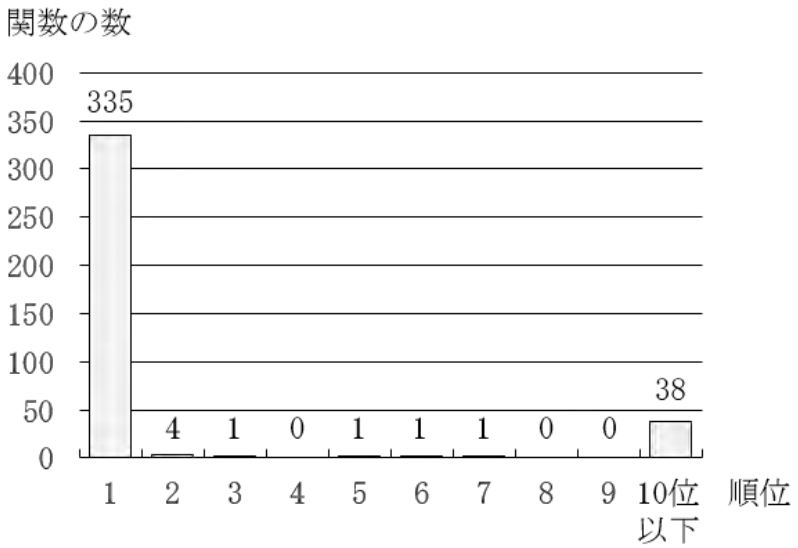


図 4-5　bash における提案方式の検索順位

Figure 4-5　Rank result of bash.

提案方式と BinDiff の実行時間を評価した．BinDiff は全ての関数に対して 3.7 秒で処理を完了した．提案方式は 1 つの関数に対して平均 0.9 秒，最大 19 秒，全ての関数の検索に対して 339.1 秒という結果となった．BinDiff は関数の呼び出しを辿り連鎖的に対応付けを行うため全体として高速に動作した．提案方式は検索する関数に対して全ての関数との編集距離を計算するため, 関数 1 つあたりの計算時間は高速となったが, 全体としては BinDiff より時間を要した.

### 4.6.2　GNU binutils による評価

GNU binutils の 2 つのバージョンのプログラムを用いて同様に提案方式の評価 2 を行った．binutils には 15 個のプログラム addr2line, ar, as, c++filt, elfedit, gprof, ld, nm, objcopy, objdump, ranlib, readelf, size, strings, strip が含まれる. 使用したバージョンは表 4-3 のとおりである.

bash と同様に，binutils 2.22 でシンボルが削除された 10668 個の関数に対して提案方式を用いて binutils 2.25.1 の関数を検索した．提案方式を BinDiff と比較した結果を表 4-4 binutils における提案方式と BinDiff の比較評価に示す. BinDiff は 9635 個の関数を正解し，提案方式は 9651 個の関数を正解した．特に，BinDiff が不正解となった 1033 個の関数のうち 674 個（65.2%）に対して正解し，bash と同様の評価結果となった.

表 4-3 比較に使用した binutils のバージョン

Table 4-3 binutils versions.

| No. | 評価プログラム | 公開時期 |
|---|---|---|
| 1 | binutils 2.22 | 2011 年 11 月 21 日 |
| 2 | binutils 2.25.1 | 2015 年 7 月 21 日 |

表 4-4 binutils における提案方式と BinDiff の比較評価

Table 4-4 Experiment result of binutils comparison.

| | | BinDiff | | 合計 |
|---|---|---|---|---|
| | | 正解 | 不正解 | |
| 提案方式 | 正解 | 8977 | 674 | 9651 |
| | 不正解 | 658 | 359 | 1017 |
| 合計 | | 9635 | 1033 | 10668 |

提案方式と BinDiff の実行時間を評価した．BinDiff は全ての関数に対して 4.5 秒で処理を完了した．提案方式は 1 つの関数に対して平均 0.9 秒，最大 24.8 秒，全ての関数の検索に対して 1 プログラムあたり平均 640.1 秒であり，bash と同様の傾向を示した．

### 4.6.3 Emdivi による評価

マルウェア検体を用いて評価を行った．表 4-5 は一連の APT 攻撃で用いられた Emdivi という RAT である．このマルウェアにはバージョンがハードコードされており，検体 1〜6 は t17，検体 7〜11 は t20 というバージョンを持つ．これらのマルウェアに対して，表 4-6 に示す 3 つの状況を想定して評価 3〜5 を行った．

表 4-5　評価に使用した Emdivi 検体

Table 4-5　Emdivi samples for evaluation.

| 検体 No. | 検体名 | 関数の数 |
|---|---|---|
| 検体 1 | Emdivi t17.08.21 | 1018 |
| 検体 2 | Emdivi t17.08.26 | 1009 |
| 検体 3 | Emdivi t17.08.30 | 957 |
| 検体 4 | Emdivi t17.08.30 | 949 |
| 検体 5 | Emdivi t17.08.31 | 1145 |
| 検体 6 | Emdivi t17.08.34 | 1100 |
| 検体 7 | Emdivi t20.02 | 1079 |
| 検体 8 | Emdivi t20.03 | 1543 |
| 検体 9 | Emdivi t20.03 | 1035 |
| 検体 10 | Emdivi t20.06 | 1204 |
| 検体 11 | Emdivi t20.08 | 2308 |

表 4-6　評価における Emdivi 検体の役割

Table 4-6　The role of Emdivi samples under evaluation.

| 評価 No. | 既知マルウェア | 解析対象 |
|---|---|---|
| 評価 3 | 検体 1（t17 系） | 検体 2〜6　（t17 系） |
| 評価 4 | 検体 7（t20 系） | 検体 8〜11（t20 系） |
| 評価 5 | 検体 1（t17 系） | 検体 7〜11（t20 系） |

検体 1，検体 7 において検索元関数として使用した関数を表 4-7，表 4-8 に示す．この表ではこの関数の特徴を示すため，基本ブロック（BB）の数，命令数，主な処理内容を記載している．C2 通信はファイアウォールのアクセス制御を回避するため一般的に HTTP が使用される．今回の評価に使用した関数は HTTP のリクエスト文を構築する処理と暗号化を行う処理である．インシデントレスポンスでは影響範囲を調査するため，プロキシサーバーに記録されたログを分析して他の感染端末や通信内容を特定する．これらの関数を静的解析することはログを調査する上で重要な意味を持つ．

表 4-7　評価に使用した検体 1 の関数

Table 4-7　Function name of sample 1 for evaluation.

| 関数 | BB 数 | 命令数 | 関数内で行う主な処理 |
|---|---|---|---|
| sub_40801C | 119 | 1026 | POST リクエスト文の構築 |
| sub_40CA5E | 28 | 275 | GET リクエスト文の構築 |
| sub_40204D | 13 | 157 | XXTEA による暗号化 |

表 4-8　評価に使用した検体 7 の関数

Table 4-8　Function name of sample No. 7 for evaluation.

| 関数 | BB 数 | 命令数 | 関数内で行う主な処理 |
|---|---|---|---|
| sub_415FB1 | 185 | 1254 | POST リクエスト文の構築 |
| sub_414808 | 72 | 698 | GET リクエスト文の構築 |
| sub_40113A | 45 | 263 | AES による暗号化 |

検体 1 と検体 7 における各 3 つの関数に対して BinDiff と提案方式で比較を行った結果を表 4-9，表 4-10 に示す．セルの左側は，BinDiff が出力した解答に対して正解か不正解かを「○」「×」で示したものである．右側は，提案方式において正解となるべき関数が 10 位以内に表示されたかどうかを「○」「×」で示したものであり，さらに正解が出力された順位を記載している．正解となるべき関数がプログラムに存在しない場合は「×無」と記載している．

評価 3 の結果を表 4-9 に示す. バージョンが近いほど BinDiff も提案方式も正解率が高くなる傾向がみられたが. 検体 1 と検体 4, 検体 1 と検体 5 では提案方式のみが上位に正解を出力できた関数が存在した．BinDiff が不正解を出力した場合，これまでは情報が全くなく最初からコードを読み解く必要があったが，提案方式では失敗しても 10 位以下の関数を順番に調べていくことができる.

提案方式が正解を上位に出力できなかった関数は, 関数内の制御フローグラフが大きく変化していたため正しく判定できなかった．逆に，提案方式が正解を上位に出力していた関数は関数が大きく変化していなかった. 提案方式が正解した中に BinDiff が誤答を出力したものがあったが，呼出元となる関数における制御フローグラフが大きく変化して正しく判定できず，そのため連鎖的に判断を誤っていた.

表 4-9　検体 1 における BinDiff と提案方式による評価結果

Table 4-9　BinDiff and proposing method evaluation result.

| 検体 1 の関数 | 検体 2 | 検体 3 | 検体 4 | 検体 5 |
|---|---|---|---|---|
| sub_40801C | ○, ○1st | ○, ○1st | ○, ○3rd | ×, ○2nd |
| sub_40CA5E | ○, ○1st | ○, ○1st | ×, ○7th | ×, ×30th |
| sub_40204D | ○, ○1st | ○, ○1st | ○, ○1st | ○, ○1st |

評価 4 の結果を表 4-10 に示す. 検体 7 と検体 8, 検体 7 と検体 11 において提案方式の優位性が認められた. 検体 11 においては, 検体 7 の sub_40113A に相当する AES 暗号化関数は存在しなかった. JPCERT/CC によると, t17 で使用された暗号化関数は XXTEA であり, t20 において t20.6 以下は AES，t20.7 から t20.25 までは XXTEA が使用されていることが報告されている[103]．本評価ではこの報告と一致することが確認された.

表 4-10　検体 7 における BinDiff と提案方式による評価結果

Table 4-10　BinDiff and proposing method evaluation result.

| 検体 7 の関数 | 検体 8 | 検体 9 | 検体 10 | 検体 11 |
|---|---|---|---|---|
| sub_415FB1 | ○, ○1st | ○, ○1st | ○, ○1st | ×, ○1st |
| sub_414808 | ×, ○10th | ○, ○1st | ○, ○1st | ×, ○2nd |
| sub_40113A | ○, ○1st | ○, ○1st | ○, ○1st | ×, ×無 |

評価 5 の結果を表 4-11 に示す．HTTP リクエスト文を構築する処理は実装が大きく変化しており，新たに開発されたコードであると推測された．このため正解は存在しなかった．ただし検体 11 では t17 と同じ XXTEA が使用されたため，正解を出力することができた．

表 4-11　検体 1 における BinDiff と提案方式による評価結果

Table 4-11　BinDiff and proposing method evaluation result.

| 検体 1 の関数 | 検体 7 | 検体 8 | 検体 9 | 検体 10 | 検体 11 |
|---|---|---|---|---|---|
| sub_40801C | ×, ×無 | ×, ×無 | ×, ×無 | ×, ×無 | ×, ×無 |
| sub_40CA5E | ×, ×無 | ×, ×無 | ×, ×無 | ×, ×無 | ×, ×無 |
| sub_40204D | ×, ×無 | ×, ×無 | ×, ×無 | ×, ×無 | ×, ○1st |

実行時間については，BinDiff は全体で 11.0 秒，提案方式は関数あたり最大で 1.7 秒であり，解析者が手動で検索することを想定すると十分な速度であった．

### 4.6.4　Emdivi と PlugX における全関数の評価

BinDiff と提案方式を用いてマルウェア検体における全ての関数の比較を行った．評価 6 では 4.6.3 項における Emdivi 検体 1, 2 を使用し，評価 7 では実際の APT で使用された PlugX と呼ばれる表 4-12 の RAT を使用した．その結果を

表 4-13，表 4-14 に示す．また，非常に大きい画像であるため一部となるが，可視化したグラフを図 4-6，図 4-7 に示す．この図は検体 1 のコールグラフであり，頂点は関数を意味する．BinDiff と提案方式を検体 1 の全ての関数に適用して目視で正解かどうかを判定した．正解かどうか判断できないものはともに不正解とした．たとえ正解であっても実際に解析する際に正解が判断できないと意味をなさないためである．評価結果は頂点の色で表現している．青色は BinDiff のみが正解したもの，赤色は提案方式のみが正解したもの，紫色は両方とも正解したもの，灰色は両方とも失敗したものである．この検体における関数呼び出しは非常に多く，全ての辺を記載すると現実的な図とならなかったため，各頂点に対して 1 辺だけをランダムに抜粋している．

表 4-12　評価に使用した Plugx 検体

Table 4-12　PlugX samples for evaluation.

| 検体 No. | 検体名 | 評価の役割 | 関数の数 |
|---|---|---|---|
| 検体 12 | PlugX | 既知マルウェア | 472 |
| 検体 13 | PlugX | 解析対象 | 517 |
| 検体 14 | PlugX | 解析対象 | 517 |
| 検体 15 | PlugX | 解析対象 | 495 |
| 検体 16 | PlugX | 解析対象 | 495 |
| 検体 17 | PlugX | 解析対象 | 495 |
| 検体 18 | PlugX | 解析対象 | 495 |
| 検体 19 | PlugX | 解析対象 | 495 |
| 検体 20 | PlugX | 解析対象 | 472 |
| 検体 21 | PlugX | 解析対象 | 472 |

表 4-13　Emdivi における提案方式と BinDiff の比較評価

Table 4-13　Experiment result of Emdivi comparison.

| | | BinDiff | | 合計 |
|---|---|---|---|---|
| | | 正解 | 不正解 | |
| 提案方式 | 正解 | 725 | 40 | 765 |
| | 不正解 | 3 | 250 | 253 |
| 合計 | | 728 | 290 | 1018 |

表 4-14　PlugX における提案方式と BinDiff の比較評価（平均）

Table 4-14　Experiment result of PlugX comparison (average).

| | | BinDiff | | 合計 |
|---|---|---|---|---|
| | | 正解 | 不正解 | |
| 提案方式 | 正解 | 413.9 | 13.9 | 427.8 |
| | 不正解 | 25.8 | 18.4 | 44.2 |
| 合計 | | 439.7 | 32.3 | 472.0 |

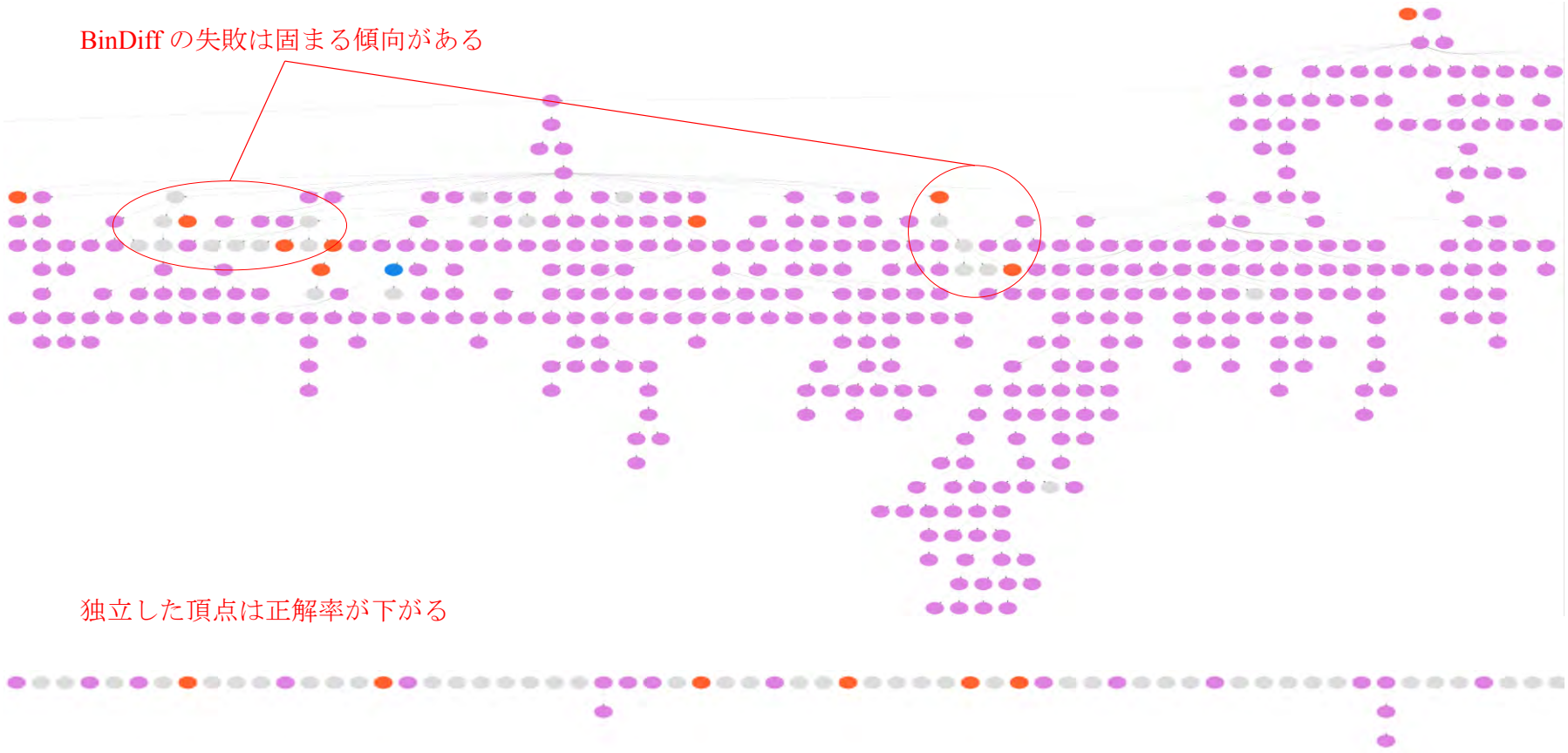


図 4-6　Emdivi の全関数に対する評価結果（一部）

Image 4-6　BinDiff and proposing method result for Emdivi

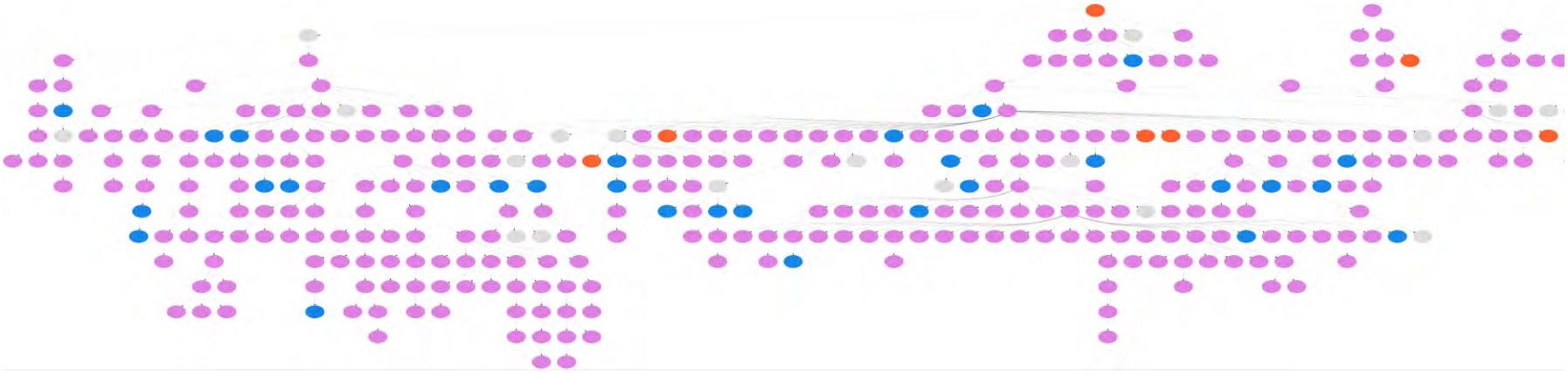

図 4-7　PlugX の全関数に対する評価結果（一部）

Image 4-7　BinDiff and proposal method result for PlugX

BinDiff は関数の呼び出しを辿り連鎖的に対応付けを行うため，コールグラフにおいて失敗した頂点が固まる傾向があった．一方で提案方式は関数呼び出しの関係に依存しないため，失敗した頂点は散らばった．

関数呼び出しは call 命令によって行われるが, ジャンプ先のアドレスが動的に計算される場合がある．このような場合はコールグラフにおいて辺を設定することができず，独立した関数となる. このような関数において BinDiff は呼び出し関係を利用することができないため失敗が目立っていた．

## 4.7　議論

解析者が手動で関数の検索を行う場合は逆アセンブリを先頭から読む必要があるため，相当の作業量がかかることが想定されるが，提案方式を使用すると解析者にとって負担にならない時間で目的の関数を特定することができる．インシデントレスポンスにおけるマルウェア解析に活用した場合，マルウェアの通信や暗号処理に関わる関数を特定して，その中の処理を過去のコードと比較して差分を解析するが，例えば今回の評価に使用した Emdivi では解析者が一見するだけで過去のマルウェアと同じ暗号方式が使用されていることが特定できている．さらに，通信先や暗号方式が厳密に特定できるかどうかでインシデントレスポンスの方針が大きく変わることも考えられる．

一方で，解析対象マルウェアが過去に解析したどのマルウェアに類似するかどうかは解析者が特定する必要がある．新たに開発されたマルウェアや機能については適用できない．また，関数構造を持たないマルウェアや極端に難読化されたマルウェアに対しても提案方式を適用できない．パッキングされたマルウェアや他のプログラムにインジェクトされた実行コードは，メモリーから実行コードを取り出してから逆アセンブルした上で提案方式を適用する必要がある．

## 4.8　小括

近年，日本においても APT 攻撃による大規模な被害を経験し，インシデントレスポンス

の重要性が再認識された．深刻なインシデントにおける詳細調査は解析作業を伴うため多大な時間とコストが必要となるが, “効果的”な対策が求められる場面においてはこの作業に取り組まなければならない．マルウェアの亜種が共通的に使用された場合，以前解析したマルウェアの関数に相当するコードの場所を特定できると，速やかに解析に取りかかることができる．本章では，過去に解析したマルウェアの関数の制御フローグラフと命令列を用いて相当する関数を検索する新しい方式（BinGrep）を提案し，“効率的”にマルウェア解析が行えることを示した．

提案方式と BinDiff について正常プログラムを用いて評価を行った．提案方式は複数の候補を出力するため上位から 10 位以内を正解と定義した．GNU bash と GNU binutils では，11,049 個の関数のうち 90.3% について正解を出力した．特に，BinDiff が失敗した 1,069 個の関数のうち 65.8%について提案方式が正解を出力し，BinDiff と提案方式を併用することでより多くの関数を特定できることを示せた．また，実際に APT 攻撃で使用された RAT である Emdivi と PlugX について評価を行った．Emdivi の 11 個の検体を用いて，プロキシサーバーのログ調査に必要となる HTTP リクエストを構成する処理と暗号処理を行う 27 個の関数を評価した結果，85%について正解を出力した．さらに，Emdivi と PlugX のそれぞれ 2 検体における全関数に対して評価を行い，Emdivi の 1,018 個の関数のうち 75%, PlugX の 472 個の関数のうち 90%について正解を出力し，提案方式がマルウェア解析において有効であることを示せた．

本方式は検索したい関数が存在しない場合でも関数の候補を上位から出力していたが，このような場合は該当する関数が存在しないことを示すアルゴリズムが求められる．また，本論文では評価プログラムとマルウェアにおいて関数をどれだけ正しく検索できるかを評価したが，提案方式を活用することで実際のインシデントレスポンスにおける作業量や精度の改善にどれだけ貢献できるか評価することも必要である．

# 第 5 章

# CSIRT に対する貢献と今後の展望

## 5.1 CSIRT に対する貢献

大規模な APT などによる攻撃を受けてサービスの停止や情報漏洩，信頼の失墜など組織の活動に大きな影響を与えてしまうという事件が日常的に発生している．サイバーセキュリティが経営課題として認識されている現代において，組織の成長戦略を実現するためには CSIRT が健全に機能することが重要なミッションとなっているが, CSIRT が行う業務の中でも特にインシデントレスポンスは緊急性と確実性が求められる課題となっている．そこで，本論文では CSIRT が行うインシデントレスポンスにおける課題を分析し，業務を“効果的”・“効率的”に行うための 2 つの技術を提案した．

1 つは，Web ブラックリストを活用するための分類の提案である．インシデントレスポンスの「トリアージ」「対応実施」の行程において合理的な思考に基づいた迅速な判断と適切な再発防止を行うため，Web ブラックリストを「所有者」「コンテンツ」「現在の状態」「最終確認」という 4 項目で分類することを提案した．この分類を用いることで「トリアージ」における端末特定の要否，次の調査項目，優先度といった判断が,「対応実施」における通信の遮断の要否といった判断がマニュアルに基づいて速やかに合理的に行えるようになった．

もう 1 つは，フォレンジックにおけるマルウェア解析を効率化する手法の提案である．インシデントレスポンスの「事象の分析」の行程において，過去に解析したマルウェアを用いて解析したいマルウェアの関数のコードの場所を特定する手法を提案した．これによ

り通信先や暗号処理の解析に速やかに取りかかることができるようになり，影響範囲が特定できるようになった．

これら 2 つの提案は現在の一般的な CSIRT が抱えるインシデントレスポンスの課題を解決しうることを示したが，将来的に CSIRT にとって最適な運用や CSIRT そのものの形態は時代に合わせて変化していくはずである．そこで，本章では CSIRT の将来の展望について述べ，その中での本論文の位置づけについて議論したい．

## 5.2 クラウド化による新たな CSIRT の課題

現在の CSIRT が抱えつつある課題として，クラウドサービスの利用の促進がある．組織が利用するシステムはこれまで自組織のデータセンターに構築していたが，現在は資産を持たずにパブリッククラウド上に実装，もしくはサービスとして利用する傾向がますます加速している．例えば，従業員の Web アクセスを代理して URL フィルタリングなどの機能を提供するプロキシサーバーがクラウドサービスとして提供される．クライアント PC にはプロキシサービスの IP アドレスを設定するだけで，簡単に利用することができる．ところが，従来のように自前で構築していたオンプレミスのシステムのような柔軟なセキュリティログの取得ができなくなってしまう場合がある．クライアントのプライベート IP アドレスが NAT によって同一のグローバル IP アドレスに変換されてしまい，クライアントが識別できなくなってしまうという問題がある．プロキシサーバーのユーザー認証を利用してアクセスを区別することもできるが，ユーザー認証を行わないマルウェアのアクセス元は区別することができず，感染端末を特定できない．また，パブリッククラウドサービスは多数のユーザーが利用する共用型のサービスであり機能のカスタマイズ性に乏しい．

クラウドが主流となる時代に向けて組織のガバナンスを維持するための仕組みとして，CASB（Cloud Access Security Broker）といったサービスや製品が提供され始めている [104]．企業が利用するクラウドサービスに対して認証やシングルサインオン，アクセス制御，データの暗号化，ログ取得などの機能を提供する．

また，ネットワークではなく端末上で検査，対処を行うことでマルウェア感染を直接的に検出するための EDR（Endpoint Detection and Response）といった製品も出現している

[105]．プログラムがアクセスした URL，作成したファイルやレジストリ，生成したプロセスや実行したプログラムや API などをチェックすることで不審な挙動をアラートとして出力する機能を持ち，さらにはリモートから端末を隔離してフォレンジックを行うことができる．クラウドサービスが導入された環境においては，このような機構を積極的に活用する必要性も出てくる．

組織ネットワークの内部と外部の境界が曖昧になってくると，内部不正といった問題もますます深刻な課題となってくる．UEBA（User and Entity Behavior Analytics）はエンドポイントやネットワークの様々なログを使用してユーザーの異常な行動を発見する製品である[106]．

クラウドサービスの利用が促進される時代において，CSIRT は従来の延長線上とは異なる機構を利用してインシデントレスポンスにあたる必要が出てくる．どのような技術が主流になったとしても，脅威情報に与えられる分類はサービス提供者目線でなく CSIRT 目線で与えられるべきものであり，インシデントレスポンスにおいて合理的に意思決定するために必要となる情報の在り方として本論文が貢献できると考えている．

## 5.3 CSIRT が対象とするスコープの拡大

中期的な展望として，CSIRT は形を変えながら発展すると考えられる．組織が存続するために考えなければならないセキュリティは，TCP/IP ネットワークに接続された汎用 OS のコンピューターだけとは限らない場合もある．データの分析や処理を行うためのコンピューター技術を総称して IT（Information Technology）と呼んでいたのに対して，組み込み装置などの制御システムを運用するハードウェアに働きかける技術を総称して OT（Operational Technology）と呼ばれている．生産ラインや社会インフラを支える OT についてもサイバーセキュリティの脅威に関わる問題は重要である．かつて IT と OT は別の技術であって，お互いのネットワークには「エアギャップ」によって隔離されているため安全であると認識されていたが，技術が成熟するに従って IT と OT が相互に影響するようになり，境界が徐々に薄まってきている．2010 年に発生したイランの核燃料施設を狙った Stuxnet や 2017 年に爆発的に感染が広まった WannaCry に代表されるように，IT と OT は

独立したものとは見なせなくなっている．組織は IT だけでなく OT までをスコープに含めたセキュリティを考える必要に迫られている．

また，組織の運営や提供するサービスにおけるセキュリティに責任を持つ CSIRT に対し，組織が提供する製品のセキュリティに責任を持つ PSIRT（Product Security Incident Response Team）という組織がある[107]．自社製品に関連する情報セキュリティ事故の未然防止と，インシデント発生時の対応を目的とした社内専門チームである．FIRST が制定したフレームワークの中では PSIRT は CSIRT とは別の機能であることが明言されているが，お互いに連携して相乗効果を発揮することが期待されている[108]．どちらも組織の持続的な発展のためのセキュリティの営みであることに変わりはなく，次第に CSIRT と PSIRT を包括するメタな概念によって同じ組織に統合されていくと考えている．

この状況から伺えるように，組織が持続的に発展するために考えなければならないセキュリティのスコープはますます拡大していくはずであり，CSIRT は形を変えながら発展すると考えられる．本論文では x86，x64 環境で動作する Windows マルウェアを対象としたが，その他のコンピューター環境で動作するマルウェアも関数やモジュールといった構造を持つ限り，同様の手法が適用できると考えている．

## 5.4 AI 主体の CSIRT

さらなる将来の展望として，究極的には CSIRT 機能が全て自動化され人間が関与する余地のない世界が実現する可能性も目指すことができる．現在も，アンチウイルス機能のように一部の用途に機械学習や深層学習を用いた，いわゆる AI（Artificial Intelligence）を活用したと謳う製品が実用化されている．将来的にはマルウェアの判別だけでなく，インシデント判定，影響調査，再発防止策の実施，自動復旧までインシデントレスポンスの全ての行程の完全な自動化を実現できる AI が出現するかもしれない．現在のような機械学習や深層学習を用いたアプローチは大量の教師データを必要とするうえ，正解となる根拠を出力することが難しく，日進月歩のセキュリティの世界においてプロフェッショナルが運用する CSIRT の機能を真に肩代わりする AI 技術はまだ出現していないが，ブレークスルーとなる新しい AI 技術が出現して理想的なインシデントレスポンスの実現を目指す時代の到

来も十分に考えられる．そのような AI を設計する際には，CSIRT が行うべき正しい対応を正解として定義する必要があるが，本論文におけるインシデントレスポンス業務の課題分析の考え方が貢献できると考える．

# 第 6 章

# 結論

本論文では，経営課題として一般的に認識されるようになったサイバーセキュリティの発展に寄与することを目標に，CSIRT のインシデントレスポンスをより“効果的”・“効率的”に実現するための技術を検討した．要約と貢献は以下の通りである．

第 1 章では，本論文の背景と目的について述べた．組織としてより高度なサイバーセキュリティを実現していく重要性を説明し，それを実現するためには CSIRT の能力が要となることを述べた．

第 2 章では，CSIRT の業務における課題分析を行った．CSIRT に求められる役割と，現状の CISRT が抱える一般的な問題意識を整理し，現在行われている取り組みや研究についてサーベイを行った．CISRT における問題意識の中でもインシデントレスポンスに注目し，「トリアージ」「事象の分析」「対応計画」「対応実施」という 4 つの行程で業務を分析し，2 つの課題を取り上げて説明した．1 つは，「トリアージ」「対応実施」の対応において，合理的な思考に基づいた迅速な判断と適切な再発防止が行われていないこと，もう 1 つは「事象の分析」の対応において，フォレンジックにおけるマルウェア解析の際に類似マルウェアの情報が十分に活用されておらず時間を要してしまうことを指摘した．

第 3 章では，ブラックリストにマッチしたアクセスに対する CSIRT の対応を合理的かつ迅速に行うという課題について検討を行った．合理的に判断するための情報を事前に与えることで，事象を正しく把握して“効果的”な対応を可能とすること，マニュアルに基づいた判断により時間をかけずに“効率的”なトリアージを可能とすることを目指し，Web ブラックリストを「所有者」「コンテンツ」「現在の状態」「最終確認」という 4 項目で分類するこ

とを提案した．ブラックリストの作成者はなぜ悪性と判断したのかその理由が分かるはずである．実際に，公開されている 38 種類のブラックリストを調査したところ，完全に求める情報が含まれていたブラックリストは存在しなかったが，それぞれのブラックリストにおいて断片的に存在していた．これは提案する分類に関わる情報を得ることの実現可能性を示している．また，実際にブラックリストに一致してインシデント判定が求められた 400 件の悪性 Web サイト情報について手動で調査を行い，一定数の Web サイト情報が外部からの調査によって定義に従った分類ができることを示した．この分類を活用することで速やかな対応ができることをケーススタディとして紹介し，提案分類が有効に働くことを示した．

第 4 章では, 発生した深刻なインシデントに対する“効果的”な対策のためにマルウェア解析を行う状況を想定し, 過去に解析したマルウェアファミリーを用いて“効率的”にマルウェア解析を行うため，解析したいマルウェアの関数のコードの場所を特定する手法を提案した．従来手法である BinDiff に代表されるパッチ解析用のコード「比較」ツールを使用した場合の課題を示し，制御フローグラフの編集距離と命令列の最長共通部分列を用いて関数を「検索」する BinGrep を提案した．BinGrep は，GNU bash と binutils では 11,049 個の関数のうち 90.3% について正解を出力できた．実際に APT 攻撃で使用されたマルウェアで評価したところ，Emdivi の 11 検体の評価では，インシデントレスポンスにおいて重要であった 27 個の関数の 85%について正解を出力できた．また，Emdivi と PlugX のそれぞれ 2 検体の全関数について評価を実施し，マルウェア解析において提案方式が有効であることを評価し，“効率的”に解析作業に取りかかることを示した．

第 5 章では，時代とともに出現する新しい技術や意識の変化に伴って変わっていくであろう CSIRT の将来像についてその展望と，本論文の位置づけについて議論した．現在直面している課題としてクラウドサービス利用の促進を取り上げ，脅威が見えなくなりつつあるといった現在の CSIRT 業務に与えている影響と技術を考察した．また，組織が対応していかなければならないサイバーセキュリティとして OT への対応や PSIRT といった新しい役割が求められていることを説明し，近い将来には CSIRT の役割や体制も変化することを予想した. 究極的な理想像として CSIRT は AI 技術を用いて全て自動化されるべきである．このような展望において，本論文で提案した 2 つの技術の関わりについて議論した．

# 参考文献


[1] 日本シーサート協議会：CSIRT：構築から運用まで，エヌティティ出版（2016）.

[2] 寺本直城，杉浦芳樹，林郁也，矢寺顕行，福本俊樹，近藤光，杉原大輔：我が国における CSIRT の現状と課題，経営情報学会 2015 年秋季全国研究発表大会，Vol. 2015f，pp.57-60（2015）.

[3] JPCERT/CC：CSIRT ガイド（オンライン），入手先〈https://www.jpcert.or.jp/csirt_material/files/guide_ver1.0_20151126.pdf〉（参照 2018-10-12）.

[4] FIRST：FIRST - Improving Security Together (Online), available from〈https://www.first.org〉(accessed 2018-10-12).

[5] APCERT：APCERT (Online), available from 〈https://www.apcert.org〉(accessed 2018-10-12).

[6] 日本シーサート協議会：CSIRT - 日本シーサート協議会（オンライン），入手先〈http://www.nca.gr.jp〉(参照 2018-10-12).

[7] 杉原大輔, 日本における企業内 CSIRT の現状と課題 －NCA への早期加盟チームの実態から－（オンライン），入手先〈https://www.jstage.jst.go.jp/article/kaichi/17/0/17_5/_pdf/-char/ja〉(参照 2018-10-12).

[8] 寺島健一, 日本企業における CSIRT 構築の事例 －C 社 CSIRT 構築における制度変化－, 経営情報学会 2013 年秋季全国研究発表大会, Vol. 2013f, pp.277-280（2013）.

[9] West-Brown, M. J., Stikvoort, D., Kossakowski, Klaus-Peter, Killcrece, G., Ruefle, R. and Zajicek, M.: Handbook for Computer Security Incident Response Teams (CSIRTs) 2nd ed., Carnegie Mellon University (Online), available from〈https://resources.sei.cmu.edu/asset_files/Handbook/2003_002_001_14102.pdf 〉（ accessed

2018-10-12).

[10] ENISA: A step-by-step approach on how to setup a CSIRT, available from〈 https://www.sbs.ox.ac.uk/cybersecurity-capacity/system/files/ENISA%20-%20CSIRT%20setting%20up%20guide.pdf〉(accessed 2018-10-12).

[11] NHA Rahman and KKR Choo: A State-of-the-Art Survey on Computer Security Incident Handling, Journal in Computers and Security, Vol.49, Issue C, pp.45-69 (2015).

[12] 平井達哉，本川祐治，佐々木慎一，丹京真一：企業における CSIRT の活動とそれを支援する情報共有システム，情報処理学会デジタルプラクティス，Vol.9，No.3，pp.627-642 (2018).

[13] 長谷川皓一，山口由紀子，嶋田創，高倉弘喜：標的型攻撃に対するインシデント対応支援システム，情報処理学会論文誌，Vol.57，No.3，pp.836-848 (2016).

[14] 川口信隆，築地原護，井手口恒太，谷川嘉伸，冨村英勤：不審活動の端末間伝搬に着目した標的型攻撃検知方式，情報処理学会論文誌，Vol.57，No.3，pp.1022-1039 (2016).

[15] 三村守, 田中英彦：多変量解析による標的型攻撃の分類, 情報処理学会論文誌, Vol.54, No.12，pp.2461-2471 (2013).

[16] 渡部正文，島成佳，今橋泰則，吉岡克成，高木大資：企業グループに送られた標的型攻撃メールのソーシャルエンジニアリング視点からの分析, 情報処理学会論文誌, Vol.57，No.12，pp.2731-2742 (2016).

[17] Microsoft: Security Intelligence Report Volume 22 (online), available from〈https://www.microsoft.com/en-us/security/intelligence-report〉(accessed 2017-12-10).

[18] CERT-MU: THE WANNACRY RANSOMWARE (online), available from〈http://cert-mu.govmu.org/English/Documents/White%20Papers/White%20Paper%20-%20The%20WannaCry%20Ransomware%20Attack.pdf〉(accessed 2017-12-10).

[19] SANS Institute: Using IOC (Indicators of Compromise) in Malware Forensics (online), available from〈https://www.sans.org/reading-room/whitepapers/forensics/ioc-indicators-compromise-malware-forensics-34200〉(accessed 2017-12-10).

[20] abuse.ch: SSL Blacklist (online), available from 〈https://sslbl.abuse.ch〉 (accessed 2017-12-10).

[21] Xylitol: Cybercrime Tracker (online), available from 〈http://cybercrime-tracker.net〉 (accessed 2017-12-10).

[22] Malware Domain List (online), available from 〈http://www.malwaredomainlist.com〉 (accessed 2017-12-10).

[23] RiskAnalytics: Malware-Domains (online), available from 〈http://www.malware-domains.com〉 (accessed 2017-12-10).

[24] Blue Coat: Blue Coat WebPulse (online), available from 〈https://www.bluecoat.com/sites/default/files/editor_files/bcs_WebPulse_Tech_Overview_wp_v1b.pdf〉 (accessed 2017-12-10).

[25] FFRI: BlackURL (online), available from 〈http://www.ffri.jp/assets/files/services/ctrg/BlackURL_brochure.pdf〉 (accessed 2017-12-10).

[26] Cisco: IronPort AsyncOS 7.7.5 for Web (online), available from 〈https://www.cisco.com/cisco/web/support/JP/docs/SEC/WebSecur/WebSecurAppliance/UG/002/WSA_7.7.5_UserGuide-J.pdf〉 (accessed 2017-12-10).

[27] FireEye: iSIGHT Intelligence Subscriptions (online), available from 〈https://www.fireeye.com/products/isight-cyber-threat-intelligence-subscriptions.html〉 (accessed 2017-12-10).

[28] PaloAlto Networks: AutoFocus (online), available from 〈https://www.paloaltonetworks.com/products/secure-the-network/subscriptions/autofocus〉 (accessed 2017-12-10).

[29] Nappa, A., Rafique, MZ. and Caballero, J.: Driving in the Cloud: An Analysis of Drive-by Download Operations and Abuse Reporting, Proc. 10th International Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2013), pp.1-20, Springer (2013).

[30] Kotov, V., Massacci, F.: Anatomy of Exploit Kits Preliminary Analysis of Exploit Kits as Software Artefacts, Proc. 5th international conference on Engineering Secure

Software and Systems (ESSoS 2013), pp.181-196, ACM (2013).

[31] Malwarebytes Labs: State of Malware Report 2017 (online), available from 〈https://www.malwarebytes.com/pdf/white-papers/stateofmalware.pdf〉(accessed 2017-12-10).

[32] A10 Networks: Thunder SSLi (online), available from 〈https://www.a10networks.com/products/thunder-series/ssl-decryption-encryption-and-inspection-ssl-insight〉(accessed 2017-12-10).

[33] PaloAlto Networks: Blocking Suspicious DNS Queries with DNS Proxy Enabled (online), available from 〈https://live.paloaltonetworks.com/t5/Management-Articles/Blocking-Suspicious-DNS-Queries-with-DNS-Proxy-Enabled/ta-p/66037〉(accessed 2017-12-10).

[34] JPCERT/CC：インシデントハンドリングマニュアル（オンライン），入手先〈https://www.jpcert.or.jp/csirt_material/files/manual_ver1.0_20151126.pdf〉(参照 2017-12-10).

[35] CrowdStrike: Falcon Intelligence (online), available from 〈https://www.crowdstrike.com/products/falcon-intelligence〉(accessed 2017-12-10).

[36] ZeroCERT: Safeguard (online), available from 〈https://center.zerocert.org/safeguard〉(accessed 2017-12-10).

[37] abuse.ch: ZeuS Tracker (online), available from 〈https://zeustracker.abuse.ch〉(accessed 2017-12-10).

[38] ALIEN VAULT: ThreatCrowd (online), available from 〈https://www.threatcrowd.org〉(accessed 2017-12-10).

[39] NoVirusThanks: Hijacked Urls Database (online), available from 〈http://www.hijackedurls.com〉(accessed 2017-12-10).

[40] abuse.ch: Ransomware Tracker (online), available from 〈https://ransomwaretracker.abuse.ch〉(accessed 2017-12-10).

[41] Web Security Guard: Website Database (online), available from 〈http://www.websecurityguard.com/database.aspx〉(accessed 2017-12-10).

[42] Malc0de: Malc0de Database (online), available from 〈http://malc0de.com/database〉(accessed 2017-12-10).

[43] MITRE: Standardizing Cyber Threat Intelligence Information with the Structured Threat Information eXpression (STIX) (online), available from〈https://www.mitre.org/sites/default/files/publications/stix.pdf〉(accessed 2017-12-10).

[44] Tanaka, Y. and Goto, A.: Suspicious FQDN Evaluation based on Variations in Malware Download URLs. Proc. IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2017), pp.811-818 (2017).

[45] JPCERT/CC：インシデント報告対応レポート（オンライン），入手先〈https://www.jpcert.or.jp/pr/2018/IR_Report20180116.pdf〉(参照 2018-04-01).

[46] MISP project: Open Source Threat Intelligence Platform & Open Standards For Threat Information Sharing (online), available from〈http://www.misp-project.org〉(accessed 2017-12-10).

[47] ENISA: Threat Taxonomy. A tool for structuring threat information (online), available from〈https://www.enisa.europa.eu/topics/threat-risk-management/threats-and-trends/enisa-threat-landscape/etl2015/enisa-threat-taxonomy-a-tool-for-structuring-threat-information〉(accessed 2017-12-10).

[48] eCSIRT: European CSIRT Network IST-2001-37558 (online), available from〈http://www.ecsirt.net/eCSIRT-WP1-final-report.pdf〉(accessed 2017-12-10).

[49] FIRST: CSIRT Case Classification (online), available from〈https://www.first.org/resources/guides/csirt_case_classification.html〉(accessed 2017-12-10).

[50] CIRCL: Taxonomy - Schemes of Classification in Incident Response and Detection (online), available from 〈 https://www.circl.lu/pub/taxonomy 〉 (accessed 2017-12-10).

[51] Kuhrer, M., Holz, T.: An Empirical Analysis of Malware Blacklists, Praxis der Informationsverarbeitung und Kommunikation (PIK 2012), Vol.35, Issue 1, pp.11?16 (2012).

[52] Kuhrer M., Rossow C. and Holz T.: Paint it Black: Evaluating the Effectiveness of

Malware Blacklists, Proc. 17th International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2014), pp.1-21, Springer-Cham (2014).

[53] Sheng, S., Wardman, B., Warner, G., Cranor, LF. and Hong, J.: An Empirical Analysis of Phishing Blacklists, Proc. 6th Conference on Email and Anti-Spam (CEAS 2009).

[54] Antonakakis, M., Perdisci, R., Lee, W., Vasiloglou II, N. and Dagon, D.: Detecting Malware Domains at the Upper DNS Hierarchy, Proc. 20th USENIX Conference on Security (SEC 2011), pp.27, Berkeley (2011).

[55] Antonakakis, M., Perdisci, R., Nadji, Y., Vasiloglou, N., Abu-Nimeh, S., Lee, W. and Dagon, D.: From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware, Proc. 21st USENIX conference on Security symposium (Security 2012), pp.24, Berkeley (2012).

[56] Bilge, L., Kirda, E., Kruegel, C. and Balduzzi, M.: EXPOSURE: Finding Malicious Domains Using Passive DNS Analysis, Proc. 18th Annual Network and Distributed System Security Symposium (NDSS 2011).

[57] Rahbarinia, B., Perdisci, R., Antonakakis, M. and Dagon, D.: SinkMiner: Mining Botnet Sinkholes for Fun and Profit. Proc. 6th USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET 2013).

[58] Sinha, S., Bailey, M. and Jahanian, F.: Shades of Grey: On the effectiveness of reputation-based “blacklists”, Proc. 3rd International Conference on Malicious and Unwanted Software (MALWARE 2008), pp.57-64 (2008).

[59] Rossow, C., Czerwinski, T., Dietrich, CJ. and Pohlmann, N.: Detecting Gray in Black and White, MIT Spam Conference 2010 (SC 2010).

[60] Thomas, K., Grier, C., Ma, J., Paxson, V. and Song, D.: Design and Evaluation of a Real-Time URL Spam Filtering Service. Proc. IEEE Symposium on Security and Privacy (SP 2011), pp.447-462 (2011).

[61] SANS Institute: InfoSec Reading Room, Malware 101-Viruses (online), available from 〈https://www.sans.org/reading-room/whitepapers/incident/malware-101-vir

uses-32848〉(accessed 2018-04-04).

[62] Department of the Army: FM 2-22.3 (FM 34-52) HUMAN INTELLIGENCE COLLECTOR OPERATIONS (online), available from 〈https://fas.org/irp/doddir/army/fm2-22-3.pdf〉(accessed 2018-04-04).

[63] 株式会社カスペルスキー：BLUE TERMITE ～日本を標的にする APT 攻撃～（オンライン），入手先〈http://media.kaspersky.com/jp/pdf/pr/Kaspersky_BlueTermiteDaily-PR-1016.pdf〉（参照 2016-08-20）.

[64] 朝長秀誠，中村祐：CODE BLUE 2015 日本の組織をターゲットにした攻撃キャンペーンの詳細，JPCERT/CC（オンライン），入手先〈https://www.jpcert.or.jp/present/2015/20151028_codeblue_ja.pdf〉（参照 2016-08-20）.

[65] 船越絢香，中村祐，竹田春樹：標的型攻撃で用いられたマルウェアの特徴と攻撃の影響範囲の関係に関する考察，コンピュータセキュリティシンポジウム 2015 論文集（CSS2015), vol.2015, No.3, pp.963-970 (2015).

[66] Kent, K., Chevalier, S., Grance, T. and Dang, H.: Guide to Integrating Forensic Techniques into Incident Response, National Institute of Standards and Technology (online), available from〈http://csrc.nist.gov/publications/nistpubs/800-86/SP800-86.pdf〉(accessed 2016-03-06).

[67] 新井悠，岩村誠，川古谷裕平，青木一史，星澤裕二：アナライジング・マルウェア フリーツールを使った感染事案対処，オライリー・ジャパン（2010）.

[68] JPCERT/CC：ログを活用した高度サイバー攻撃の早期発見と分析（オンライン），入手先〈https://www.jpcert.or.jp/magazine/acreport-emdivi.html〉（accessed 2016-12-25).

[69] JPCERT/CC：高度サイバー攻撃への対処におけるログの活用と分析方法（オンライン），入手先〈https://www.jpcert.or.jp/magazine/acreport-emdivi.html〉(accessed 2016-12-25).

[70] Grobert, F., Willems, C. and Holz, T.: Automated identification of cryptographic primitives in binary programs, Proc. 14th International Symposium Recent Advances in Intrusion Detection (RAID 2011), pp.41-60, Springer (2011).

[71] Calvet, J., Fernandez, J. M. and Marion, J.: Aligot: Cryptographic Function Identification in Obfuscated Binary Programs, Proc. ACM Conference on Computer and Communications Security (CCS 2012), pp.169-182, ACM (2012).

[72] Percival, C.: Naive differences of executable code, Binary diff/patch utility (online), available from 〈 http://www.daemonology.net/papers/bsdiff.pdf 〉 (accessed 2016-08-20).

[73] MacDonald, J.: Open-source binary diff differential compression tools VCDIFF (RFC 3284) delta compression, xdelta (online), available from〈http://xdelta.org〉 (accessed 2016-03-06).

[74] Heirbaut, J.: Diff utility for binary files, JojoDiff (online), available from 〈http://jojodiff.sourceforge.net〉(accessed 2016-08-20).

[75] Flake, H.: Structural Comparison of Executable Objects, Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2004), IEEE Computer Society, pp.161-173 (2004).

[76] Dullien, T. and Rolles, R.: Graph-based comparison of Executable Objects, Proc. of SSTIC 2005 (2005).

[77] Zynamics: Zynamics BinDiff (online), available from 〈http://www.zynamics.com/bindiff.html〉(accessed 2016-08-20).

[78] Bourquin, M., King, A. and Robbins, E.: BinSlayer: Accurate Comparison of Binary Executables, Proc. 2nd ACM SIGPLAN Program Protection and Reverse Engineering Workshop (PPREW 2013), ACM (2013).

[79] Gao, D., Reiter, M. K. and Song, D.: Binhunt: Automatically finding semantic differences in binary programs, Proc. 10th International Conference on Information and Communications Security (ICICS 2008), pp.238-255, Springer (2008).

[80] Ming, J., Pan, M. and Gao, D.: iBinHunt: Binary Hunting with Inter-procedural Control Flow, Proc. Information Security and Cryptology (ICISC 2012), pp.92-109, Springer (2012).

[81] Oh, J.: Fight against 1-day exploits: Diffing binaries vs anti-diffing binaries, Proc.

Black Hat USA 2009 (2009).

[82] Tenable Network Security: PachDiff2 High Performance Patch Analysis (online), available from〈https://www.tenable.com/blog/patchdiff2-high-performance-patch-analysis〉(accessed 2016-08-20).

[83] eEye Digital Security: eEye Binary Diffing Suite (online), available from〈https://web.archive.org/web/20080705014733/http://research.eeye.com/html/tools/RT20060801-1.html〉(accessed 2016-08-20).

[84] Zimmer, D.: IDACompare, VeriSign iDefense Labs (online), available from〈http://sandsprite.com/iDef/IDACompare/〉(accessed 2016-08-20).

[85] LeDoux, C., Lakhotia, A., Miles, C. and Notani, V.: FuncTracker: Discovering Shared Code to Aid Malware Forensics Extended Abstract, Proc. 6th USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET 2013) (2013).

[86] Ruttenberg, B., Miles, C., Kellogg, L., Notani, V., Howard, M., LeDoux, C., Lakhotia, A. and Pfeffer, A.: Identifying Shared Software Components to Support Malware Forensics, Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA 2014), pp.21-40, Springer (2014).

[87] Pewny, J., Schuster, F., Rossow, C., Bernhard, L. and Holz, T.: Leveraging Semantic Signatures for Bug Search in Binary Programs, Proc. 30th Annual Computer Security Applications Conference Pages (ACSAC 2014), pp.406- 415, ACM (2014).

[88] 中島明日香，岩村誠，矢田健：機械語命令の類似度算出による複製された脆弱性の発見手法の提案，コンピュータセキュリティシンポジウム2015論文集（CSS2015），vol.2015，No.3，pp.304-309 (2015).

[89] 岩村誠，伊藤光恭，村岡洋一：機械語命令列の類似性に基づく自動マルウェア分類システム，情報処理学会論文誌，Vol.51，No.9，pp.1622-1632 (2010).

[90] Karim, M. E., Walenstein, A., Lakhotia, A. and Parida, L.: Malware phylogeny generation using permutations of code. Journal of Computer Virology, Vol.1, Issue 1-2, pp.13-23, Springer-Verlag (2005).

[91] Gheorghescu, M.: An automated virus classification system, Virus Bulletin

Conference 2005 (2005).

[92] 岩本一樹, 和崎克己：静的解析により抽出された API 推移に基づくマルウェアの分類, 情報処理学会論文誌, Vol.54, No.3, pp.1199-1210 (2013).

[93] Hu, X., Chiueh, T. and Shin, K. G.: Large-Scale Malware Indexing Using Function-Call Graphs, Proc. 16th ACM conference on Computer and communications security (CCS 2009), pp.611-620, ACM (2009).

[94] Kinable, J. and Kostakis, O.: Malware Classification based on Call Graph Clustering, Journal in Computer Virology, Vol.7, Issue 4, pp.233-245, Springer-Verlag (2011).

[95] Levi, G.: A Note on the Derivation of Maximal Common Subgraphs of Two Directed or Undirected Graphs, Calcolo, Vol.9, pp.341-354, Springer-Verlag (1973).

[96] Gao, X., Xiao, B., Tao, D. and Li, X.: A survey of graph edit distance, Pattern Analysis and Applications, Vol.13, Issue 1, pp.113-129, Springer-Verlag (2010).

[97] Zhang, K. and Shasha, D.: Simple fast algorithms for the editing distance between trees and related problems, SIAM Journal of Computing, Vol.18, Issue 6, pp.1245-1262 (1989).

[98] Klein, P.: Computing the edit-distance between unrooted ordered trees, Proc. 6th Annual European Symposium on Algorithms, pp.91-102, Springer-Verlag (1998).

[99] Dulucq, S. and Touzet, H.: Analysis of tree edit distance algorithms, Proc. 14th annual conference on Combina¬torial Pattern Matching (CPM 2003), pp.83-95, Springer (2003).

[100] Hex-Rays: IDA Pro (online), available from 〈http://www.hex-rays.com〉 (accessed 2016-03-06).

[101] Henderson, T. and Johnson, S.: Zhang-Shasha: Tree edit distance in Python, available from 〈https://zhang-shasha.readthedocs.org/en/latest/〉 (accessed 2016-08-20).

[102] Cython: C-Extention for Python (online), available from 〈http://cython.org/〉 (accessed 2016-08-20).

[103] JPCERT/CC：Emdivi が持つ暗号化された文字列の復号（オンライン), 入手先

〈https://www.jpcert.or.jp/magazine/acreport-emdivi.html〉(accessed 2016-12-25).

[104] Gartner: Cloud Access Security Brokers (CASBs) (online), available from〈https://www.gartner.com/it-glossary/cloud-access-security-brokers-casbs/〉(accessed 2018-12-01).

[105] Gartner: Endpoint Detection and Response Solutions Reviews (online), available from〈https://www.gartner.com/reviews/market/endpoint-detection-and-response-solutions〉(accessed 2018-12-01).

[106] eSecurityPlanet: UEBA: Protecting Your Network When Other Security Systems Fail (online), available from〈https://www.esecurityplanet.com/network-security/user-and-entity-behavior-analytics-ueba.html〉(accessed 2018-12-01).

[107] FIRST: PSIRT Services Framework Version 1.0 (online), available from〈https://www.first.org/education/Draft_FIRST_PSIRT_Service_Framework_v1.0 〉 (accessed 2018-12-01).

[108] 後藤厚宏，伊藤公祐：サイバーセキュリティの技術展望～セキュアな IoT 社会に向けた取り組み～，行政&情報システム 2018 年 12 月号，pp.27-33 (2018).

[109] Felegyhazi, M., Kreibich, C. and Paxson, V.: On the Potential of Proactive Domain Blacklisting. Proc. 3rd USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET 2010).

[110] Kolbitsch, C., Comparetti, P. M., Kruegel, C., Kirda, E., Zhou, X. and Wang, X.: Effective and Efficient Malware Detection at the End Host, Proc. 18th USENIX Conference on Security (SEC 2009), pp.351-366 (2009).

[111] Bartos, K., Sofka, M. and Franc, V.: Optimized Invariant Representation of Network Traffic for Detecting Unseen Malware Variants, Proc. 25th USENIX Conference on Security (SEC 2016), pp.807-822 (2016).

[112] Saltaformaggio, B., Gu, Z., Zhang, X. and Xu, D.: DSCRETE: Automatic Rendering of Forensic Information from Memory Images via Application Logic Reuse, Proc. 23rd USENIX Conference on Security (SEC 2014), pp.255-269 (2014).

[113] Webster, A., Eckenrod, R. and Purtilo, J.: Fast and Service-preserving Recovery from

Malware Infections Using CRIU, Proc. 27th USENIX Conference on Security (SEC 2018), pp.1199-1211 (2018).

[114] 山崎勇二，後藤田 中，小野滋己，青木有香，八重樫理人，藤本憲市，林敏浩，今井慈郎，最所圭三：持続可能な大学 CSIRT を目指した対応訓練システムの開発，大学 ICT 推進協議会 2017（AXIES 2017）(2017).

[115] Eric, M. H., Michael, J. C. and Rohan, M. A.: Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains (online), available from〈https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf〉(accessed 2019-02-03).

# 謝辞

本論文は，私が情報セキュリティ大学院大学情報セキュリティ研究科情報セキュリティ専攻博士後期課程に在籍していた期間に取り組んだ研究をまとめたものです．本論文の執筆にあたり，後藤厚宏教授には指導教官として日頃より熱心にご指導をいただきました．先生からは，私が 2011 年に社会人学生として修士課程に入学してセキュリティのプロフェッショナルを目指し始めた頃からご指導いただきました．研究の方向性や進め方から論文の書き方までいつも親身に相談に乗っていただき，そのたびに研究成果の高みを目指すアドバイスをいただきました．博士後期課程へチャレンジし，研究者としてのキャリアパスを目指したいと考えるきっかけとなったのも先生の存在が何よりでした．

職場である NTT セキュリティ・ジャパン株式会社では，博士後期課程在学期間中，多くの方々からのご協力をいただきました．同大学院博士前期課程を卒業して現場業務に完全復帰してからしばらくして，忙しくも充実感に満ち溢れていた当時を思い出して再びアカデミックでの活動を続けていきたいと考えるようになった頃，明言しなかったにも関わらず当時の上司の横山恵一氏はこの想いを察して会社に働きかけ，博士後期課程への企業派遣の枠組みを作っていただきました．現在の上司である遠藤孝弘氏には，社外での活動を快く認めていただき，また社外での活動が情報発信業務として私のミッションとリンクするようにチーム編成の組み換えを行うなど，様々な配慮をしていただきました．森慎一氏にはセキュリティポリシーの有識者としてグローバルの動向や事例など豊富な知見をもとに助言をいただきました．阿部慎司氏には，学会発表や講演で出張するたびに 24 時間体制の分析サービスの現場のマネジメントをお願いしてしまいましたが，いつも快く全てのエスカレーションを引き受けていただいたおかげで，安心して発表に専念することができました．幾世知範氏には，日々のちょっとした疑問やアイデアなど雑多なトピックに対して気軽に相談させていただきました．BlackHat Europe での発表はそうした議論から実現しました．ここに挙げることができなかった職場の上司・同僚の方々にも多くのご支援をいた

だきました．皆様のサポートなくして本研究を続けることはできませんでした．深く感謝いたします．

NTT コミュニケーションズ株式会社の畑田充弘氏には，博士後期課程へのチャレンジだけでなく，マルウェア対策研究人材育成ワークショップ（MWS）や CSEC 研究会など，何かとアカデミック分野での活動を後押ししていただきました．日常業務の延長にはない立場を経験する貴重な機会をいただいただけでなく，多くの人と知り合って研究活動がより充実したものとなりました．

ゼミや輪講，合宿などで様々な助言や研究の励みをいただきました情報セキュリティ大学院大学の皆様にも重ねて感謝いたします．

最後に，毎日帰りが遅く休日も研究活動に専念していた私に対して健康を心配してくれて，いつも笑顔で生活面を支えてくれた妻に深く感謝いたします．

# 研究業績

## 査読付学術論文

A-1　羽田大樹，後藤厚宏．BinGrep: 制御フローグラフの比較を用いた関数の検索によるマルウェア解析の効率化の提案, 情報処理学会論文誌, Vol.58, No.5, pp.1151-1162, 2017.（受付日 2016 年 8 月 24 日, 採録日 2017 年 2 月 9 日）

A-2　羽田大樹，後藤厚宏．CSIRT のための Web ブラックリストの分類の提案，情報処理学会論文誌，Vol.59，No.9，pp.1-14，2018.（受付日 2017 年 12 月 11 日，採録日 2018 年 6 月 8 日）

## 査読付き国際会議

B-1　Hiroki Hada. BinGrep. BlackHat USA 2017 Arsenal.

B-2　Hiroki Hada and Tomonori Ikuse. CryptGrep: Rapidly Search a Cryptographic Function to Analyze Malware. BlackHat Europe 2018 Arsenal.

## 国内会議

C-1　羽田大樹，後藤厚宏．類似マルウェアとの差分の抽出による正確な挙動の解析手法についての検討．情報処理学会 第 74 回全国大会，March 2012.

C-2　羽田大樹，後藤厚宏．類似マルウェアとの差分の抽出による正確な挙動の解析手法についての検討．マルチメディア，分散，協調とモバイル 2012 (DICOMO 2012)，July 2012.

C-3　羽田大樹，後藤厚宏．解析済みマルウェアとの差分抽出による静的解析の効率化手法

の提案．コンピュータセキュリティシンポジウム 2012・マルウェア対策研究人材ワークショップ 2012 (CSS2012/MWS2012).

C-4 羽田大樹，後藤厚宏．解析済みマルウェアとの差分抽出による静的解析の効率化手法の提案．情報ネットワーク研究会（IN），March 2012.

C-5 羽田大樹，後藤厚宏．BinGrep: コールグラフを用いた関数の検索によるマルウェア解析の効率化の提案．暗号とセキュリティシンポジウム 2016（SCIS2016），4B1-2, January 2016.

C-6 羽田大樹，後藤厚宏．BinGrep: 制御フローグラフの比較を用いた関数の検索によるマルウェア解析の効率化の提案．第 74 回 CSEC 研究発表会, May 2016.

C-7 羽田大樹，後藤厚宏．BinGrep: 制御フローグラフの比較を用いた関数の検索によるマルウェア解析の効率化の提案．マルチメディア，分散，協調とモバイル 2016 (DICOMO 2016)，4F-5，July 2016.

C-8 羽田大樹，後藤厚宏．BinGrep: 制御フローグラフの比較を用いた関数の検索によるマルウェア解析の効率化の提案．コンピュータセキュリティシンポジウム 2016・マルウェア対策研究人材ワークショップ 2016 (CSS2016/MWS2016)，2B4-4，October 2016.

C-9 羽田大樹，後藤厚宏．CSIRT のための Web ブラックリストの分類の提案．暗号とセキュリティシンポジウム 2018 (SCIS2018), 3F4-3, January 2018.

C-10 羽田大樹，後藤厚宏．（招待論文）BinGrep：制御フローグラフの比較を用いた関数の検索によるマルウェア解析の効率化の提案，ソフトウェアエンジニアリングシンポジウム（SES) 2018.

C-11 柴田龍平，羽田大樹，横山恵一．Js-Walker: JavaScript API hooking を用いた解析妨害 JavaScript コードのアナリスト向け解析フレームワーク．コンピュータセキュリティシンポジウム 2016・マルウェア対策研究人材ワークショップ 2016 (CSS2016/MWS2016)，3B2-2，October 2016.

# その他

D-1　折原慎吾，鐘本楊，神谷和憲，松橋亜希子，阿部慎司，永井信弘，羽田大樹，朝倉浩志，田辺英昭．セキュリティのためのログ分析入門 サイバー攻撃の痕跡を見つける技術（Software Design plus シリーズ）．

D-2　佐々木良一（監修），電子情報通信学会（編）．現代電子情報通信選書「知識の森」 ネットワークセキュリティ（共著）．

D-3　C-7 において優秀プレゼンテーション賞を受賞．