

東京電機大学

博士論文

ニューノーマル時代に備えたマルウェアの分類  
手法に関する研究

Research on Classification of Malware  
for the New Normal Era

令和 05 年 度 3 月 18 日

周 家興



## あらまし

テレワークなどの新しい働き方が普及するにつれ、これまでのサイバーセキュリティ対策だけでは対応しにくく、見直す必要がでてきている。働き方、勤務環境と管理体制などの急速な変化に乗じた「ニューノーマルな働き方を狙った」サイバー攻撃が増加していることも一因にある。さらに、テレワークとオンライン会議などの利用環境の変化は、テレワーク用ソフトウェアなどの脆弱性問題を顕在化させ、二重脅迫は、ランサムウェアによる攻撃が金銭目当てとするサイバー攻撃としてのビジネス化を定着させたと言える。ニューノーマル時代における情報セキュリティ対策の課題は、マルウェアが関与あるいは起点としたソフトウェアの脆弱性悪用、ランサムウェアによる攻撃であり、対策を講じる必要がある。

本論文では、ニューノーマル時代に備え、顕在化したマルウェア脅威の特定と被害の低減のために、マルウェアの分類手法を提案する。顕在化したサイバーセキュリティの課題である、マルウェアが関与あるいは起点としたソフトウェアの脆弱性悪用、ランサムウェアによる攻撃に着目したマルウェアの分類手法を検討した後、ニューノーマル時代に向けたマルウェアの分類手法について、次の三つの研究を取りまとめる。

- テレワーク製品を狙ったマルウェアが攻撃対象としたソフトウェアの脆弱性を特定する研究
- ランサムウェアによる被害低減のためのファミリーを分類する研究
- ランサムウェアに限らず、マルウェア全般を対象としたファミリーを分類する研究

「マルウェアが攻撃対象とした脆弱性を特定する研究」では、静的解析を用いてIoTマルウェアが攻撃対象とした脆弱性を特定する。脆弱性を特定する手法については、公開サイトから提供されているマルウェア検体データセットを使って、その有効性を示す。その後、マルウェアファミリーとそれらが攻撃対象とした脆弱性との関係を示す。この手法を用いることにより、マルウェアが攻撃対象とした脆弱性を特定し、さらに攻撃手法と攻撃目的を把握することが可能となる。

「ランサムウェアによる被害低減のためのファミリーを分類する研究」では、ランサムウェアに呼び出されたAPIから各ファミリーの動作パターンを機械学習しファミリーを特定する手法についてである。複数サイトから提供されているランサムウェアのデータセットを使って特定する手法の有効性を示す。その後、ランサムウェアが攻撃に使用するAPIグループを明らかにすると共に、ランサムウェアファミリー毎に使用するAPIグループの傾向を示す。この手法を用いることにより、ランサムウェアの攻撃手法を特定し、ランサムウェアによる被害を抑止することが可能となる。

最後に、「マルウェア全般を対象としたファミリーを分類する研究」では、ランサ

マルウェアなど顕在化したマルウェア脅威だけでなく、幅広くマルウェア対策を実現するために、より汎用性の高いマルウェアファミリーを分類する手法の有効性を示す。ここでは、機械学習を用いて、マルウェアが攻撃で使った API と操作したフォルダパスから、マルウェアファミリーの動作特性と動作パターンを学習することで分類する。さらに、特徴量寄与度の分析手法を用いて各マルウェアファミリーがよく使用する API と動作目的を示す。この手法を用いることにより、マルウェアのファミリー、攻撃目標と感染経路などを特定し、より包括的なマルウェア対策が可能となる。

論文の構成は以下となる。第 1 章は、“序論”である。ここでは、ニューノーマル時代におけるサイバーセキュリティにある課題を記述してから、本研究の目的を述べ、本論文の構成および得られた成果を述べる。第 2 章は、“ニューノーマル時代のマルウェア脅威”であり、ニューノーマル時代に向け顕在化したマルウェア脅威の状況をまとめてから、本研究で提案するマルウェア対策を概説する。第 3 章は、“マルウェアが攻撃対象とした脆弱性の特定手法”である。この章では、テレワーク製品を狙ったマルウェアに悪用された脆弱性の特定という課題を解決するために、静的解析を用いて、自動的にマルウェアに悪用された脆弱性のエクスプロイトコードを抽出し、脆弱性を特定する手法を提案している。第 4 章は、“ランサムウェアの分類”である。この章では、ランサムウェア攻撃による被害の特定と低減のための分類という課題を解決するために、ランサムウェアに呼び出された API グループ間の相関性を特徴量にして、機械学習でランサムウェアの動作パターンを把握し、ランサムウェアのファミリー、目的と攻撃手段などを特定する手法を提案している。第 5 章は、“マルウェアの分類”である。ランサムウェアに限らず、マルウェア全般を対象としたファミリーの分類手法が必要であるという課題を解決するために、機械学習を用いて、マルウェアに使用された API グループ間の相関性とマルウェアに操作されたフォルダから動作パターンと動作特徴を習得してから、分類手法を提案している。第 6 章は“結論”であり、本論文の内容の総括と今後の課題である。

# 目次

あらまし

<b>第 1 章 序論</b>	<b>7</b>
1.1 研究の背景	7
1.2 本研究の目的	8
1.3 本論文の構成	9
<b>第 2 章 ニューノーマル時代におけるマルウェア脅威</b>	<b>12</b>
2.1 脆弱性を悪用したサイバー攻撃による被害の激増	12
2.2 二重脅迫型ランサム攻撃による被害の激増	13
2.3 マルウェア脅威への対策及び解決したい課題	15
<b>第 3 章 マルウェアが攻撃対象とした脆弱性の特定手法</b>	<b>17</b>
3.1 はじめに	17
3.2 脆弱性の特定手法	17
3.3 特定した脆弱性に基づく関係性調査	19
3.3.1 データセット	19
3.3.2 マルウェアファミリーの特徴と脆弱性間の関係性調査	21
3.4 考察	28
3.4.1 検体数の不均衡さが調査結果に与える影響について	28
3.4.2 2019 年以降の IoT マルウェアの特徴について	28
3.5 まとめ	28
<b>第 4 章 ランサムウェアに対する分類</b>	<b>30</b>
4.1 はじめに	30
4.2 提案システム	30
4.2.1 ランサムウェア分類に使用する特徴量の有用性の検証	31
4.2.2 提案システムの概要	33
4.2.3 特徴量の処理フロー	34
4.2.4 特徴量の変換とラベル付けの流れ	35
4.3 提案方式の評価実験	35
4.3.1 分類器の最適化設定	35
4.3.2 評価実験で使用するデータセット	36
4.4 評価実験の結果	37

4.4.1 分類の結果	37
4.4.2 分類結果に対する分析	37
4.5 考察	38
4.5.1 ファミリー分類に各 API の寄与度	39
4.5.2 既存手法の結果との比較	41
4.6 まとめ	42
<b>第 5 章 マルウェア攻撃に対する分類</b>	<b>44</b>
5.1 はじめに	44
5.2 提案方式	44
5.2.1 API グループ間の相関係数の算出	45
5.2.2 フォルダ操作頻度の抽出	45
5.3 評価実験	46
5.3.1 評価実験の前提条件	47
5.3.2 評価実験用データへの処理フロー	47
5.4 評価実験の結果	51
5.4.1 オーバーサンプリングなし	51
5.4.2 オーバーサンプリングあり	51
5.5 考察	53
5.5.1 オーバーサンプリングなしが分類結果に与える影響について	53
5.5.2 オーバーサンプリングありにおける特徴量の寄与度について	55
5.5.3 既存手法との比較	56
5.6 まとめ	57
<b>第 6 章 結論</b>	<b>59</b>
6.1 総括	59
6.2 今後の課題	60
<b>謝辞</b>	<b>61</b>
<b>致谢</b>	<b>62</b>
<b>発表論文リスト</b>	<b>64</b>
<b>参考文献</b>	<b>66</b>

## 図目次

図 1.1 本論文の構成.....	9
図 2.1 脆弱性の年間統計.....	13
図 2.2 Java ライブラリ「Apache Log4j」の脆弱性を標的とした .....	13
図 2.3 企業・団体等におけるランサムウェア被害の報告件数の推移.....	14
図 2.4 ランサムウェアによる被害件数、感染経路と攻撃手口の統計状況.....	14
図 3.1: Radare2 のコマンド”iz”で抽出した文字列.....	18
図 3.2 データセット統計(First Seen 情報).....	20
図 3.3 データセット統計(ラベル名).....	20
図 3.4 データセット統計(アーキテクチャ種類).....	20
図 3.5 Objdump 結果の例.....	22
図 3.6 例: Radare2 のコマンド”isj”で抽出したシンボル名 .....	22
図 3.7 無効な関数名の例.....	22
図 3.8 攻撃に使用している脆弱性の状況(共通脆弱性識別子と検体数) .....	24
図 3.9 攻撃に使用している脆弱性の状況(アーキテクチャごと).....	24
図 3.10 攻撃に使用している脆弱性の状況(マルウェアファミリーごと) .....	25
図 3.11 アーキテクチャ種類ごとによく使用されたオペコードの上位 50 個.....	26
図 3.12 アーキテクチャ種類ごとによく使用された関数名の上位 50 個.....	27
図 4.1 異なるファミリーの API グループ間における相関係数の比較.....	33
図 4.2 提案システムの概要 .....	34
図 4.3 ランサムウェアを分類するための処理の概要 .....	34
図 4.4 データセット.....	36
図 4.5 混同行列.....	38
図 4.6 特徴量の重要度 .....	41
図 5.1 API グループ相関性抽出アルゴリズム：頻度の抽出.....	48
図 5.2 API グループ相関性抽出アルゴリズム：相関係数の計算.....	49
図 5.3 フォルダ操作頻度抽出アルゴリズム：フォルダリスト抽出 .....	49
図 5.4 フォルダ操作頻度抽出アルゴリズムフォルダ：操作頻度の抽出 .....	50
図 5.5 検体に付与したラベル数の分布 .....	52
図 5.6 Precision-Recall 曲線 (オーバーサンプリングなし).....	52
図 5.7 Precision-Recall 曲線 (オーバーサンプリングあり).....	53
図 5.8 rp グループ間の相関係数の統計.....	54

## 表目次

表 1.1 各章における課題と成果 .....	11
表 3.1 アーキテクチャと <b>Objdump</b> との対応 .....	19
表 4.1 API グループとその詳細 .....	32
表 4.2 API グループ間の相関係数の略称 .....	32
表 4.3 実験結果 .....	37
表 4.4 既存研究との比較 .....	42
表 5.1 API グループ .....	46
表 5.2 各系列の分類に寄与度が高いフォルダと API グループ(一部) .....	56
表 5.3 既存手法との比較 .....	57



---

## 第1章 序論

---

本章では、本研究の背景と目的、本論文の構成を示す。

### 1.1 研究の背景

テレワークなどの新しい働き方が普及するにつれ、これまでのサイバーセキュリティ対策だけでは対応できず、見直す必要がでてきている。多くの企業が新型コロナウイルス感染対策として、オンライン会議などのテレワークを導入しているが、このような事業変化に乗じたサイバー攻撃が増加していることも一因にある。

今後のニューノーマル時代における情報システムには、情報セキュリティ対策の視点から次のような課題がある。

- ①. オフィス出勤を前提としないテレワークであり、多くの人が同時に VPN 経由で社内システムに接続する場合、処理速度が遅くなるなどの可用性につながる問題が生じる可能性がある；
- ②. テレワークの普及と共に、社内のいろいろな手続きはクラウドサービスの利用にシフトしている傾向があり、従来のサイバーセキュリティでは対応しきれない；
- ③. テレワークの場合、利用者のリモート端末そのものが、従来のセキュリティ対策での考え方であった「境界型セキュリティ」では防御できない。さらに、クラウドサービスを利用することが増えることで「境界型セキュリティ」では防御できる範囲が狭くなり、ゼロトラストセキュリティなどの新しいモデルに移行する必要がある。

IPA の「情報セキュリティ 10 大脅威 2021」によると、組織編の第 1 位が「ランサムウェアによる被害」、組織編の第 3 位が「テレワークなどのニューノーマルな働き方を狙った攻撃」となった[IPA2021]。この報告によれば、「ランサムウェアによる被害」では、ランサムウェアにより暗号化したデータを復旧するための金銭要求に加

え、暗号化する前にデータを窃取しておき、支払わなければデータを公開するぞと脅迫する「二重脅迫 (Double Extortion)」と呼ばれる攻撃も確認されている。また、「ニューノーマルな働き方を狙った攻撃」の発生要因としてテレワーク用ソフトウェアの脆弱性の悪用、急速なテレワーク移行による管理体制の不備、私物 PC や自宅ネットワークの利用を挙げている。

いずれもコロナ禍の前にも発生していたサイバー攻撃ではあり、多くの場合、マルウェアが関与あるいは起点としている。さらに、オンライン会議などのテレワーク環境の変化は、テレワーク用ソフトウェアの脆弱性対応問題を顕在化させ、二重脅迫型のサイバー攻撃は、ランサムウェアによる攻撃が金銭目当てとするサイバー攻撃としてのビジネス化を定着させたと言える。

ニューノーマル時代に備えた情報セキュリティ対策では、マルウェアが関与あるいは起点としたソフトウェアの脆弱性を悪用、ランサムウェアによる攻撃がサイバーセキュリティの課題であり、対策を講じる必要がある。

## 1.2 本研究の目的

本研究の目的は、ニューノーマル時代に備え、顕在化したマルウェア脅威の特定と被害の低減のために、マルウェアの分類手法を提案することにある。そこで、本研究では、顕在化したサイバーセキュリティの課題である、マルウェアが関与あるいは起点としたソフトウェアの脆弱性悪用、ランサムウェアによる攻撃に着目したマルウェアの分類手法を検討した後、ニューノーマル時代に備えたマルウェアの分類手法について、次の三つの研究を取りまとめる。

- テレワーク製品を狙ったマルウェアが攻撃に使用したソフトウェアの脆弱性の特定の研究
- ランサムウェアによる被害の低減のためのファミリー分類の研究
- マルウェアファミリーに限らず、汎用性のより高いマルウェアファミリーの分類の研究

「マルウェアが攻撃対象とした脆弱性の特定手法」では、静的解析を用いてマルウェアが攻撃で使った脆弱性を特定する。脆弱性の特定には、VirusShare[CFVS]という公開されたサイトから提供されているマルウェア検体データセットを通じて、脆弱性の特定手法の有効性を示す。その後、マルウェアファミリーとそれらに使用された脆弱性の関係性を示す。

「ランサムウェアによる被害の低減のためのファミリー分類の研究」では、ランサムウェアに呼び出された API から各ファミリーの動作パターンを機械学習で学習してから、複数のウェブサイトが提供しているマルウェア検体データセットを通じてファ

ミリーの特定手法の有効性を示す。その後、ランサムウェアが攻撃でよく使用する API を特定することで、各ランサムウェアファミリーがよく使用する API の傾向を示す。

最後に、「汎用性のより高いマルウェアファミリーの分類の研究」では、顕在化したマルウェア脅威だけでなく、全般的なマルウェア対策を実現するために、マルウェアの攻撃タイプに限らず、より汎用性の高いマルウェアファミリーの分類手法の有効性を示す。ここでは、機械学習を用いて、マルウェアが攻撃で使った API と操作したフォルダパスから、マルウェアファミリーの動作特性と動作パターンを学習することで分類を行う。その後、特徴量寄与度の分析手法を用いて各マルウェアファミリーのよく使用する API と動作目的を示す。

### 1.3 本論文の構成

本論文は、序論と結論を含め、図 1.1 に示す 6 章からなっている。

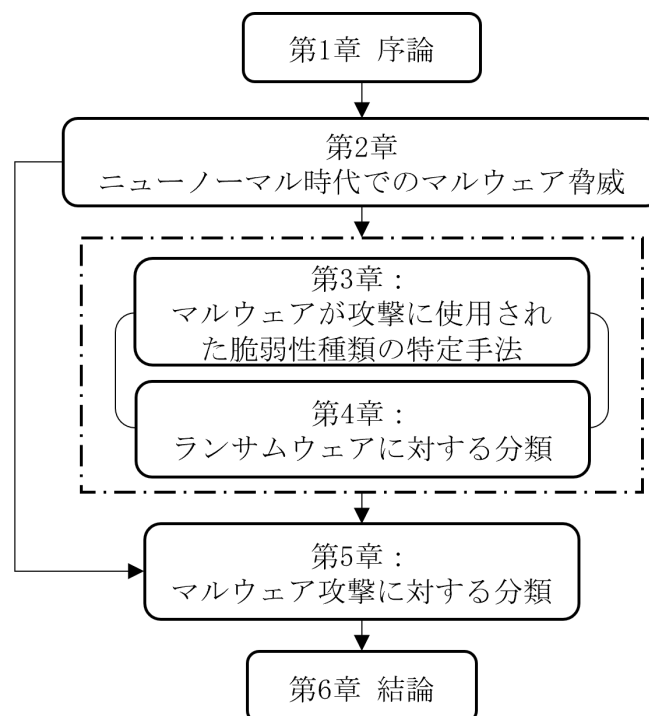


図 1.1 本論文の構成

第 2 章では、ニューノーマル時代に備え、顕在化したマルウェア脅威の状況をまとめてから、本研究で提案するマルウェア対策を概説する。

第 3 章から第 5 章では、各顕在化したマルウェア脅威に対する対策法について述べる。各章における研究課題と成果を表 1.1 に示す。第 3 章では、テレワーク製品を狙ったマルウェアが攻撃に使用したソフトウェアの脆弱性の特定という課題を解決する

ために、静的解析を用いて、自動的にマルウェアが攻撃した脆弱性のエクスプロイトコードを抽出し、脆弱性の共通脆弱性識別子の特定手法を提案している。第4章では、ランサムウェア攻撃による脅威の特定と被害の低減のための分類という課題を解決するために、ランサムウェアに呼び出されたAPIグループ間の相関性を特徴量にして、機械学習でランサムウェアの動作パターンを把握することから、ランサムウェアの系列、目的と攻撃手段などの特定手法を提案している。第5章では、マルウェアファミリーに限らず、より汎用性の高いマルウェアファミリー分類手法が必要であるという課題を解決するために、機械学習を用いて、マルウェアに使用されたAPIグループ間の相関性とマルウェアに操作されたフォルダから動作パターンと動作特徴を習得してから、分類を行う手法を提案している。

最後に、第6章は結論であり、本論文の内容の総括である。

表 1.1 各章における課題と成果

章	項目	説明
第 3 章	目的	テレワーク製品を狙ったマルウェアが攻撃に使用したソフトウェアの脆弱性を特定する.
	課題	ソフトウェアの脆弱性を攻撃するマルウェアによる被害の特定と軽減
	課題の解決法	静的解析を用いて、自動的にマルウェアが攻撃したソフトウェアの脆弱性のエクスプロイトコードを抽出し、脆弱性の共通脆弱性識別子の特定手法を提案する.
	成果	静的解析を用いて自動的にマルウェアが攻撃した脆弱性の特定が可能となる. また、いくつかのマルウェアファミリーがサイバー攻撃に使用した特定の脆弱性を明らかにした.
第 4 章	目的	ランサムウェアの動作パターンに基づき、ランサムウェアファミリーを分類する.
	課題	ランサムウェアによるサイバー攻撃被害の特定と軽減
	課題の解決法	機械学習でランサムウェアに呼び出された <b>Application Programming Interface (API)</b> グループ間の相関性からランサムウェアの動作パターンを把握し、ランサムウェアファミリー、目的と攻撃手段などの特定手法を提案する.
	成果	機械学習モデルの一つである <b>Support Vector Machine(SVM)</b> を使用して分類精度 98.2% を達成した. また、特徴量寄与度の分析から、各ランサムウェアファミリーが攻撃に使用する重要な API の特定を可能とする.
第 5 章	目的	ニューノーマル時代に備え、顕在化したマルウェア脅威は、脆弱性への攻撃とランサムウェアに限らないことから、より包括的な手法を用いてマルウェアファミリー分類する.
	課題	より包括的なマルウェアファミリーの分類手法
	課題の解決法	マルウェアの動作パターンとして <b>API</b> グループ間の相関性、特徴ある動作としてフォルダ操作頻度を使用し、機械学習による分類を行い、マルウェアの攻撃目的と手段などを分析する手法を提案する.
	成果	オーバーサンプリングありの場合で正確度は 99% に達成し、既存手法より優れたことを確認した. また、分類に寄与度が高い特徴量の分析により、各ファミリーが攻撃対象とするフォルダと操作タイプ、および攻撃に重要な <b>API</b> グループの特定を可能とする.

---

## 第2章 ニューノーマル時代におけるマルウェア脅威

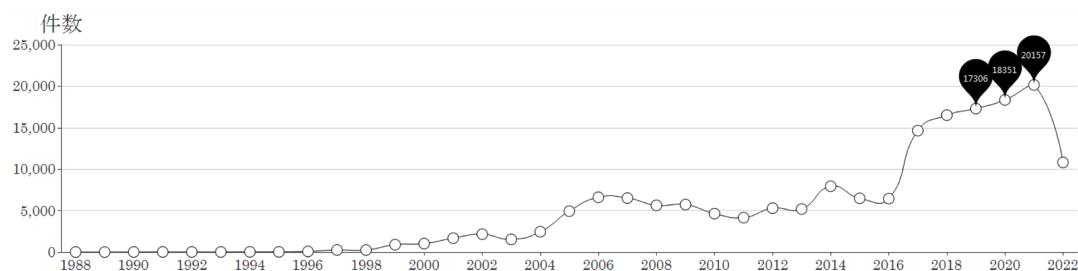
---

2018年12月、経済産業省は「2025年の崖」を解決するために「デジタルトランスフォーメーションを推進するためのガイドライン(DX推進ガイドライン) Ver.1.0」を発表した[DXGUIDE]. さらに、2020年、新型コロナウイルス感染の拡大は、組織・企業での対面の業務などを止め、既存のビジネスモデルではこの状況に対応できないことを示し、ビジネスモデルの改革が求められるようになった。改革を促せるDXの推進がコロナ禍において急速的に進められていったが、それに伴い、サイバーセキュリティにおいて、次の二つの問題が顕在化した。①ニューノーマル時代に突入するにあたり、テレワークで必要なツール、具体的にはVPN、ルータ、会議用ソフトウェアなどの利用ニーズの激増に伴い、これら設備の脆弱性を突いたマルウェア攻撃による被害が増加している。②従来からあった暗号化したデータを復旧するために金銭要求を要求するサイバー攻撃に加え、暗号化する前にデータを窃取しておき、支払わなければデータを公開するぞと脅迫する二重脅迫型のサイバー攻撃は、ランサムウェアによる攻撃が金銭目当てとするサイバー攻撃としてのビジネス化を定着させた。

### 2.1 脆弱性を悪用したサイバー攻撃による被害の激増

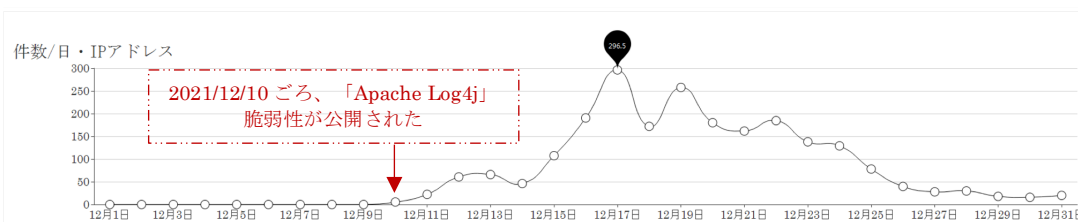
コロナ禍に突入し、テレワーク製品ニーズの増加に伴い、これら製品に狙った攻撃も急増している。特に、テレワーク製品の脆弱性を悪用した攻撃の増加は顕著である。

米国脆弱性データベース National Vulnerability Database の年間統計結果によると、脆弱性の数は過去10年間で3倍以上に増加していると(図 2.1)、2019年～2021年の間に、脆弱性の増加率が大きい[NVDS]. また、警察庁の報告によると、Java ライブラリ「Apache Log4j」の脆弱性が公表された以降、同脆弱性を標的としたアクセスの急増が観測されたとしている(図 2.2)[NPACYBER].



出典：National Vulnerability Database - Statistics [NVDs]

図 2.1 脆弱性の年間統計



出典：サイバー空間をめぐる脅威の情勢等－警察庁 [NPACYBER]

図 2.2 Java ライブラリ「Apache Log4j」の脆弱性を標的とした  
アクセス数の推移（宛先ポート別）

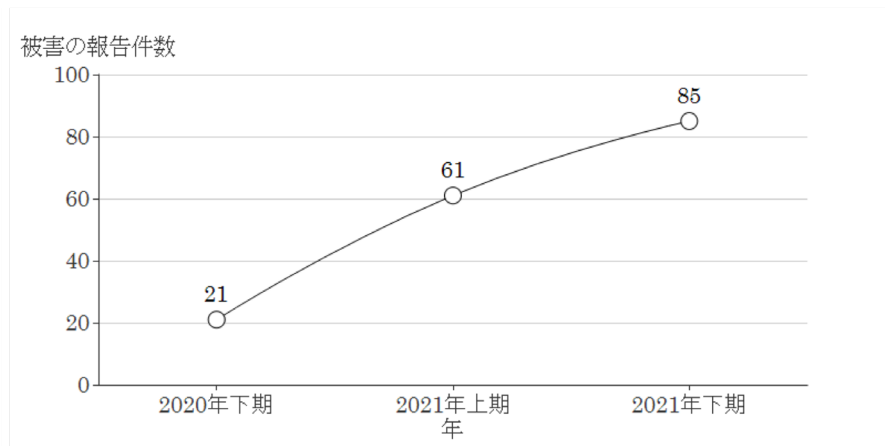
## 2.2 二重脅迫型ランサム攻撃による被害の激増

二重脅迫型のランサム攻撃は、対象とする PC やサーバー上のデータを暗号化することで身代金を要求する従来型のランサムウェアと異なり、データの暗号化だけでなく、内部情報の窃取を組み合わせた攻撃で攻撃活動である。二重脅迫は、暗号化したデータを復元するために金銭を要求する一つ目の脅迫と、金銭の支払いがない場合は窃取した内部情報を Web サイト上で公表する、公表されたくなければ金銭の支払いに応じろという二つ目の脅迫からなる[LAC]。コロナ禍の前には、従来型ランサムウェアによる被害が多かったが、テレワークの常態化と共に、二重脅迫型ランサム攻撃が増加している。

警察庁の令和 3 年度のサイバー空間をめぐる脅威の情勢の調査によると、日本国内では、令和 2 年度下半期から令和 3 年度の下半期までに、警察庁に報告のあったランサムウェア被害の合計件数は 146 件であり、右肩上がり増加している(図 2.3)[NPACYBER]。被害企業・団体などの規模別報告件数の調査結果によると、中小企業の被害件数が最も多い。感染経路の調査結果(有効回答は 76 件)によると、過半数の攻撃は VPN 機器からの侵入となっている。さらに、金銭の要求手口が確認できた被害事例(100 件)のうち、85 件(85.0%)の被害の手口は二重脅迫である(図 2.4) [NPACYBER]。

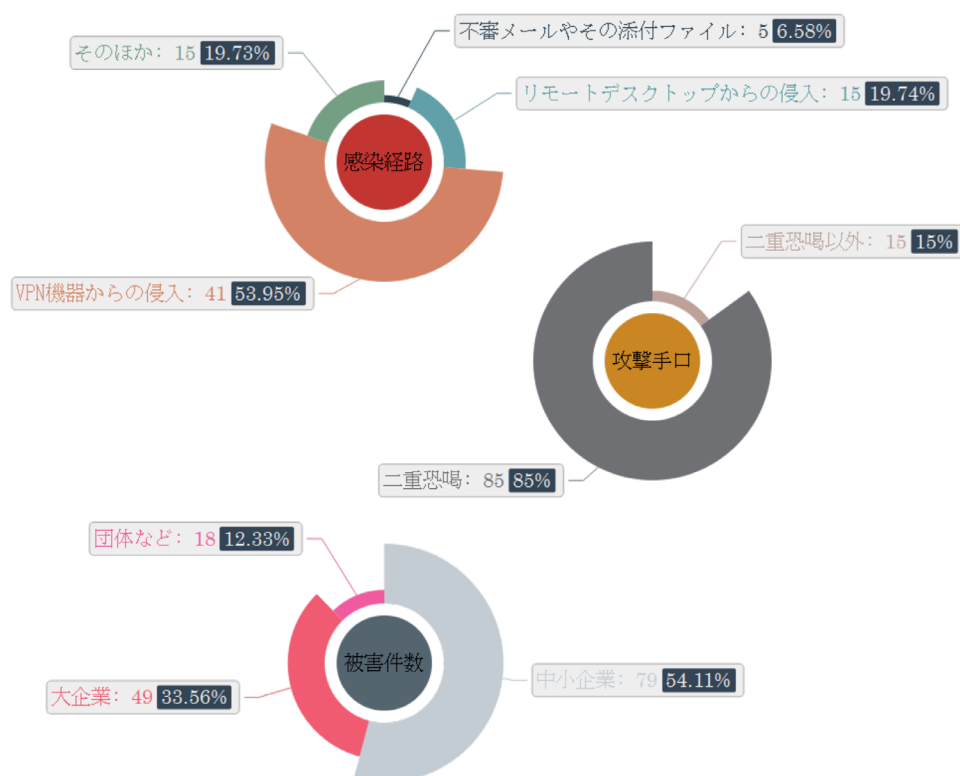
顕在化した問題による脅威の特定と被害の低減のためには、顕在化した問題に絞っ

た対策法を検討するだけではなく、マルウェアによる脅威全般への対策も検討していく必要がある。



出典：サイバー空間をめぐる脅威の情勢等－警察庁 [NPACYBER]

図 2.3 企業・団体等におけるランサムウェア被害の報告件数の推移



出典：サイバー空間をめぐる脅威の情勢等－警察庁 [NPACYBER]

図 2.4 ランサムウェアによる被害件数、感染経路と攻撃手口の統計状況



## 2.3 マルウェア脅威への対策及び解決したい課題

本研究の目的は、ニューノーマル時代に備え、顕在化したマルウェア脅威の特定と被害の低減のために、マルウェアの分類手法を提案することである。その目的のためには、本論文では、次に示す3つのマルウェア対策関連手法および分類手法に関する研究を取りまとめた。

### (1) テレワーク製品を狙ったマルウェアが攻撃に使用した脆弱性の特定に関する課題

テレワーク製品の脆弱性を悪用したサイバー攻撃の増加は顕著であり、脆弱性を突いた脅威の特定と被害の低減のためには、マルウェアが攻撃に使用したソフトウェアの脆弱性の特定が必要不可欠である。

一つ目の手法として、マルウェアが攻撃に使用した脆弱性(以下、共通脆弱性識別子と呼ぶ)の特定手法である。テレワーク製品などの脆弱性を悪用したサイバー攻撃の被害を低減するためには、どの脆弱性が悪用されたのかを特定することで、その脆弱性の修正に役に立つだけでなく、マルウェアの攻撃手口を明らかにでき、被害をも抑制できる。第3章では、マルウェア検体のバイナリファイルに保存されている文字列を抽出し、公開されている脆弱性データベースと照合して共通脆弱性識別子を特定するという脆弱性の特定手法を提案する。また、脆弱性の共通脆弱性識別子を特定するにあたり、表層解析と静的解析手法を用いて、マルウェアが攻撃に使用した脆弱性の共通脆弱性識別子とアーキテクチャ種類やマルウェアファミリーの関係などを調査した結果を示す。この手法を用いることにより、マルウェアが攻撃対象とした脆弱性を特定し、さらに攻撃手法と攻撃目的を把握することが可能となる。

### (2) ランサムウェアの分類に関する課題

従来のランサムウェアはユーザのPCを感染し、データを暗号化したうえで、身代金を要求するタイプであるが、ニューノーマル時代に向け激増しているランサムウェアは従来のものと異なり、窃取したデータの公開を止める代わりに身代金の支払いを要求する二重脅迫型である。二重脅迫型の場合、ランサムウェアが関与する場合と、データを暗号化して身代金を要求するというランサム技術が使われる場合もあることから、以降、二重脅迫型ランサム攻撃とする。従来のランサムウェアに感染した場合は、データを復元できれば問題は解決するが、二重脅迫型ランサム攻撃では、身代金を振り込まなければ、窃取したデータを公開し、被害に遭ったことを第三者に知られてしまうため、企業にとってはよりインパクトが高い。したがって、ランサムウェア脅威の特定と被害の低減のためのアプローチが必要不可欠である。

二つ目の手法として、ランサムウェアによる脅威の特定と被害の低減のための分類手法である。ニューノーマル時代を迎えるにあたり二重脅迫型ランサム攻撃による被害を低減するためには、ランサムウェアファミリーを分類することで、そのランサムウェアの系列、機能、目標と感染経路などを明らかにする必要がある。第4章では、

ランサムウェアのファイルのアクセス、名前の変更、削除、暗号化などの動作に使用した API グループ間の相関性に着目し、ランサムウェアファミリーの分類手法を提案する。また、提案手法である特徴量の寄与度分析手法を用いて、ランサムウェアファミリーの分類に重要な API を特定し、そのファミリーのランサムウェアが攻撃によく行う動作タイプを明らかにする。

### (3) マルウェア全般を対象とした分類に関する課題

ニューノーマル時代を迎えるにあたり、オンライン会議などのテレワークを狙ったマルウェアの種類が急増しているとともに、その被害も飛躍的に増加している。そのため、これらのマルウェアによる被害を軽減するためには、対策を立てる前に、ニューノーマル時代に向けた汎用性のより高いマルウェア全般を対象とした分類手法が必要不可欠である。

三つ目の手法として、特定のマルウェア種類に対する分類ではなく、すべてのマルウェア種類に対して、汎用性のより高く、全体的な分類手法である。ニューノーマル時代において、危険度が高い脅威はテレワーク製品の脆弱性を悪用したマルウェア脅威や二重脅迫型ランサム攻撃の脅威だけでなく、他のいろいろな種類のマルウェア脅威にも対処していく必要がある。さまざまなマルウェア脅威から情報資源を守るためには、マルウェア脅威にあわせた対策を作成することであり、特定のマルウェアファミリーに限らず、より汎用性が高いマルウェアファミリー分類手法はマルウェア脅威にあわせた対策を適用するにあたり効率的な手法である。第 5 章では、マルウェアが使用した API グループと操作したフォルダに着目し、API グループ間の相関性とフォルダ操作頻度を特徴量として使用し、機械学習で分類する手法を提案する。また、提案手法を用いて、マルウェアファミリーが攻撃によく使用する API グループの種類と動作を明らかにする。

---

## 第3章 マルウェアが攻撃対象とした脆弱性の特定手法

---

本章では、ソフトウェアの脆弱性を攻撃するマルウェアによる被害の特定と軽減という1つ目の課題を解決するために、テレワーク製品などIoTを狙ったマルウェア(以下、IoT マルウェア)が攻撃に使用した脆弱性の特定手法を提案する。また、提案手法により特定した脆弱性の共通脆弱性識別子とアーキテクチャ種類やマルウェアファミリーの関係、アーキテクチャ種類ごとにオペコードと関数名の使用状況の調査結果を示す。

### 3.1 はじめに

本章では、テレワーク製品などを狙ったIoT マルウェアを対象にして、静的解析を用いて、自動的にマルウェアが攻撃に使用した脆弱性の攻撃コードであるエクスプロイトコードを抽出し、脆弱性の共通脆弱性識別子の特定手法を提案する。また、提案手法を用いて、マルウェアが攻撃に使用した脆弱性の共通脆弱性識別子とアーキテクチャ種類やマルウェアファミリーの関係性を調査(以下、共通脆弱性識別子との関係性調査)すると共に、IoT マルウェアが攻撃に使用した脆弱性をアーキテクチャ種類ごとにオペコードと関数名の使用状況の視点から調査した結果(以下、実装との関係性調査)を示す。

本章の構成は次のとおりである。3.2 節で、マルウェアが攻撃に使用した脆弱性の特定手法を述べる。3.3.1 節では、脆弱性の特定手法の調査で使用するデータセットの詳細を示す。3.3.2.3 節では、調査内容と結果を示す。3.4 では、調査内容に対する考察、調査内容における不足点について述べる。3.5 節はまとめである。

### 3.2 脆弱性の特定手法

本節では、マルウェアが攻撃に使用した脆弱性の特定手法を述べる。

マルウェアが攻撃に使用する脆弱性を特定するために、各検体のバイナリファイ

ルに保存されている文字列を抽出し、公開されている脆弱性データベースと照合する。具体的な手法は：

- (1) Radare2 のコマンド”iz”で検体から文字列を抽出した後、文字列に含まれる攻撃に使用するリクエスト情報を抽出する。
- (2) リクエスト情報と、Exploit DataBase[EPITBASE]および National Vulnerability Database(NVD)[NVD2013]などの脆弱性データベースとを照合して攻撃に使用する脆弱性の共通脆弱性識別子である CVE 番号または脆弱性に付与された脆弱性識別子を特定する。

例えば、Radare2 のコマンド”iz”を実行すると、図 3.1 のような結果が表示される。エクスプロイトコードであるリクエスト情報(赤文字)を脆弱性データベースで検索すると、攻撃に使用する脆弱性は CVE-2017-17215 であると特定できる。このように、検体ごとに攻撃に使用する脆弱性を特定していく。

```
[0x00008190]> iz
[Strings]
nth paddr vaddr len size section type string
0 0x00014ebc 0x0001cebc 806 807 .rodata ascii POST /ctrlt/DeviceUpgrade_1 H
TTP/1.1\r\nContent-Length: 430\r\nConnection: keep-alive\r\nAccept: /**\r
\nAuthori
zation: Digest username="dslf-config", realm="HuaweiHomeGateway", nonce="
88645cefb1f9edee336e3569d75ee30", uri="/ctrlt/DeviceUpgrade_1", response="3
612f843a42db38
f48f59d2a3597e19c", algorithm="MD5", qop="auth", nc=00000001, cnonce="248
d1a2560100669"\r\n\r\n<?xml version="1.0" ?><s: Envelope xmlns:s="http://s
chemas.xmlsoap.or
g/soap/envelope/" S:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
><s: Body><u: Upgrade xmlns:u="urn:schemas-upnp-org: service:WANPPPCon
nection:1"><NewStat
USURL>$C/bin/busybox wget -3 164.68.116.122 -1 /tmp/kh -r /mips; /bin/busybo
x chmod 777 * /tmp/kh; /tmp/kh huawei</NewStatusURL><NewDownloadURL
>$(echo HUAWEIUPNP)
</NewDownloadURL></u:Upgrade></s:Body></s: Envelope>\r\n\r\n
```

図 3.1: Radare2 のコマンド”iz”で抽出した文字列

ただし、すべての検体からエクスプロイトコードであるリクエスト情報を抽出できるわけではなく、52,617 検体中 10,213 検体からエクスプロイトコードを抽出できた。

表 3.1 アーキテクチャと Objdump との対応

アーキテクチャ	Objdump 種類
ARM	objdump
I386	objdump
X86	objdump
AMD	objdump
MIPS	mips-linux-gnu-objdump
Superh	sh4- linux -gnu-objdump
Powerpc	powerpc- linux -gnu-objdump
Motorola	m68k- linux -gnu-objdump
Sparc	sparc64- linux -gnu-objdump
Aarch64	aarch64- linux -gnu-objdump

### 3.3 特定した脆弱性に基づく関係性調査

本節では、マルウェアファミリーの特徴と攻撃に利用した脆弱性間の関係を明らかにするために、脆弱性特定手法に基づいた調査で使用するデータセット(3.3.1 節)とマルウェアファミリーの特徴と攻撃に利用した脆弱性間の関係性調査の内容(3.3.2 節)について述べる。

#### 3.3.1 データセット

本調査で使用する検体は、ファイルタイプ ELF 形式のマルウェアを IoT マルウェアと想定し、公開されているマルウェアレジポトリ Virusshare[CFVS]から入手した。データセットの総数は 52,617 検体である。Virustotal から入手した各検体の First Seen 情報(図 3.2)をみると、IoT マルウェア数は 2016 年以降、年々増加していることが分かる。

また、AVClass[SRKC2016]を使用して抽出した各検体のラベル(図 3.3)から、データセットに含まれる検体の有効なラベル名は Mirai, Gafgyt と Tsunami である。SINGLETON は AVClass で有効なラベルを抽出できなかったときの名称で、検体数 900 個以下のラベルを Others とした。静的解析ツール Radare2[RADARE2]を使用して抽出した検体の実行可能なアーキテクチャ情報については、検体数が最も多いアーキテクチャ種類は ARM であり、X86 と Aarch64 の検体数は比較的少なかった(図 3.4)。

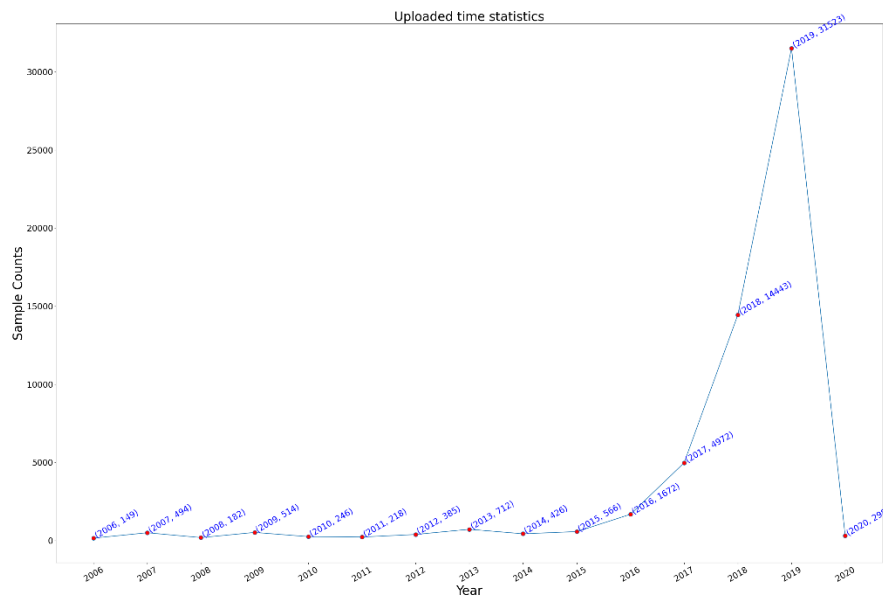


図 3.2 データセット統計(First Seen 情報)

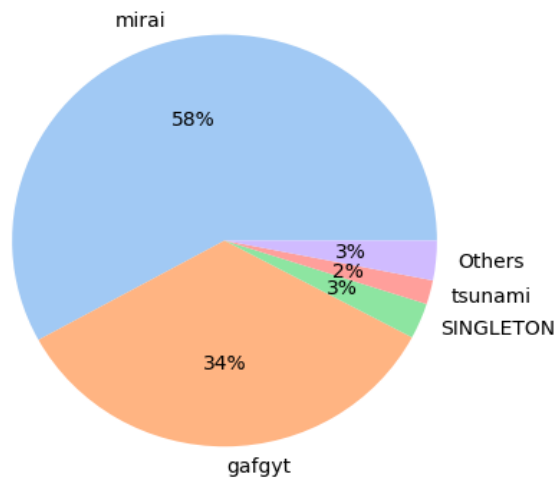


図 3.3 データセット統計(ラベル名)

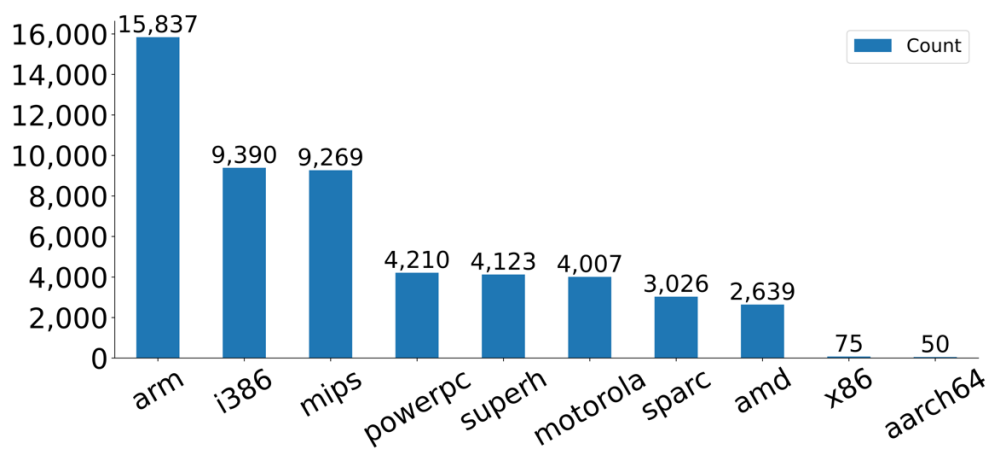


図 3.4 データセット統計(アーキテクチャ種類)

### 3.3.2 マルウェアファミリーの特徴と脆弱性間の関係性調査

本節では、脆弱性の特定手法のもとに、表層解析と静的解析手法を使用して、マルウェアファミリーの特徴と攻撃に利用した脆弱性間の関係を明らかにするための調査手法(3.3.2.1 節)と、情報抽出手法(3.3.2.2 節)、関係性調査の結果(3.3.2.3 節)を記述する。関係性調査とは共通脆弱性識別子との関係性調査と、実装との関係性調査であり、その調査内容は以下のとおりである：

- 共通脆弱性識別子との関係性調査
  - 攻撃対象とした脆弱性と検体数
  - アーキテクチャ毎の攻撃対象とした脆弱性
  - マルウェアファミリー毎の攻撃対象とした脆弱性
- 実装との関係性調査
  - アーキテクチャ毎に、使用回数が最も多いオペコード、関数名の調査

#### 3.3.2.1 調査手法

具体的な調査手法は、次の通りである。

- (1) 表層解析では、各検体から実行可能なアーキテクチャ情報を抽出する；
- (2) 静的解析で、検体ごとに、使用しているオペコードおよび関数名を抽出する；
- (3) 上述までの調査に加えて、ツール AVClass で検体のファミリー名を抽出し、アーキテクチャ種類ごとに、マルウェアファミリーごとに、Executable and Linkable Format (ELF)フォーマットごとに、攻撃に使用する脆弱性の実装状況と傾向、およびオペコードと関数名の使用傾向を調査する。

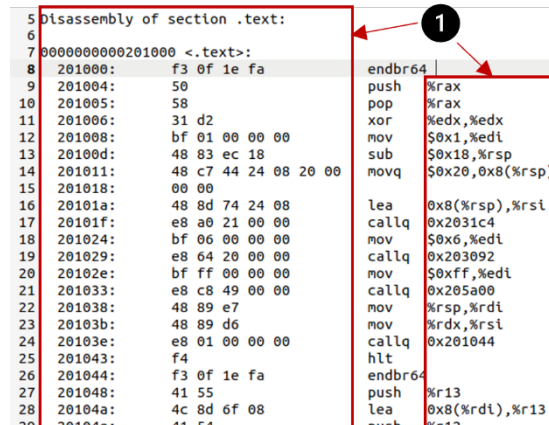
#### 3.3.2.2 情報の抽出手法

マルウェア検体からの情報抽出の手順について述べる。

##### (1) オペコード抽出

オペコード抽出においては、データセットの検体数が5万個を超えていることから、Radare2よりも効率の良いツール Objdump[GNUOBJ]を使用した。また、アーキテクチャ種類が異なると、対応する Objdump も異なるため、表 3.1 にあるアーキテクチャ種類に対応した Objdump を使用して抽出した。

抽出においては、Objdump コマンドのパラメータ“-d”をつけて検体で使用している関数のオペコードを抽出した。Objdump の結果は図 3.5 の示すように、特定のフォーマットで保存しているため、無関係な情報をフィルタリングする必要がある。無関係な情報とは無効なオペコードまたは機械語コード、アドレスなど(図 3.5 の①)であり、例えば“ud0”，”ud1”などが”Undefined Instruction”のことを指すため、これらは無効なオペコードとみなし、除外した。



Address	Disassembly	Comment
201000	f3 0f 1e fa	endbr64
201004	50	push %rax
201005	58	pop %rax
201006	31 d2	xor %edx,%edx
201008	bf 01 00 00 00	mov \$0x1,%edi
20100d	48 83 ec 18	sub \$0x18,%rsp
201011	48 c7 44 24 08 20 00	movq \$0x20,0x8(%rsp)
201018	00 00	
20101a	48 8d 74 24 08	lea 0x8(%rsp),%rsi
20101f	e8 a0 21 00 00	callq 0x2031c4
201024	bf 06 00 00 00	mov \$0x6,%edi
201029	e8 64 20 00 00	callq 0x203092
20102e	bf ff 00 00 00	mov \$0xff,%edi
201033	e8 c8 49 00 00	callq 0x205a00
201038	48 89 e7	mov %rsp,%rdi
20103b	48 89 d6	mov %rdx,%rsi
20103e	e8 01 00 00 00	callq 0x201044
201043	f4	hlt
201044	f3 0f 1e fa	endbr64
201048	41 55	push %r13
20104a	4c 8d 6f 08	lea 0x8(%rdi),%r13
20104c	41 54	push %r12

図 3.5 Objdump 結果の例

## (2) 関数名抽出

検体中使用している関数名の抽出は Radare2 のコマンド”isj”で検体のシンボル情報を抽出し、その中にタイプが”FUNC”であるシンボル名を関数名として抽出した(図 3.6).

```
vaddr=0x00000040 paddr=0x00000040 ord=011 fwd=NONE sz=11 bind=GLOB
AL type=FUNC name=int_cmp
```

図 3.6 例: Radare2 のコマンド”isj”で抽出したシンボル名

抽出した関数名にもノイズがあり、例えば”

pB7B07565E191EBA8F1CE74DD846F3788”, ”\_L\_lock\_18”, ”\_L\_unlock\_60”  
のような関数名は無効な関数名とみなし、除外した(図 3.7).

```
_Unwind_Resume
__cxa_call_unexpected
__gxx_personality_v0
p4E316BDE3CEC6A0F6B7E28BB06AEDBE6
pA4EBBE49B0D9908BCD646D27F772EE29
p04F1EDB1E9A348D09099142544211FE1
_L_lock_18
_L_unlock_60
```

図 3.7 無効な関数名の例

### 3.3.2.3 関係性調査の結果

本節では、脆弱性の特定手法に基づいた関係性調査の経由で明らかにしたマルウェア



アフファミリーの特徴と攻撃に利用した脆弱性間の関係を示す。

### 3.3.2.3.1 共通脆弱性識別子との関係性調査の結果

#### (1) 攻撃対象とした脆弱性と検体数

- 10,213 検体から合計 46 個の攻撃に使用する脆弱性を特定した。特定した脆弱性および検体数を図 3.8 に示す。横軸は脆弱性の名前で、縦軸は攻撃に使用する脆弱性を実装していた検体数である。
- 上位三位はすべてルータを対象とした脆弱性である。
- 検体数が最も多い脆弱性は CVE-2017-17125(Huawei HG532 における入力確認に関する脆弱性[CVE17215])であり、9,060 検体に実装されていた。二番目は CVE-2014-8361(Realtek SDK の miniigd SOAP サービスに存在する任意のコード実行を許してしまう脆弱性[CVE8361])であり、2,358 検体で実装され、三番目は Linksys E-series の Remote Code Execution[JULERCE] (CVE 番号の付与なし)で、2,225 検体で実装されていた。

#### (2) アーキテクチャ毎の攻撃対象とした脆弱性

実行可能なアーキテクチャごとの結果を図 3.9 に示す。横軸はアーキテクチャの種類であり、縦軸は脆弱性の名称である。グラフにあるサークルのサイズが大きいほど、該当するアーキテクチャベースのシステムを対象とした攻撃に使用する脆弱性の検体数が多いことを意味する。10,213 検体から 46 個の攻撃に使用する脆弱性を特定したが、これらの検体の中に、アーキテクチャの種類数は 9 個しかなく、X86 向けの検体から脆弱性を抽出できなかった。

#### (3) マルウェアファミリー毎の攻撃対象とした脆弱性

- 結果を図 3.10 に示す。横軸はファミリー名であり、縦軸は脆弱性の名称である。グラフにあるサークルのサイズが大きいほど、該当するマルウェアファミリーが攻撃に使用する脆弱性の検体数が多いことを意味する。
- マルウェアファミリー Mirai は、脆弱性 CVE-2017-17215, Linksys E-series: Remote Code Execution, CVE-2014-8361 を攻撃に使用しており、ファミリー Gafgyt と Tsunami は脆弱性 CVE-2017-17215 を攻撃に使用している。SINGLETON という名前は、VirusTotal の結果から有効なファミリー名を抽出できなかったことを表し、これらのファミリー名が不明の検体は脆弱性 Linksys E-series: Remote Code Execution をよく利用している

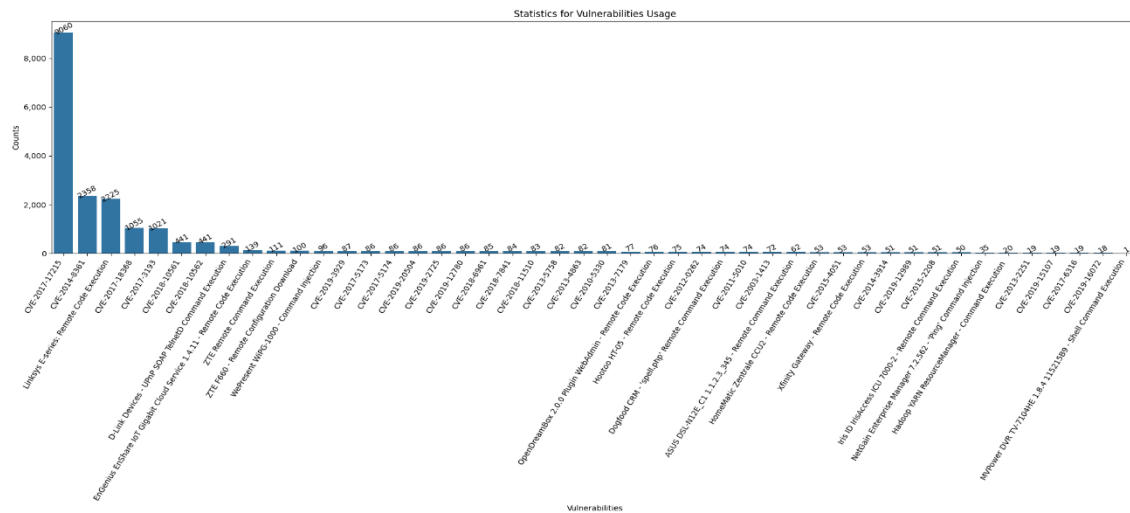


図 3.8 攻撃に使用している脆弱性の状況(共通脆弱性識別子と検体数)

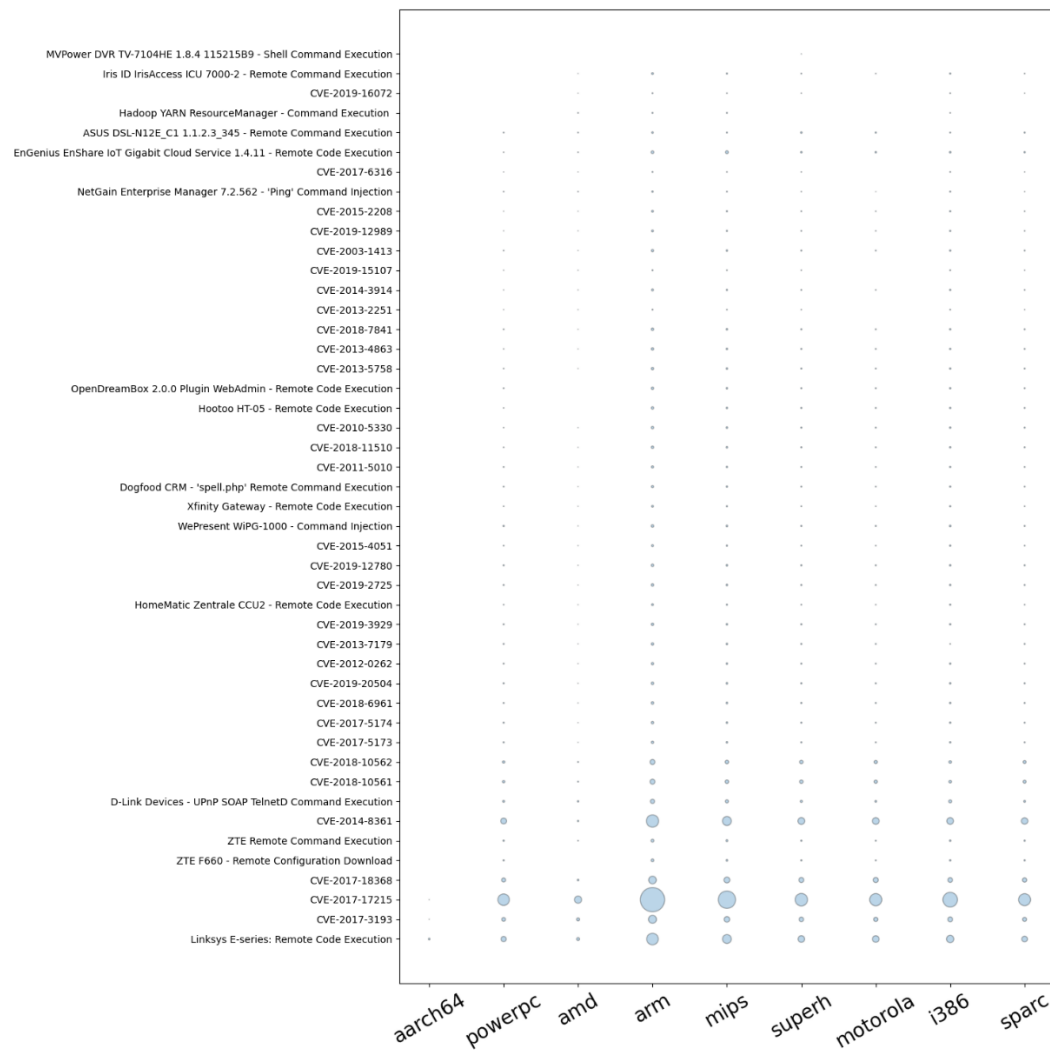


図 3.9 攻撃に使用している脆弱性の状況(アーキテクチャごと)

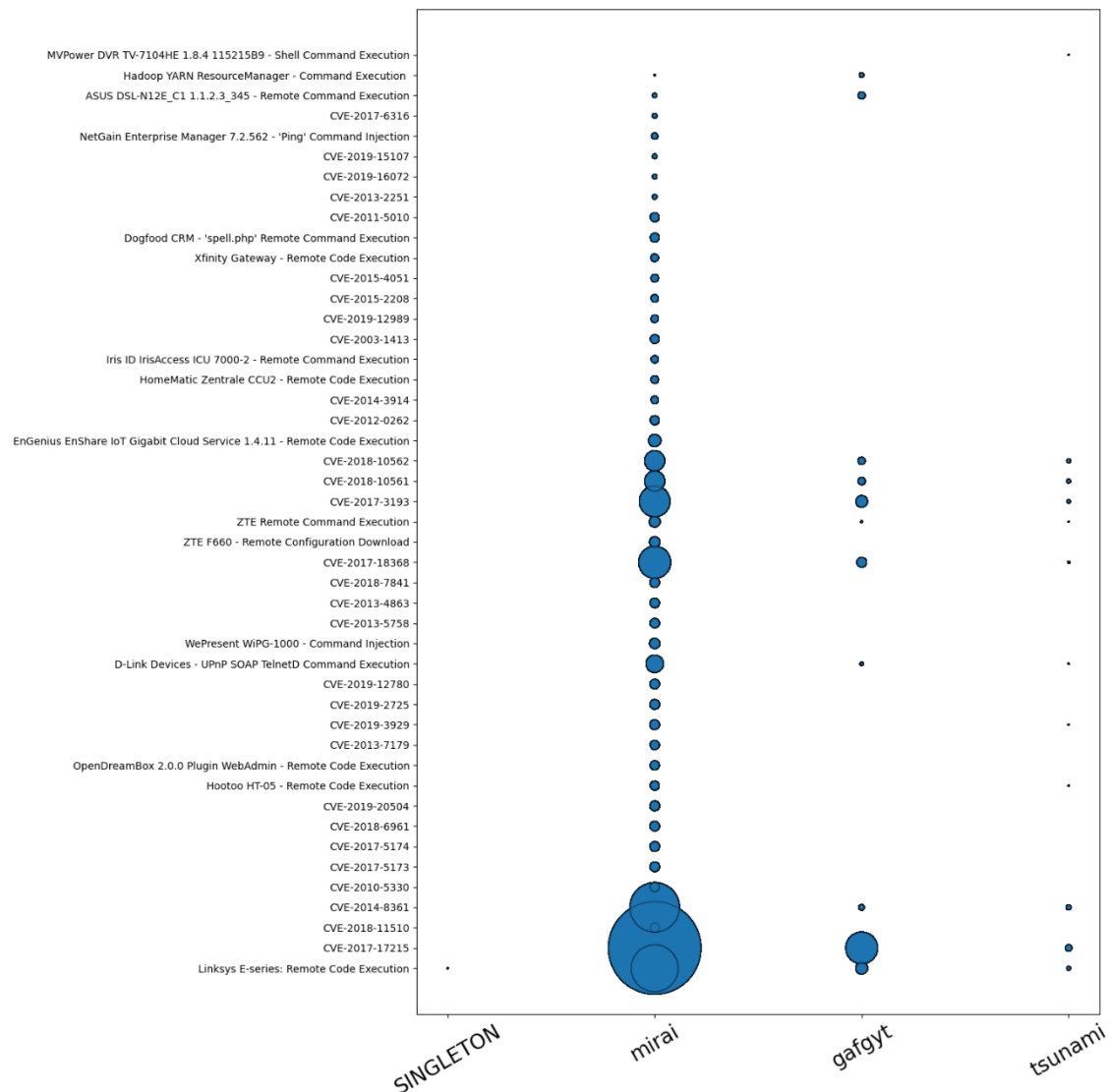


図 3.10 攻撃に使用している脆弱性の状況(マルウェアファミリーごと)

### 3.3.2.3.2 実装との関係性調査の結果

#### (1) アーキテクチャ毎に、使用回数が最も多いオペコード

データセットで特定できたアーキテクチャ種類は計 10 個であり、アーキテクチャ種類ごとに使用回数の高いオペコードと関数名の上位 50 個を図 3.11 と図 3.12 に示す。横軸は関数名であり、縦軸は関数名の使用回数である。

- 図 3.11 の示すように、Aarch64, ARM と Superh 以外のアーキテクチャの場合には各オペコードの使用回数の差が非常に大きい。また、アーキテクチャ ARM の検体数が最も多いが、上位 50 個のオペコードの使用回数はほぼ同じで 1 万回をわずかに超えている。要因としては、各検体で使用しているオペコードの種類がそれぞれ異なることに起因すると推定できる。

- 検体の実行可能なアーキテクチャ I386, Motorola と AMD において, 使用回数が最も多いオペコードは 80 万回を超えていた. 要因としては, これらのアーキテクチャを対象とする検体の場合, 使用するオペコードが特定の種類が集中しているか, マルウェアの動作が類似していたと推定できる.
- アーキテクチャ Superh の検体数は 4 千個を超えたが, 抽出できたオペコードが”nop”と”add”の二種類オペコードに留まった. 要因としてはオペコードの抽出ができなかったことによる.

Top 50 Frequently Used Opcodes for Each Architecture

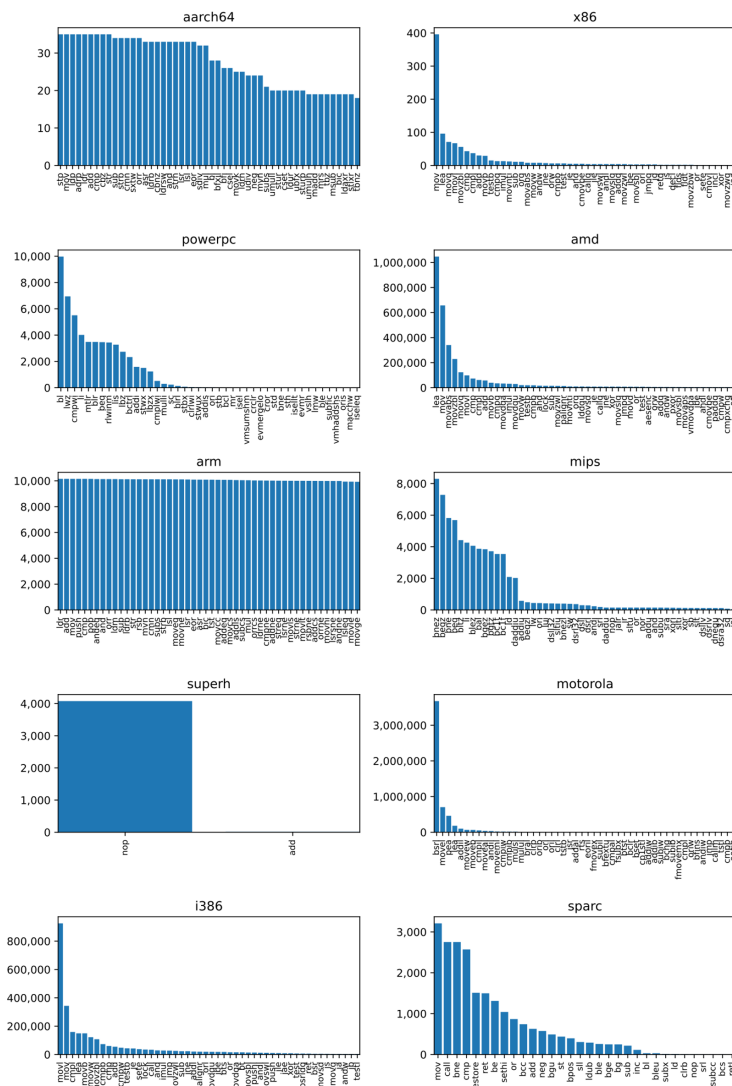


図 3.11 アーキテクチャ種類ごとによく使用されたオペコードの上位 50 個

(2) アーキテクチャ種類ごとに, 使用回数が最も多い関数名

- 図 3.12 の示すように, アーキテクチャ Motorola と I386 の場合, 使用回数が最も多い関数名と残りの関数名との差が大きい, 他のアーキテクチャの場合,



### 3.4 考察

本節ではマルウェアファミリーの特徴と攻撃に利用した脆弱性間の関係を明らかにした際の調査結果に対して掘り下げることで、以下のことを確認した：

- (1) (3.4.1 節)不均衡データセットは調査結果をある程度に影響したが、マルウェアの動作傾向の調査には差し支えない
- (2) (3.4.2 節) 2019 年以降に登録された IoT マルウェアは、過去に使用されていない脆弱性を使用し、2018 年までに登録されたものよりある程度進化している

#### 3.4.1 検体数の不均衡さが調査結果に与える影響について

検体が使用しているオペコード、関数名の回数および攻撃に使用する脆弱性の共通脆弱性識別子と検体数を対象として IoT マルウェアの基礎情報を調査した。図 3.4 にあるアーキテクチャごとの検体数で示す通り、各アーキテクチャの検体数が不均衡であるため、調査結果にある程度の影響を与えていると考える。しかし、本章は IoT マルウェアの基礎情報および動作傾向の調査を目的とするため、全体的には不均衡データセットでも動作傾向を把握できると考える。

#### 3.4.2 2019 年以降の IoT マルウェアの特徴について

攻撃に使用する脆弱性の実装状況から 2018 年までに登録された IoT マルウェア(以下、2018 年以前検体と呼ぶ)と 2019 年以降に登録されたもの(以下、2019 年以後検体と呼ぶ)とを比較調査した。比較調査から、10,213 検体(2018 以前は 3,451 検体で、2019 年以後は 6,762 検体である)から特定した 46 個の攻撃に使用する脆弱性中、11 個の脆弱性は 2018 年以前検体と 2019 年以後検体の両方とも使用され、残りの 35 個の脆弱性は 2019 年以後検体でのみ使用している。これらの 35 個の脆弱性には 2019 年に公開されたものがあり、最も古い脆弱性は 2013 年に公開されたものもある。

この状況から、2019 年以後検体は、過去に使用されていない脆弱性を使用し、2018 年以前検体よりある程度進化していると言える。

### 3.5 まとめ

本章ではソフトウェアの脆弱性を攻撃するマルウェアによる被害の特定と軽減という 1 つ目の課題を解決するために、IoT マルウェアが攻撃に使用した脆弱性の特定手法を提案した。

提案手法は、静的解析手法を用いて各 IoT マルウェア検体のバイナリファイルに含まれている文字列を抽出し、公開されている脆弱性データベースと照合することで、IoT マルウェアが攻撃対象とした脆弱性を特定することである。

つまり、「①共通脆弱性識別子との関係性、②実装との関係性」という二つ視点から、

IoT マルウェアを調査対象にして提案手法を用いて調査を行った。調査結果は次の知見を明らかにした：

- (1) 10,213 検体から合計 46 個の攻撃に使用する脆弱性を特定し、検体数が最も多い脆弱性のトップ 3 は CVE-2017-17125(Huawei HG532 における入力確認に関する脆弱性)、CVE-2014-8361(Realtek SDK の miniigd SOAP サービスに存在する任意のコード実行を許してしまう脆弱性[CVE8361])と Linksys E-series の Remote Code Execution[JULERCE] (CVE 番号の付与なし)である；
- (2) 2019 年以後検体は、過去に使用されてない脆弱性を使用し、2018 年以前検体よりある程度進化している。

---

## 第4章 ランサムウェアに対する分類

---

本章では、ランサムウェアによるサイバー攻撃被害の特定と軽減という2つ目の課題を解決するために、ランサムウェア脅威の特定と被害の低減ための分類手法を提案する。提案手法は、ランサムウェアに呼び出されたAPIグループに着目し、これらのAPIグループ間の相関係数を特徴量にして、ランサムウェアを分類する手法を提案する。また、提案手法を用いて、各ランサムウェアファミリー分類に寄与度が高いAPIを特定した。

### 4.1 はじめに

ランサムウェアの被害を低減するために、ランサムウェアのファイルのアクセス、名前の変更、削除、暗号化などの動作に使用したAPIグループ間の相関性に着目し、ランサムウェアファミリーの分類手法を提案する。また、提案手法に基づく特徴量の寄与度分析手法により、ランサムウェア分類に寄与度が高いAPIを特定した調査結果を示す。

本章の構成は次のとおりである。4.2.1節で、ランサムウェアに呼び出されたAPIグループ間の相関係数をファミリー分類に特徴量として使用する際の有用性を明らかにしたうえで、APIグループ間の相関係数に基づいた提案方式の詳細を述べる。4.2節では、入手したデータセットを用いて提案手法の評価実験の設定を説明してから、4.3節では、評価実験の結果を述べる。4.4節では、特徴量寄与度分析手法を用いて、ランサムウェアファミリー分類に寄与度が高いAPIを特定した調査結果を示す。4.5節はまとめである。

### 4.2 提案システム

本節では、APIグループ間の相関係数を特徴量として利用する際の有用性を検証したうえで、APIグループ間の相関係数に基づいたランサムウェアファミリーの分類手法の詳細を述べる。各小節の概要は以下のように示す：



- 4.2.1 節では、API グループ間の相関係数を特徴量として利用する際の有用性の検証内容を述べる
- 4.2.2 節では、ランサムウェアファミリー分類の提案システムの概要を述べる。
- 4.2.3 節では、分類するために、特徴量に対する処理のフローを述べる。
- 4.2.4 節では、処理された特徴量を機械学習モデルに適用できるフォーマットへの変換と、ラベル付けの内容を述べる。

#### 4.2.1 ランサムウェア分類に使用する特徴量の有用性の検証

本小節では、ランサムウェア検体に呼び出された API グループ間の相関係数を特徴量として使用する際の有用性を明らかにするために、検証実験を行う。

Subedi らの研究[SBD2018]で提案された手法では DLL 間のコサイン類似度が用いられているため、API グループ間の相関によってランサムウェアが分類できると考える。したがって、API グループ間の相関性を特徴量として使用する前に、その相関性について調査した。

##### 4.2.1.1 相関性の調査方法

本調査では、API グループ間のピアソン相関係数を算出してから、統計図表を作成することで、特徴量の有用性を調査する。

具体的な調査手順は以下のように示す：

##### (1) API グループを定義する

(ア) 動的解析では、ランサムウェアの挙動を反映した OS やリソースと相互作用するシステムコールを利用するため[HBZ2018]、調査対象のコンテンツではまずタイプ別に API グループを特定し、そのうち 5 つの API グループを事前調査で使用した。ファイル関連の API である FileAPI グループ、暗号関連の API である CryptAPI グループ、レジストリキーの変更に使用する RegistryAPI グループ、ランサムウェアがネットワーク通信に使用する SocketAPI グループ、ランサムウェアがスレッドやファイルの実行に使用する ProcessAPI グループである。詳細は表 4.1 のとおりである。また、API グループ間の相関係数の略称は表 4.2 のように示している。

##### (2) 調査対象としたランサムウェアファミリーの API の呼び出し頻度を使用して API グループ間の相関係数を計算する。

(ア) 調査対象としたランサムウェアファミリーは、Cerber、CryptoWall、CryptoLocker、Jigsaw、Locky、Genasom、Petya、Reveton、TeslaCrypt である。表 4.1 にある API の呼び出し頻度を抽出し、API グループ間の相関係数を下記の数式(4-1)で算出する[SCIPY]：

$$R_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} * C_{jj}}} \quad (4-1)$$

ここで、 $i, j$  は API グループ  $i$  と  $j$  であり、 $R_{ij}$  は API グループ  $i, j$  間の相関係数である。C は API グループの共分散行列である。

表 4.1 API グループとその詳細

API グループ	API
FileAPI	FindNextFile, FindFirstFile, FindFirsFileEx, SetFilePointer, SetFilePointerEx, GetFileSize, GetFileSizeEx, SetFileAttributes, GetFileType, CopyFileEx, CopyFile, DeleteFile, EncryptFile, NtReadFile, NtWriteFile, GetFileAttributes, GetFileAttributesEx
CryptAPI	CryptDeriveKey, CryptDecodeObject, CryptGenKey, CryptImportPublicKeyInfo, CryptAcquireContext, CryptAcquireContextW
RegistAPI	RegCloseKey, RegCreateKeyExW, RegDeleteKeyW, RegQueryValueExW, RegSetValueExW, RegEnumKeyExA, RegOpenKeyExW, NtQueryValueKey, NtOpenKey
SocketAPI	socket, InternetOpen, shutdown, sendto, connect, bind, listen, accept, recv, send, InternetOpenUrl, InternetReadFile, InternetWriteFile
ProcessAPI	CreateThread, CreateRemoteThread, NtResumeThread, NtGetContextThread, NtSetContextThread, CreateProcessInternalW, NtOpenProcess, Process32NextW, Process32FirstW, NtTerminateProcess

表 4.2 API グループ間の相関係数の略称

略称	全称
FC	API Group for File and API Group for Crypt
FR	API Group for File and API Group for Registry
FS	API Group for File and API Group for Socket
CR	API Group for Crypt and API Group for Registry
CS	API Group for Crypt and API Group for Socket
RS	API Group for Registry and API Group for Socket
FP	API Group for File and API Group for Process
CP	API Group for Crypt and API Group for Process
RP	API Group for Registry and API Group for Process
SP	API Group for Socket and API Group for Process

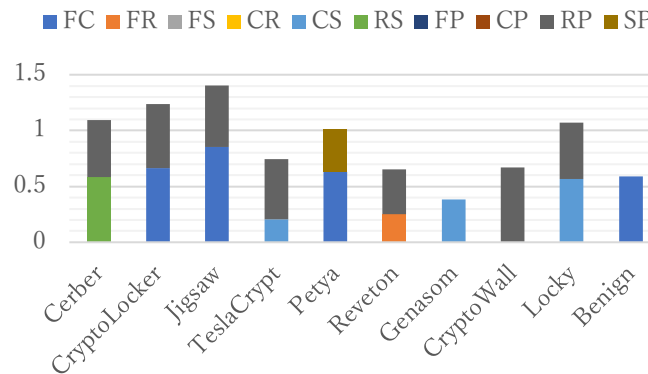


図 4.1 異なるファミリーの API グループ間における相関係数の比較

#### 4.2.1.2 調査方法に基づいた検証結果

図 4.1 に示すように、API グループ間のファミリーによる相関係数の比較した。図中、横軸はランサムウェアのファミリー名、縦軸は相関係数であり、相関係数の範囲は $[-1, 1]$ である。ただし、この統計図は縦棒を組み合わせたものであるため、合計は 1 を超える。

また、相関係数が 0.2 より小さい場合、統計は作成されない。図 4.1 に示すように、各ランサムウェアファミリーの API グループ間に相関の違いがあることがわかり、提案手法ではファミリーを分類する際に API グループ間の相関係数を特徴量として利用できることを明らかにした。しかし、分類する際には、データからマルウェアファミリーのすべての特徴を把握できるように、相関係数が 0.2 より小さくても特徴量として利用することとした。

#### 4.2.2 提案システムの概要

本小節では、API グループ間の相関性を特徴量とするランサムウェア分類の提案システムの概要を述べる。

提案システムの概要を図 4.2 に示す。

##### (1) 図 4.2 の①：マルウェア検体の動的解析

(ア) 動的解析によりマルウェア検体を実行させ、そのマルウェア検体の解析レポートを作成する。

##### (2) 図 4.2 の②：特徴量の処理

(ア) 生成されたレポートから、表 4.1 にある API の呼び出し頻度を抽出する。

(イ) 各 API グループ間のピアソン相関係数を取得する。

##### (3) 図 4.2 の③：機械学習でファミリーの分類

(ア) Support Vector Machine (SVM)を用いて分類する。

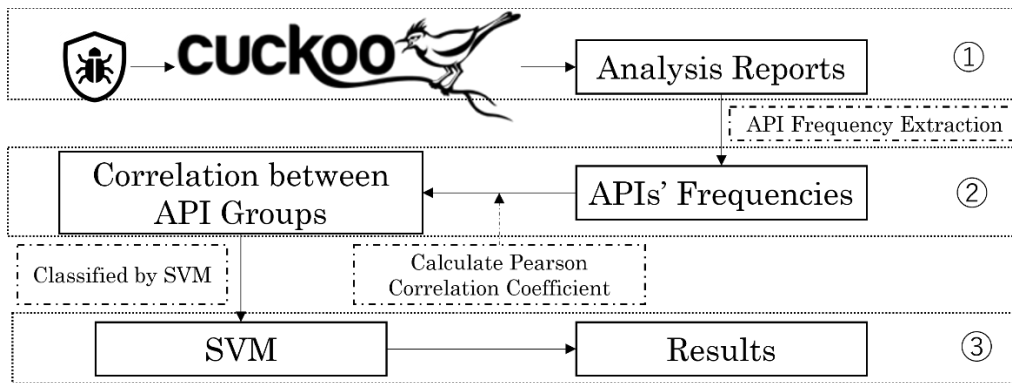


図 4.2 提案システムの概要

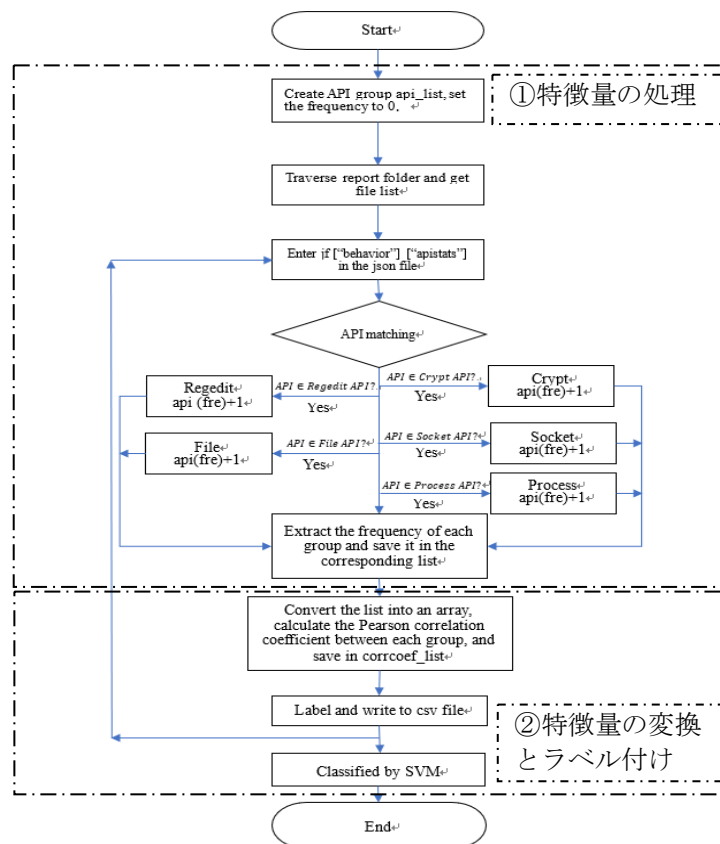


図 4.3 ランサムウェアを分類するための処理の概要

#### 4.2.3 特徴量の処理フロー

動的解析レポートから特徴量を抽出する際に、処理が必要であり、処理の流れは図 4.3 の「①特徴量の処理」の示すように、詳細は次の通りである。

##### (1) 処理フローの初期設定

- (ア) 表 4.1 に示す 5 つの API グループを作成し、API 呼び出し頻度を 0 に初期化した後、レポートフォルダを巡回し、各ファイルに対して処理フローを適用

する。

(2) 処理フローの適用

- (ア) `api_list` の API が["apistats"]に含まれる場合、`api_list` にある対応する API の呼び出し頻度を 1 増加し、対応するリストに保存する (`api (fre)` は API の呼び出し頻度である)。解析レポートは、JavaScript Object Notation (JSON)形式で保存し、呼び出された API の統計情報は["apistats"]に保存する。

#### 4.2.4 特徴量の変換とラベル付けの流れ

処理された特徴量を図 4.3 の「②特徴量の変換とラベル付け」の示すように機械学習モデルに適用できるフォーマットに変換し、ラベル付けを行う。詳細は以下のとおりである。

(1) 特徴量の変換と保存

- (ア) 特徴量の抽出後、頻度リストを配列に変換し、API グループ間のピアソン相関係数を計算し、"`corrcoef_list`"に保存する。1つのファイルに対して"`corrcoef_list`"が作成されたら、ラベルを付けてカンマ区切り値 (CSV) ファイルに書き出す。

(2) ラベル付け

- (ア) ラベルは、サンプルのファミリー名である。この作業を繰り返し、レポートフォルダ内のすべての JSON ファイルにおいて、対応する API の呼び出し頻度を抽出する。

最後に、作成された CSV ファイルを入力データとして、SVM により分類する。

### 4.3 提案方式の評価実験

本節では、ランサムウェアに呼び出された API グループ間の相関性を特徴量とするランサムウェアの分類手法の有用性と有効性を評価するために、その評価実験の内容を述べる。各小節の概要は以下のように示す：

- 4.3.1 節では、分類器の性能を改善するためのパラメータチューニングの詳細を説明する。
- 4.3.2 節では、評価実験で使用するデータセットの詳細を述べる

#### 4.3.1 分類器の最適化設定

分類器の学習不足または過学習の常態を解消すること、それに付随して分類器の性能を改善するために、分類器の最適なパラメータを求める必要がある。最適なパラメ

ータの求める手法と取得した最適なパラメータは以下のように示す：

#### 1) 最適なパラメータの求める手法

評価実験では、Python 3.7 sklearn[SKLEARN]を使用し、4.3.2 節で述べたデータセットと GridSearchCV を用いて最適なパラメータを求め、ShuffleSplit を用いてクロスバリデーションを行う。SVM のパラメータの範囲を以下の示すように設定した：

- C: [1, 10, 100, 1000]
- kernel: [linear, rbf, poly, sig-moid],
- gamma: [0.001, 0.0001],
- degree: [2, 3, 4]

#### 2) 取得した最適なパラメータ

取得した最適なパラメータは SVC, C = 1000, kernel = linear である。

### 4.3.2 評価実験で使用するデータセット

評価実験に使用するランサムウェアのサンプルは、複数の公開サイト [HBANAS][VIRUSARE][VIRUSGN][THEZOO]から入手し、各ファミリーのサンプル数を図 4.4 に示す。入手した良性ソフトウェア(正常ファイル)はすべてポータブル実行可能ファイル (PE) format であった。さらに、SVM に使用するトレーニングデータとテストデータは、sklearn の train\_test\_split を使用してランダムに分割している。テストデータの大きさは 30%である。データセットがアンバランスなため、SVC のパラメータ「class\_weight」を「balanced」[SKLEARN][EEOSUNA]に設定し、アンバランスなデータセットが分類に与える影響を軽減している。

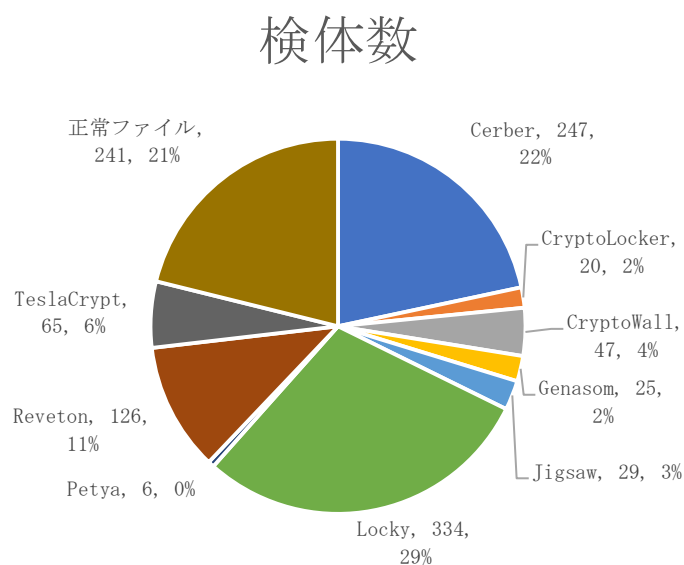


図 4.4 データセット

## 4.4 評価実験の結果

本節ではランサムウェアの分類手法の有用性と有効性を確認する評価実験の結果について説明する。各小節の概要は以下のように示す：

- 4.4.1 節ではランサムウェアファミリー分類の結果を記述し、さらに、この結果に対する分析を述べる。
- 4.4.2 節では分類結果の混同行列を用いて、誤分類状況と誤分類原因について述べる。

### 4.4.1 分類の結果

分類結果の評価尺度は、precision、recall、F1、support である。実験結果を表 4.3 に示すが、これは ShuffleSplit クロスバリデーションに基づいている。

ランサムウェア分類の精度は 0.98 に達し、先行研究[MEDHAT18]における学習結果精度 0.933、テスト結果精度 0.9414 より高い値を示している。これらの結果は、提案手法がランサムウェアの分類に有用であると考えられる。

表 4.3 実験結果

Families	Precision	Recall	F1	Support
Cerber	1.00	0.99	0.99	67
CryptoLocker	0.80	1.00	0.89	4
CryptoWall	1.00	0.89	0.94	18
Genasom	1.00	1.00	1.00	5
Jigsaw	1.00	0.92	0.96	12
Locky	0.99	1.00	1.00	119
Petya	1.00	0.50	0.67	2
Reveton	1.00	1.00	1.00	27
TeslaCrypt	1.00	0.95	0.97	20
Benign	0.94	1.00	0.97	68

### 4.4.2 分類結果に対する分析

分類結果を分析するために、図 4.5 に示すような混同行列を作成した。

#### 1) ランサムウェアファミリーの誤分類の確認

図 4.5 から、以下の誤分類状況を確認できた。

(ア) ランサムウェアファミリー Cerber 検体の一つ、CryptoWall 検体の二つおよび TeslaCrypt 検体の一つは正常ファイルに誤分類してしまった。

(イ) ランサムウェアファミリー Jigsaw 検体の一つは CryptoLocker に誤分類し

てしまった。

(ウ) ランサムウェアファミリーPetya 検体の一つは Locky に誤分類してしまった。

## 2) 誤分類原因

図 4.5 の示すように、例えば、ランサムウェアファミリーJagsaw の検体の一つが CryptoLocker に誤分類された。原因としては、図 4.1 にある Jagsaw と CryptoLocker の API グループ間の相関係数には FC(ファイルに関する API グループと暗号関連の API グループ間の相関係数)と RP(レジストリ変更関連の API グループとプロセス関連の API グループ間の相関係数)がある。そして二つファミリーにおける RP と FC の比率も非常に近いため、誤分類してしまったと考えられる。

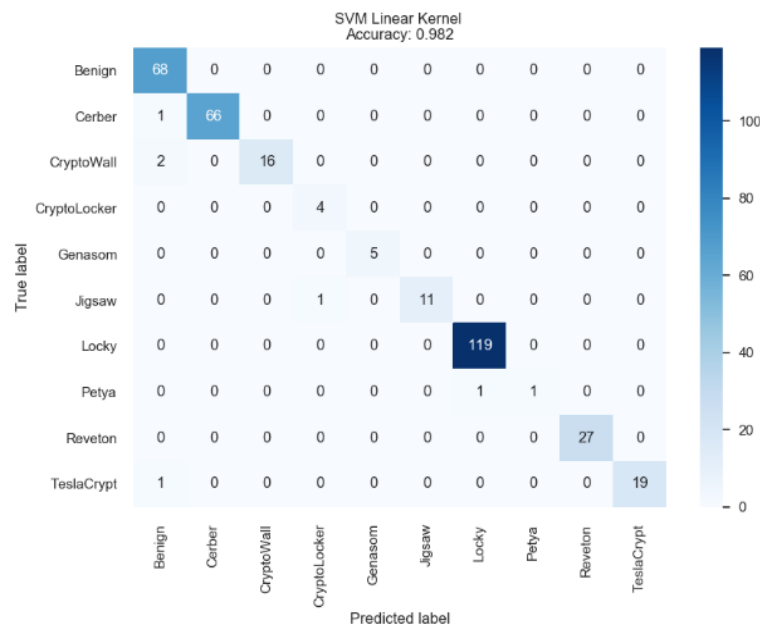


図 4.5 混同行列

## 4.5 考察

本節では、ランサムウェアファミリー分類における各ファミリーの分類に寄与度の高い API を確認するための特徴量分析内容を述べてから、提案方式の優れたところと弱さを示すための既存手法との比較内容を説明する。各小節の概要は以下のように示す：

- API 頻度に基づいたランサムウェアファミリー分類における各 API の寄与度の考察内容(4.5.2 節)を述べる。
- API グループ間の相関性に基づいたランサムウェアファミリー分類手法と既存手法との比較(4.5.2 節)を記述する



#### 4.5.1 ファミリー分類に各 API の寄与度

本節では API 頻度に基づいたランサムウェアファミリー分類における各 API の寄与度の考察内容を述べる。まずランサムウェアファミリー分類に各 API の寄与度の確認方法を説明してから、確認結果と分析結果を述べる。

##### (1) 各 API の寄与度の確認方法

(ア) 本手法では、`OneVsRestClassifier` では、クラスごとに 1 つの分類器を学習し、そのクラスのサンプルを陽性サンプル、それ以外のサンプルを陰性サンプルとするため、各分類後の特徴量 (`feature_importances_`) の `OneVsRestClassifier` を通して、出力する分類器として `RandomForestClassifier` を使用した。

##### (イ) 各 API の寄与度を確認する理由

特徴抽出の実験では、API の頻度から相関係数を計算し、API グループ間の相関の寄与度は、個々の API の寄与度を示すことができないため、特徴量として API の頻度を用いた。API の寄与度から、どの API がどの種類のランサムウェアを分類するために重要であるかを判断できる。

##### (ウ) 各 API の寄与度の確認手順

- ① まず API の頻度を特徴量とし、`OneVsRestClassifier`、`RandomForestClassifier` を分類器としてランサムウェアを分類した。この実験に使用したランサムウェアのサンプルは図 4.4 に示すとおりである。
- ② 通常のソフトウェアも含めると 10 ファミリーあるので、`RandomForestClassifier` のパラメータ `"n_estimators"` を 10 とした。また、ランサムウェアのサンプルの分割には `sklearn` の `train_test_split` を使用し、テストデータのサイズは 30%としている。API の頻度を特徴量とした分類の精度は 99%であった。
- ③ この実験結果は、API の頻度を特徴量として利用できることを示している。`OneVsRestClassifier` の特性を利用して、各分類後に現在のファミリーを分類するための API の寄与度を出力した。

##### (2) 各 API の貢献度の確認結果

(ア) ランサムウェアファミリーの分類に対する各 API の重要度（貢献度）を図 4.6 に示す。色の濃さは貢献度を表している。色が濃いほど、貢献度が高くなる。例えば、「`Process32Firstw`」は `Genasom` の分類に大きな影響を与えるが、`Reveton` にはあまり影響を与えない。同様に、`"NtOpenKey"` は `Locky` に大きな影響を与えるが、`Genasom` にはほとんど影響を与えないなど。表 4.3 の実験結果と図 4.5 の混同行列から、提案手法が `Cerber`、

CryptoWall、TeslaCrypt を良性、Jigsaw を CryptoLocker、Petya を Locky に誤分類している原因を明らかにした。まず、"SetFeilePointerEx", "NtWriteFile", "CryptoAcquireContextW", "RegEnumKeyExA", "NTGetContextThread" は分類上ほぼ同じ影響を与えた。

(イ) CryptoWall では、「GetFileSizeEx」、「RegOpenKeyExW」、「bind」、「Process32NextW」がほぼ同じ程度の影響を与えたことが誤判定の原因であることを明らかにした。

(ウ) TeslaCrypt については、「GetFileSizeEx」、「RegOpenKeyExW」、「InternetReadFile」が分類にほぼ同じ影響を与えた。

(エ) また、Petya については、"GetFileType", "RegOpenKeyExW", "send" の影響はほぼ同じであるため、Locky と誤判定している。

(オ) 最後に、Jigsaw は CryptoLocker と誤分類された原因は、"RegSetValueExW", "RegOpenKeyExW", "NtOpenKey", "connect", "Process32NextW", "Process32FirstW" が分類にほぼ同じ影響を与えたことである。

### (3) 分析結果

上記の結果から、ランサムウェアファミリーごとに分類に影響を与える API が異なることを明らかにした。したがって、ランサムウェアの駆除や分類を行う手法を提案する場合には、図 4.6 に示すように、ランサムウェアファミリーの分類に大きな影響を与える API を選択すればよいことになる。

しかし、提案手法は、無駄な API を呼び出してランサムウェアの挙動パターンを変えてしまうという弱点がある。私たちは API グループ間の相関係数を用いてランサムウェアを分類しているため、ランサムウェアメーカーが意図的に無駄な API を大量に呼び出した場合、私たちが提案した手法の精度が大きく低下してしまう。例えば、図 4.1 に示すように、CryptoLocker と Jigsaw の FC と FS の値は非常によく似ている。この場合、攻撃者がファイル関連の API を大量に呼び出すと、提案した手法では CryptoLocker と Jigsaw の分類が困難になる。



表 4.4 既存研究との比較

	特徴量抽出方法	分類方法	特徴量分析	精度
[MEDHAT18]	API 関数、暗号署名、ファイルキーワード、ファイル拡張子	新しいフレームワークによる YARA ルール	YARA ルールと特徴量グループ	94.14%
[ISLAM13]	動的な特徴量セット(API コール)、静的な特徴量	複数の特徴量	統合された特徴量セット	97.4%
[KAK19]	API コール、使用システムライブラリ、ファイル操作	Decision Tree, Random Forest, SVM, Hidden Markov Model	動的の特徴量	HMM:93.38%(API-Bigram) USL-Bigram+FS; J48:100% RF:100% SVM:100%
[SPR18]	ファイルパス間の類似性、ネットワークトラフィック、ミューテックス名、レジストリ名、リソース名のクラスタリング	機械学習による複数のインスタンス学習	オペレーティングシステムやネットワークリソースとのインタラクション	RF:94.3% Linear SVM:94.4% MLP: 93.8%
[KZWE16]	システムコール	ConvNet+LSTM	API の使用状況	89.4%
[LW19]	API シーケンス	BLSTM	API の使用状況	97.85%
[NBRRL19]	API コール、ほかの動的と静的特徴量	Machine Learning	動的と静的の特徴量	AdaBoost Classifier:93.84%
提案手法	API グループ間の相関係数	Linear SVM	動作間の相関性、各 API の重要度	98.2%

## 4.6 まとめ

本章では、ランサムウェアによるサイバー攻撃被害の特定と軽減という 2 つ目の課題を解決するために、ランサムウェア脅威の特定と被害の低減ための分類手法を提案した。提案手法は、ランサムウェアに呼び出された API グループに着目し、これらの API グループ間の相関係数を特徴量にして、ランサムウェアファミリー分類する手法である。

提案手法の評価結果としては、良性分類の結果は、すべて 0.9 以上で、良性ソフト

ウェアとして正しく分類されていると考えられ、ランサムウェアの分類精度は0.98に達し、先行研究の結果より優れている。また、本提案手法では、特徴量と動作の相関を分析することで、ランサムウェアの動作パターンを容易に把握し、各ランサムウェアファミリーの最も重要なAPIを決定することが可能であり、この機能により他のランサムウェア研究者が必要とする解析時間を短縮でき、二重脅迫型ランサム攻撃への対策の作成にも役に立つと考えている。

---

## 第5章 マルウェア攻撃に対する分類

---

本章では、より包括的なマルウェアファミリーの分類手法という3つ目の課題を解決するために、ランサムウェアに限らず、汎用性がより高いマルウェア分類手法を提案する。提案手法はマルウェアが悪意動作を実行するときに使用した API グループと操作したフォルダに着目し、API グループ間の相関性とフォルダ操作頻度を特徴量として使用し、機械学習で分類することである。提案手法を用いてマルウェアファミリーを特定した後、そのマルウェアファミリーが攻撃によく使用する API グループ、API の使い道を特定した結果を示す。

### 5.1 はじめに

既存研究で提案された分類手法はマルウェアを検出または分類できるが、それらの特徴量の抽出が複雑で難しいにも関わらず、マルウェアの動作の特徴も十分に表現することができていない。また、分類により、マルウェアファミリー、機能、目標、感染経路などの情報と紐づけることができるだけでなく、分類情報からマルウェアによる被害種別も想定できる。本章では、その課題を解決する提案方式について説明する。

本章の構成は次のとおりである。5.2 節で、本章の提案手法の概要を説明する。5.3 節では、本提案手法の有効性を評価するためのデータセットの詳細、特徴量の抽出方法及び実験の前提条件を説明してから、5.4 節では評価実験の結果を述べる。5.5 節では、評価実験の実験結果に対して考察した内容を説明する。

### 5.2 提案方式

本節では、マルウェアファミリーに限らず、より汎用性の高いマルウェアファミリー分類のための提案方式の概要と提案方式にある各手順の詳細を述べる。

提案方式は、マルウェアの動作パターンと特徴ある動作とを特徴量とする機械学習による分類手法である。本章では、動作パターンとして API グループ間の相関性、特

特徴ある動作としてフォルダ操作頻度を使用し、提案方式の有効性を明らかにする。動作パターンとして API グループ間の相関性、特徴ある動作としてフォルダ操作頻度を使用した提案方式(以下、提案分類方式と呼ぶ)は、API グループ間の相関係数の算出、フォルダ操作頻度の抽出、特徴量変換とラベル付けの 3 ステップからなる。

### 5.2.1 API グループ間の相関係数の算出

提案分類方式では、動作パターンとして API グループ間の相関性を使用する。動作パターンとして API 間の関係ではなく、API グループ間の関係に着目する理由としては、マルウェアファミリー毎の動作パターンや動作間関係を明らかにするために、抽象度を上げた API グループ化が有効であるという考え方に基づく。API グループとして、マルウェアのファイル操作に関連する FileAPI グループ、ファイル暗号化に関連する CryptoAPI グループ、レジストリ操作に関連する RegistryAPI グループ、ネットワーク通信に関連する SocketAPI および、スレッドまたはファイル実行に関連する ProcessAPI の 5 つのグループを使用する(表 5.1)。そして、API グループ間の相関係数については、API グループ毎の操作頻度を抽出した後、数式(5-1)[VS19]から算出する。

$$R_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} * C_{jj}}} \quad (5-1)$$

$i, j$  は API グループ  $i$  と  $j$  であり、 $R_{ij}$  は API グループ  $i$  と  $j$  間の相関係数であり、 $C$  は API グループの共分散行列である。

### 5.2.2 フォルダ操作頻度の抽出

提案分類方式では、特徴ある動作としてフォルダ操作頻度を使用する。フォルダに着目する理由としては、感染した PC のシステム情報、利用者情報、利用者が使用するアプリケーション情報など、マルウェアの用途が、参照するフォルダによって現れてくるであろうという考え方に基づく。

フォルダ操作については、実際にフォルダ配下にあるファイルを開く、ファイルをコピーする、ファイルに書き出す、ファイルから読み出す、ファイルを削除する、という 5 つの操作を使用する。フォルダ操作頻度の抽出では、これら 5 つのファイル操作が行われたフォルダ名を抽出する。

最後に、API グループ間の相関係数の計算とフォルダ操作頻度の抽出の処理を終了した後、相関係数と頻度を一つリストにまとめ、特徴量とする。

表 5.1 API グループ

グループ名	API
FileAPI	FindNextFile, FindFirstFile, FindFirsFileEx, SetFilePointer, SetFilePointerEx, GetFileSize, GetFileSizeEx, SetFileAttributes, GetFileType, CopyFileEx, CopyFile, DeleteFile, EncryptFile, NtReadFile, NtWriteFile, GetFileAttributes, GetFileAttributesEx
CryptAPI	CryptDerveKey, CryptDecodeObject, CryptGenKey, CryptImportPublicKeyInfo, CryptAcquireContext, CryptAcquireContextW
RegistryAPI	RegCloseKey, RegCreateKeyExW, RegDeleteKeyW, RegQueryValueExW, RegSetValueExW, RegEnumKeyExA, RegOpenKeyExW, NtQueryValueKey, NtOpenKey
SocketAPI	socket, InternetOpen, shutdown, sendto, connect, bind, listen, accept, recv, send, InternetOpenUrl, InternetReadFile, InternetWriteFile
ProcessAPI	CreateThread, CreateRemoteThread, NtResumeThread, NtGetContextThread, NtSetContextThread, CreateProcessInternalW, NtOpenProcess, Process32NextW, Process32FirstW, NtTerminateProcess

### 5.3 評価実験

本節で、FFRI が収集したマルウェアの動的解析ログである FFRI Dataset 2016[FFRI16]を使用した提案分類方式の有効性の検証(以下、評価実験)について述べる。各小節の概要は以下のように示す：

- 5.3.1 節では、評価実験を実施する前提条件の詳細にいて述べる。前提条件は以下のとおりである：
  - データセットの不均衡問題に対する対応
  - 教師データとテストデータの割合
  - 評価指標の選択
- 5.3.2 節では、前提条件に基づいて、評価実験用のデータへの処理フローについて述べる。処理フローには以下の手順が含まれている。
  - 特徴量の抽出手順
  - 分類のためのラベル付け手順
  - 特徴量数の最適化手順



### 5.3.1 評価実験の前提条件

本小節では、評価実験を実施する前提条件の詳細について述べる。

- データセットの不均衡問題に対する対応
  - 図 5.5 に検体に付与したラベル数の分布を示す。図中のひとつのブロックがひとつのマルウェアファミリーに該当する。この図からもわかる通り、マルウェアファミリー毎に、ブロックの大きさが異なっている。すなわち、マルウェアファミリー毎に検体として登録されている数に偏りがあり、FFRI Dataset 2016 は不均衡データセットであると言える。不均衡データセットは機械学習の精度に影響することから[PF2011], RandomOversample を使用してオーバーサンプリング有無による分類を試みる[LNA2017].
- 教師データとテストデータの割合
  - テストデータサイズは、全体の 30% とする[SMJ2004].
- 評価指標の選択
  - 有効性の評価にあたっては、評価指標として、Precision-Recall 曲線と Matthews Correlation Coefficient (MCC) 値[MBW1975]を用いる。MCC 値とは偽陽性と真陽性を考慮し、クラスが非常に異なるサイズであっても使用できるバランスの取れた尺度であり [BJE2017], 数式(5-2)から算出する。

$$MCC = \frac{c \times s - \sum_k^{\mathcal{K}} p_k \times t_k}{\sqrt{(s^2 - \sum_k^{\mathcal{K}} p_k^2) \times (s^2 - \sum_k^{\mathcal{K}} t_k^2)}} \quad (5-2)$$

この数式において、 $\mathcal{K}$  はクラスの総数である。

- $t_k = \sum_i^{\mathcal{K}} c_{ik}$  はクラス  $k$  の実際に発生した回数,  $c$  は混同行列である
- $p_k = \sum_i^{\mathcal{K}} c_{ki}$  はクラス  $k$  の予測された回数である
- $c = \sum_k^{\mathcal{K}} c_{kk}$  は正確に予測された検体の数である
- $s = \sum_i^{\mathcal{K}} \sum_j^{\mathcal{K}} c_{ij}$  は検体の総数である

### 5.3.2 評価実験用データへの処理フロー

本節で提案分類方式の有効性検証のために実施した FFRI Dataset 2016 から評価実験用データへの処理手順について述べる。

### 5.3.2.1 特徴量抽出手順

提案分類方式を検証するにあたり，FFRI Dataset 2016 から API グループの頻度の抽出，フォルダ操作頻度の抽出の具体的な手順について述べる．

#### (1) API グループの頻度の抽出

API 頻度の抽出アルゴリズムを図 5.1 に示す．表 5.1 にある各 API の頻度を FFRI Dataset2016 から抽出した後，各 API グループ間の相関係数を計算し，相関係数リスト( $list_{corrcoef}$ )に保存する(図 5.2)．

#### (2) フォルダ操作頻度

フォルダ操作頻度の抽出アルゴリズムを図 5.3，図 5.4 に示す．まず FFRI Dataset2016 からフォルダに関する文字列を抽出した(図 5.3)後，抽出された文字列がファイルかフォルダかを判断する．フォルダである場合は，第 5.2.2 節に記述した 5 つの操作頻度を抽出する．ファイルである場合には，ファイル名を削除し，フォルダ名を抽出し操作頻度を積算する(図 5.4)．

---

#### Algorithm 1: Correlation coefficient between API Groups

---

**Input:**

```

APIs' statistic information set:  $S$ ;
API groups nested array:  $list_{apig}$ ;
//  $list_{apig}$  Structure:
//  $\{ \{ api_1 : fre_{api_1}, api_2 : fre_{api_2}, \dots \},$ 
//  $\{ api_1 : fre_{api_1}, api_2 : fre_{api_2}, \dots \},$ 
//  $\{ api_1 : fre_{api_1}, api_2 : fre_{api_2}, \dots \},$ 
//  $\{ api_1 : fre_{api_1}, api_2 : fre_{api_2}, \dots \},$ 
//  $\{ api_1 : fre_{api_1}, api_2 : fre_{api_2}, \dots \} \}$ 

```

**Output:**

```

 $list_{corrcoef}$ ;

Create the frequency list of file operation-related:  $list_{files}$ ;
Create the frequency list of crypto-related:  $list_{crypto}$ ;
Create the frequency list of register-related:  $list_{registry}$ ;
Create the frequency list of socket-related:  $list_{socket}$ ;
Create the frequency list of process-related:  $list_{process}$ ;
Create the correlation coefficient list :  $list_{corrcoef}$ ;
// the frequencies of the APIs are extracted
// when they match
for  $process_{id}$  in  $S$  do
  for  $api_{name}$  in  $S[process_{id}]$  do
    for  $num$  in  $list_{apig}$  do
       $list_{apig}[num][api_{name}] \leftarrow$ 
       $list_{apig}[num][api_{name}] +$ 
       $list_{apig}[process_{id}][api_{name}];$ 

```

---

図 5.1 API グループ相関性抽出アルゴリズム：頻度の抽出

```

// Extract the frequency of the api and
// save it in their respective lists
// listapig Structure:
// [{api1: freapi1, api2: freapi2, ...},
// {api1: freapi1, api2: freapi2, ...},
// {api1: freapi1, api2: freapi2, ...},
// {api1: freapi1, api2: freapi2, ...},
// {api1: freapi1, api2: freapi2, ...}]
for num in listapig do
    for fileAPI in listapig[num] do
        listfiles ← listapig[num][fileAPI];
// Using numpy to calculate the
// Pearson Correlation Coefficient
// between API groups
fcorr ← numpy.correcoef(listfiles, listcrypto);
rcorr ← numpy.correcoef(listfiles, listregistry);
scorr ← numpy.correcoef(listfiles, listsocket);
pcorr ← numpy.correcoef(listfiles, listprocess);
ccorr ← numpy.correcoef(listcrypto, listregistry);
cscorr ← numpy.correcoef(listcrypto, listsocket);
cpcorr ← numpy.correcoef(listcrypto, listprocess);
rscorr ← numpy.correcoef(listregistry, listsocket);
rpcorr ← numpy.correcoef(listregistry, listprocess);
spcorr ← numpy.correcoef(listsocket, listprocess);
listcorrcoef ← fcorr, rcorr, scorr, pcorr, ccorr,
cscorr, cpcorr, rscorr, rpcorr, spcorr;
return listcorrcoef

```

図 5.2 API グループ相関性抽出アルゴリズム：相関係数の計算

**Algorithm 2: Folder Frequency Extraction Algorithm****Input:**

”summary” information from samples log’s generic: *GS*;

// *GS* Structure:

// {file<sub>opened</sub> : [filepath<sub>1</sub>, filepath<sub>2</sub>, ...],

// file<sub>copied</sub> : [filepath<sub>1</sub>, filepath<sub>2</sub>, ...],

// file<sub>written</sub> : [filepath<sub>1</sub>, filepath<sub>2</sub>, ...],

// file<sub>read</sub> : [filepath<sub>1</sub>, filepath<sub>2</sub>, ...],

// file<sub>deleted</sub> : [filepath<sub>1</sub>, filepath<sub>2</sub>, ...]}

**Output:**

dict<sub>folder</sub>;

Create file path array: list<sub>folder</sub> = [];

Create the ”summary” subscript list: list<sub>sub</sub> =

[file<sub>opened</sub>, file<sub>copied</sub>, file<sub>written</sub>, file<sub>read</sub>, file<sub>deleted</sub>];

Create folder frequency dictionary: dict<sub>folder</sub> = {};

Create folder operation type list: list<sub>fre</sub> = [0, 0, 0, 0, 0];

for num in list<sub>sub</sub> do

if list<sub>sub</sub>[num] is in *GS* then

for num<sub>opened</sub> in *GS*[num] do

if *GS*[list<sub>sub</sub>[sum]][num<sub>opened</sub>] is directory then

| list<sub>folder</sub> ← *GS*[list<sub>sub</sub>[num]][num<sub>opened</sub>];

else

| list<sub>folder</sub> ← *GS*[list<sub>sub</sub>[num]][num<sub>opened</sub>][0];

図 5.3 フォルダ操作頻度抽出アルゴリズム：フォルダリスト抽出

```

for  $num$  in  $list_{folder}$  do
     $dict_{folder}[list_{folder}[num]] \leftarrow list_{fre};$ 
for  $num$  in  $list_{sub}$  do
    if  $list_{sub}[num]$  is in  $GS$  then
        for  $num_{opened}$  in  $GS[list_{sub}[num]]$  do
            if  $GS[list_{sub}[num]][num_{opened}]$  is
                directory then
                    if  $GS[list_{sub}[num]][num_{opened}]$ 
                        is in  $dict_{folder}$  then
                             $dict_{folder}[GS[list_{sub}[num]][num_{opened}]] \leftarrow$ 
                                 $dict_{folder}[GS[list_{sub}[num]][num_{opened}]] +$ 
                                    1
                        else if
                             $GS[list_{sub}[num]][num_{opened}][0]$ 
                                in  $dict_{folder}$  then
                                     $dict_{folder}[GS[list_{sub}[num]][num_{opened}][0]]$ 
                                         $[num] \leftarrow$ 
                                             $dict_{folder}[GS[list_{sub}[num]][num_{opened}][0]]$ 
                                                 $[num] + 1$ 
            return  $dict_{folder}$ 

```

図 5.4 フォルダ操作頻度抽出アルゴリズムフォルダ：操作頻度の抽出

#### 5.3.2.2 分類のためのラベル付け手順

特徴量を抽出した FFRI Dataset 2016 の検体解析ログへのラベル付けについて述べる。ラベル付けは、特徴量を抽出した後、分類器を使って分類した結果の有効性を検証するためのものである。ベンダーが付与した検体名を使用し、ベンダーが付与した系列名に合致するよう分類が可能かを評価する。なお、評価実験でラベルを抽出できない検体解析ログを使用しないこととした。

ラベル付けにあたり、マルウェアのラベル付けツール avclass[SRPC16]を使用し、FFRI Dataset2016 が提供する各ウイルス対策ベンダーが付与した系列名を[“virustotal”]エントリから抽出する。avclass で抽出した各系列のうち、頻度が一番高い系列名を検体のラベルとした。なお、FFRI Dataset2016 の[“virustotal”]エントリには virustotal の各ウイルス対策ベンダーのスキャン結果が記録されている。

#### 5.3.2.3 特徴量数の最適化手順

分類器を使って分類した結果の有効性を検証するために実施した、除外すべき検体解析ログと特徴量数の最適化手順について述べる。

##### (1) 除外すべき検体解析ログ

すべての検体のラベルを抽出した後、各ラベルの検体数を積算する。その中、

検体数が 1 個しかないラベルは機械学習に使用できないため、その検体解析ログを評価実験で使用しない。

## (2) 特徴量の最適化

API グループ間の相関係数とフォルダ操作頻度を抽出したところ、特徴量の数は 9520 個に達し、機械学習の学習時間の減少およびモデルの精度を向上させるために、SelectFromModel[PF2011]を使用して特徴選択する。SelectFromModel は特徴の重要度で特徴を選択するため、今回の実験で特徴の選択用の閾値をデフォルト(すべての特徴の重要度の中央値“median”)にして、重要度が中央値より大きく、分類に最適な 234 個の特徴量を抽出した。

## 5.4 評価実験の結果

本節では、API グループ間の相関性とフォルダ操作頻度を使用したマルウェア分類手法の有効性を確認する評価実験の結果を示す。

### 5.4.1 オーバーサンプリングなし

オーバーサンプリングなしの分類では、Precision-Recall 曲線は図 5.6 の通りであり、ベンダーが付与した系列名に合致するよう分類できた正確度は 0.35 であり、精度は 0.32(赤色破線)で、MCC 値も 0.33 に留まった。この結果から、オーバーサンプリング無しの分類では、データセットの不均衡問題があり、API グループ間の相関性とフォルダ操作頻度を使用したマルウェア分類手法が有効ではないことを確認した。

### 5.4.2 オーバーサンプリングあり

不均衡データであることから、オーバーサンプリングを実施した場合には、ベンダーが付与した系列名に合致するよう分類できた正確度は 0.99 であり、精度は 100% (図 5.7 の赤色破線)となった。MCC 値は 0.988 であり、Precision-Recall 曲線(図 5.7)に示す通りであった。この結果から、オーバーサンプリングありの分類では、データセットの不均衡問題の影響を解消することで、API グループ間の相関性とフォルダ操作頻度を使用したマルウェア分類手法が有効であることを確認した。

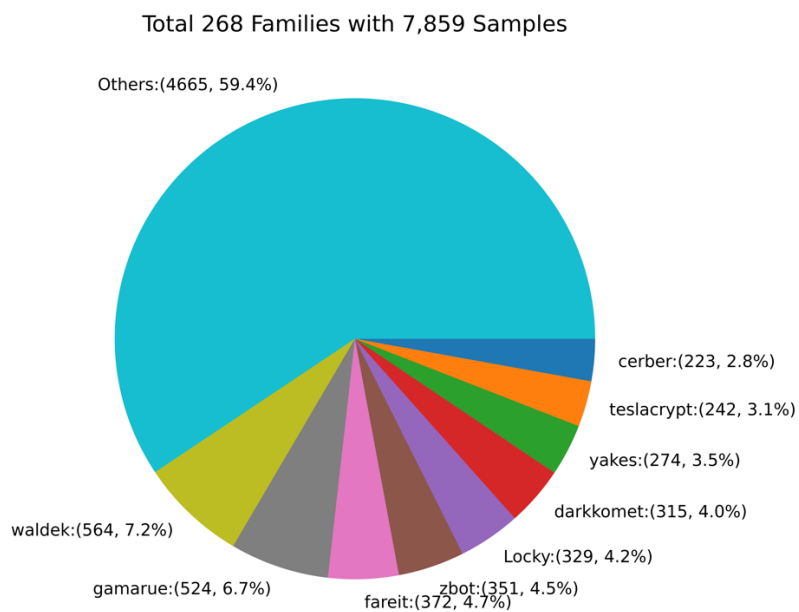


図 5.5 検体に付与したラベル数の分布

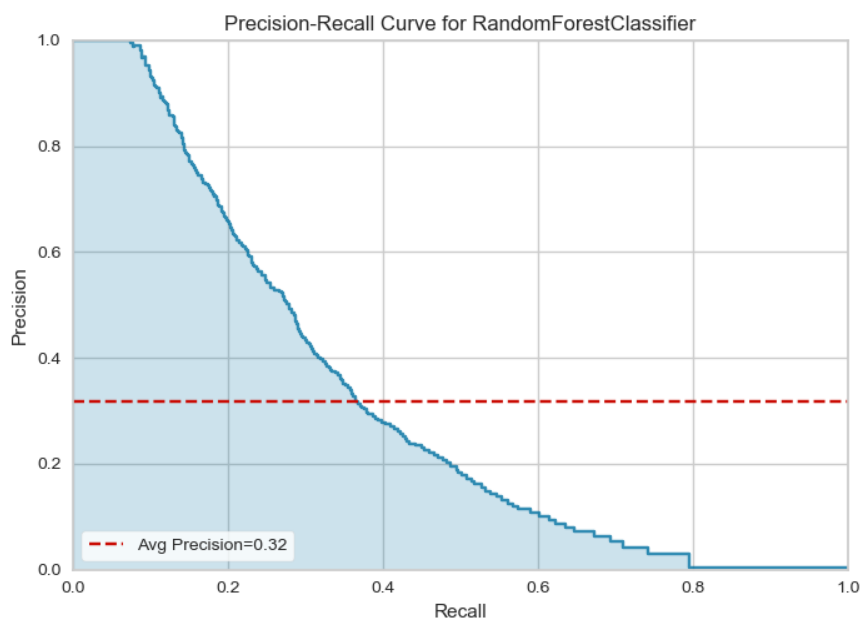


図 5.6 Precision-Recall 曲線 (オーバーサンプリングなし)

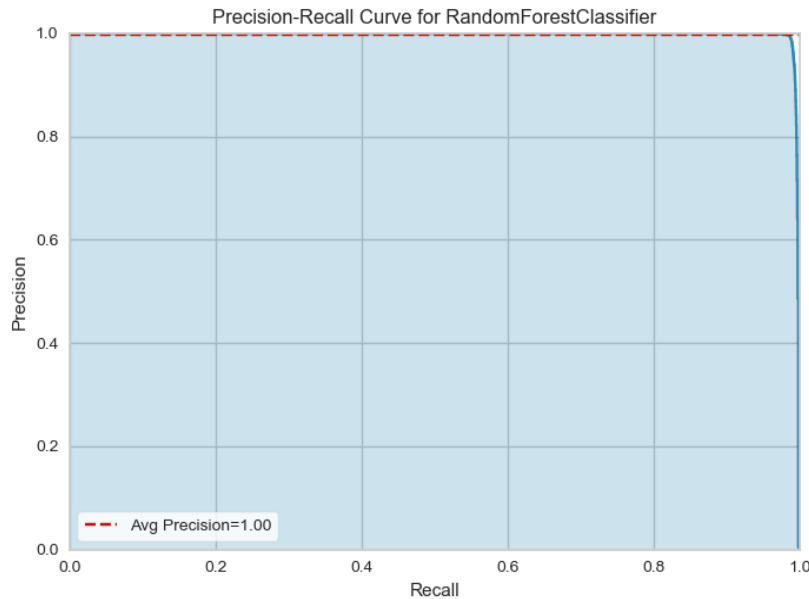


図 5.7 Precision-Recall 曲線 (オーバーサンプリングあり)

## 5.5 考察

本節では、まずオーバーサンプリング無しの分類における、低い分類精度に至る原因を明らかにするための分析内容を説明する。そして、オーバーサンプリングありの分類における、各マルウェアファミリーの攻撃によく使用する API と API の使い道を明らかにするための分析内容を示す。最後に、既存手法と比較することで、提案方式の優れた点と弱さを示す。

### 5.5.1 オーバーサンプリングなしが分類結果に与える影響について

オーバーサンプリングなしの分類精度が低い原因は、次の 2 つであると考えている。

(1) データセットの不均衡

不均衡データが原因で各マルウェアファミリーの特徴を十分学習できない可能性がある。

(2) API グループ間の相関係数の差が小さい

系列毎の **rp** (RegistryAPI と ProcessAPI) グループ間の相関係数を図 5.8 に示す。オーバーサンプリングあり/なしのいずれも、各系列の **rp** グループ間の相関係数はほぼ 1 である。差が小さいために分類が難しくなっているが、検体数を増やすと、改善する可能性があると考えている[CHCK2012]。また、オーバーサンプリングを使用して検体数を増やした場合、**rp** グループ間の相関係数の平均値は、

オーバーサンプリングなしの場合には 0.978307 であったものがオーバーサンプリングありの場合には 0.978344 となり，相関係数が改善する可能性のあることを示している．

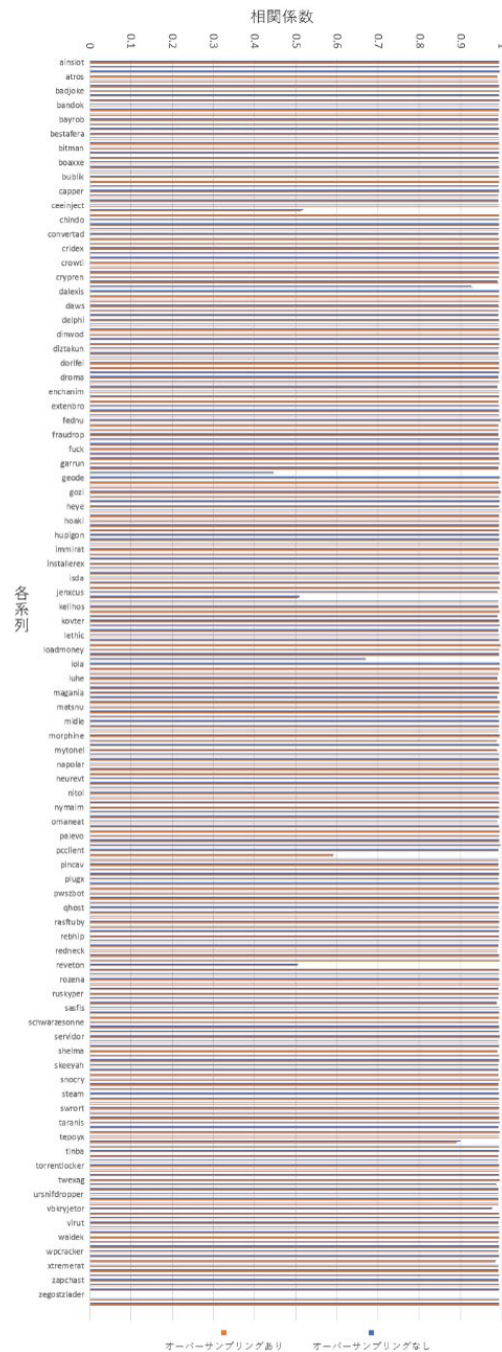


図 5.8 rp グループ間の相関係数の統計



### 5.5.2 オーバーサンプリングありにおける特徴量の寄与度について

本小節の概要は以下のように示す：

- 5.5.2.1 小節では、オーバーサンプリングありにおける特徴量の寄与度を分析する際に、使用する分類器の選定と各分類器の特徴について述べる；
- 5.5.2.2 小節では、特徴量の寄与度の分析結果を述べる。

オーバーサンプリングありにおける特徴量の寄与度については、5.3.2.3 節で抽出した 234 個の特徴量を対象とした分析について述べる。特徴量の寄与度を分析する目的としては、寄与度が高い特徴量から読み取れるマルウェアの動作上の特徴について検討することにある。

#### 5.5.2.1 特徴量の寄与度を分析する際の設定

本小節では、特徴量の寄与度を分析する際に使用する分類器の詳細について説明する。

##### (1) 分類器の選定

分析するにあたっては、分類器として `OneVsRestClassifier` と `RandomForestClassifier` を使用する。

この二つの分類器を選択した理由としては、`RandomForestClassifier` で分類した場合、特徴量の寄与度を出力できる。しかし、`RandomforestClassifier` から出力する寄与度は評価実験全体に対する寄与度であり、各系列の分類に対する寄与度ではない。したがって、`RandomforestClassifier` を `OneVsRestClassifier` の推定器として使用すれば各系列の分類に対する特徴の寄与度を出力できる。

##### (2) `RandomforestClassifier` の特徴

`RandomforestClassifier` は決定木に基づいているため[BL2001]、ツリーの決定ノードとして使用される特徴の深さを使用して、ターゲット変数の予測可能性に関する特徴の寄与度を評価できる。また、ツリーの最上部で使用される特徴は、入力データの大部分の最終的な予測決定に役立つため、これらの特徴が貢献する予測割合は特徴の寄与度の推定値として使用できる[LG2014]。

##### (3) `OneVsRestClassifier (OvR)` の特徴

`OneVsRestClassifier (OvR)` はマルチクラスストラテジーであり、クラス(本章の系列と同等)ごとに一つの分類器を適合させることにある。各分類器について、クラスはそのほかのすべてのクラスに対して適合される。計算効率( $n_{\text{クラス}}$  個分類器のみが必要)に加えて、`OvR` の利点の一つはその解釈可能性である。各クラスは一つの分類器のみで表されるため、対応する分類器を調べることでクラスに関する情報を得られる[BCM2006]。

### 5.5.2.2 特徴量の寄与度の分析結果

分析結果は、各系列の分類に寄与度が高いフォルダ上位 5 位を寄与度の高い API グループと共に表 5.2 に示す。

例えばマルウェアファミリー **cryptmod** の場合、**C:\Users\rihoko\Documents** フォルダのファイルを開くことが特徴ある動作となる。このフォルダ操作からマルウェア **cryptmod** の目的はユーザが作成したドキュメントフォルダ配下のファイルへの不正な操作であると推定でき、これはベンダー **Microsoft** の解析結果と一致している [MSRWC]。また、**cryptmod** のファイル操作と関連する暗号化、レジストリとプロセス操作を **virustotal** のスキャン結果と一致している [VSTAL]。

表 5.2 各系列の分類に寄与度が高いフォルダと API グループ(一部)

系列	寄与度が高いフォルダと操作タイプ	API グループ
sysn	C:\tmp2funuz\modules\packages を開いた回数	fr(Files-Registry) fp(Files-Process)
zegostzlad er	C:\ProgramData\Microsoft\Windows\AppRepository\Packa ges\Microsoft.Windows.Cortana_1.6.1.52_neutral_neutral_cw 5n1h2txyewy を開いた回数 C:\Users\rihoko\AppData\Roaming\Yfpey を開いた回数	fc(Files-Crypto)
fraudrop	C:\Windows\Microsoft.NET\Framework64\v4.0.30319\Setu pCache\v4.6.01055\1031 を開いた回数	fs(Files-Socket)
cryptmod	C:\Users\rihoko\Documents を開いた回数	fc(Files-Crypto) fr(Files-Registry) fs(Files-Socket) fp(Files-Process)
dynamer	c:\\$WINDOWS.~BT\Sources\Panther をコピーした回数	fc(Files-Crypto)

### 5.5.3 既存手法との比較

機械学習を用いたマルウェアの評価実験において、実験結果に影響を与える要素として、特徴量の選択と分類器の設定がある。

表 5.3 では、特徴量の選択と分類器の設定の視点から、提案分類方式の評価結果と既存手法の結果の比較を示す。既存研究では、静的な特徴量(関数の長さ、印刷可能な文字列等)、または動的な特徴量(API 名、パラメータ等)を使用してマルウェアを検知又は分類すると共に、実験精度に影響を与える要因を分析して改善方法も言及しているが、特徴量、マルウェアの動作の特徴と実験精度の関係を分析していない。一方、動作パターンとして API グループ間の相関性、特徴ある動作としてフォルダ操作頻度に着目し、それらが実験精度にどのように影響するかを分析し、API グループ間の相関を考慮することが分類に効果があることを示している。また、Kakisim らの手法で

の正確度は 100%であり、提案分類方式よりも高いが、提案分類方式は、マルウェアの動作間の関係を表現できる API グループ間の相関を明らかにしている点で優れていると考える。

表 5.3 既存手法との比較

手法	特徴量	検体数(データ分割比率)	分類器	正確度
Medhat et al.	API functions, cryptographic, etc.	2,568 (1,798:770)	検知手法そのものを提案	94.14%
Kakisim et al.	API call, usage system library	14,202 (10-分割交差検証)	HMM, SVM, J48, RF	100%
Kolosnjaji et al.	System calls	4,753 (3-分割交差検証)	DNN, HMM, SVM	98.4%
Liu and Wang	API sequences	13,518 (8:1:1)	GRU, BGRU, LSTM, BLSTM, SimpleRNN	97.8%
Stiborek et al.	Similarity between behaviors	250,527 (5-分割交差検証)	分類手法そのものを提案	94.4%
Islam et al.	API calls	2,939 (10-分割交差検証)	FLF, PSI, dynamic, integrated 手法	97.4%
Santos et al.	Opcodes sequences, system calls trace	13,189 (10-分割交差検証)	KNN, DT, SVM, Naïve Bayes, Bayesian network	96.6%
提案手法	API グループ間の相関係数とフォルダ操作頻度	Oversampling あり 148,135 (7:3)	Random Forest (n_estimators=268)	99%
		Oversampling なし 7,765 (7:3)	Random Forest (n_estimators=268)	35%

注：データ分割比率は、(訓練データ：テストデータ)あるいは(訓練データ：テストデータ：検証データ)の比率を示す。

また、提案分類方式の評価は、FFRI データセットを用いているため、次の点から未知のマルウェアの分類には限界があると考えている。

- 攻撃者が意図的に大量の無駄な API を呼び出した場合、分類精度を低下できる。
- 未知のマルウェアの場合には、フォルダを特定できない。

## 5.6 まとめ

本章では、マルウェア攻撃による被害を軽減するためには、マルウェアファミリーに限らず、汎用性のより高いマルウェアファミリー分類手法を提案した。

提案手法は、マルウェアの動作パターンとして API グループ間の相関性、特徴ある動作としてフォルダ操作頻度を使用し、精度を向上するために `SelectFromModel` で重要な特徴量を抽出した後、`RandomForest` 分類器でマルウェアファミリー进行分类することである。

提案手法の有効性の評価においては、FFRI Dataset 2016 を用いて評価実験を行い、データセットの不均衡問題を解決するためにオーバーサンプリング手法を適用する場合は、分類精度が 99%に達成、既存手法より優れたことを示した。また、特徴量寄与度分析手法を用いて、マルウェアファミリーの攻撃によく使用する API、API の使い道も明らかにした。

---

## 第6章 結論

---

### 6.1 総括

本研究では、ニューノーマル時代に備えたマルウェア脅威による被害を低減するために、マルウェアの動作特性と動作パターンに基づいてマルウェアファミリーの分類手法の提案とマルウェアの攻撃手口、動作目的と経路などの分析を目的として、次の三つの研究を行った。

- 攻撃に使用された脆弱性の特定手法
- ランサムウェアに対する分類
- マルウェア攻撃に対する分類

第3章では、マルウェアが攻撃で使用した脆弱性の特定手法を提案した。静的解析を用いて、マルウェアバイナリファイルから抽出した文字列と公開されている脆弱性データベースと照合してマルウェアが攻撃に使用した脆弱性を特定するという手法である。VirusShare から入手したテレワーク製品を狙った IoT マルウェア検体データセットを通じて、本手法を適用することで、マルウェアファミリーと各脆弱性間の関係を明らかにした。

第4章では、二重脅迫型ランサム攻撃の被害を低減するために、ランサムウェアに利用された API グループ間の相関性を特徴量にして、機械学習を用いて、ランサムウェアを分類する手法を提案した。その後、入手したマルウェア検体データセットを通じてこの手法を評価してから、各ランサムウェアファミリーが分類によく使用する API を特定できた。

第5章では、マルウェアファミリーに限らず、汎用性がより高いマルウェアファミリー分類手法を提案した。この手法では、マルウェアが攻撃に使用した API グループ間の相関性と操作したフォルダパスを特徴量にして、機械学習を用いてマルウェアの動作特性と動作パターンを学習することでマルウェアファミリーを分類する。そして、特徴量寄与度分析手法を使用して、マルウェアがよく使用する API と攻撃目的な

どを特定した。

以上の研究結果から、ニューノーマル時代に備え顕在化したマルウェア脅威による被害を低減するために、既存のマルウェア分類手法を改善する有効な手法と位置づけることができたと考えている。

## 6.2 今後の展望

今後の展望として、一つ目は本論文で提案したマルウェア分類手法の精度を向上する必要がある。二つ目は機械学習モデルとパラメータの最適な組み合わせを探索する必要がある。

まず、マルウェア分類精度の向上である。今までの教師あり機械学習を用いたマルウェア分類手法において、教師データまたは訓練用マルウェア検体データセットにファミリー名を付ける方法は、VirusTotal からマルウェア検体の解析レポートを入手して、特定のセキュリティベンダーの結果か、専門家の解析結果か、または AVClass などのツールを用いて VirusTotal 解析レポートから抽出した結果などを使用する。しかし、各セキュリティベンダーのマルウェア検知と分類のシグネチャーデータベースがそれぞれ異なることだけでなく、各ベンダーのマルウェア命名法則も異なるため、複数のセキュリティベンダーが一つマルウェア検体に同じラベルを付与しても、これらのラベルが同じものなのかは実にはわからない状態である。したがって、現状のマルウェア分類精度を向上するためには、各セキュリティベンダーが付与するラベルを統一する必要がある。

機械学習モデルとパラメータの最適な組み合わせについては、マルウェアの分類において、使用できる特徴量が非常に多いが、その中に最もマルウェアの特徴を表現できるものを見つけるのは難しい。また、機械学習を用いたマルウェア分類では、機械学習のモデルの選定にも特徴量の種類を考慮する必要がある。したがって、汎用性のより高い、精度がより高いおよび信頼できる分類結果を得るためには、機械学習モデルとパラメータの最適な組み合わせを探索する必要がある。

## 謝辞

本研究をまとめるにあたり，大変多くの方々のご協力，ご支援をいただき，ここに深く感謝の意を表します．

本稿の執筆にあたり，懇切なご指導，ご鞭撻，並びに様々なご配慮を賜った東京電機大学 先端科学技術研究科 情報通信メディア工学専攻 寺田真敏教授に深く感謝の意を表します．

本研究は，大学内外の方々のご指導，ご助力を得て進められたものである．九州工業大学 大学院工学研究院 電気電子工学研究系 廣瀬幸准教授，東海大学 情報通信学部 柿崎淑郎特任准教授，大阪大学 サイバーメディアセンター 情報セキュリティ本部 猪俣敦夫教授，国立研究開発法人 情報通信研究機構のサイバーセキュリティ研究所サイバーセキュリティ研究室の方からご指導とご支援を頂き，心より感謝いたします．そして，情報セキュリティ研究室の皆様のご支援ならびに，ご協力にも感謝いたします．

最後に，本研究を終始あたたかく見守り励ましてくれた家族に感謝します．

## 致谢

韶光易逝，岁月荏苒。博士三年，一晃而过。在恩师的帮助与亲友的支持和陪伴下，我顺利完成了博士论文。值此之际，向曾给予我莫大帮助的恩师及亲友，表示由衷的感谢。

首先要感谢我的博导寺田真敏先生。寺田先生在日立的系统开发实验室工作时，因创建了日本国内第一个漏洞信息数据库而闻名，并获得了庆应义塾大学的工学博士学位。同时，他酷爱薯条的爱好也是众人皆知，并经常在 Facebook 上分享他去过的薯条店家信息。而且不管去什么店，只要有薯条卖，他都会点上一份，就如有一次和寺田老师去吃中式火锅，在就餐结束后，他果不其然的点上了一份薯条。另外，我是他来东京电机大学任教的第一位博士生，他在指导我时，事无巨细、言传身教，让我受益匪浅。在投稿我的第一篇期刊论文时，寺田先生手把手的教我如何撰写期刊论文，如何回复同行评审的意见、如何规范的书写邮件等等，得益于寺田先生的事无巨细的指导，我的第一篇期刊论文被成功收录。能拥有这样一位优秀而又有趣的导师，荣幸至极！

其次，我要感谢我的硕导猪俣敦夫先生。同时，猪俣先生也是我的预科生导师。在我报考东京电机大学的预科生时，是猪俣先生面试了我。当时的我日语能力不高，科研基础也很薄弱，所以聊的内容不是很深入，只问了一些基础问题，比如为什么要来日本啊，在国内学的什么啊等等，虽然聊的少，但还是能感觉到猪俣先生的和蔼与亲近。正是因为猪俣先生给了我一次读预科生的机会，而且在硕士期间，猪俣先生手把手的教我如何做学术，带我入门，我才有可能读到博士课程。当然，猪俣先生也是一位走在学术前沿的大佬，而且他有收藏老物件和种植花草的爱好，他也会经常分享他新收藏的物品（有时候是瞒着他夫人买的）以及花草的照片。在求学初期，能遇到如此优秀的学术启蒙老师，实乃人生一大幸事！

此处，我还要感谢我的挚友汪逸璟。在我的印象中，他是一位天资聪慧、温文尔雅、饱读诗书、博文强识的人才，同时他也非常的善解人意，游戏也打的非常棒。在我们考上大学之后，他在京我在皖，虽相隔甚远，但我们都会在闲暇之余，聊近况，诉衷肠。而在我出国之后，他知道我独自一人在外求学不易，会和我常聊在日本的学习和生活状况，还有一直嘱咐我，一旦遇到了合适的和心动的女孩就一定要抓住机会。在我初来日本时，正是有他的陪伴与支持，让我克服了最初的不适应。同时，他也是我的感情导师，我曾经也偶遇过一位佳人，可惜因为种种原因也只是有缘无份，幸好有他在我才能走出那段低谷。人生得一知己，何其有幸！

然后，我要感谢我的大舅，他也是我东渡日本的保证人。在我初至东瀛之际，大舅在生活上，给予了我莫大的帮助，帮助我适应日本的生活习惯，教我在日本的行为处事方式。同时，在我求学时，大舅教我如何与校方联系、如何写邮件、如何准备面试等等，事无巨细的帮助我考学。在我第一次去面试东京电机大学的预科生时，是大舅陪着我



的，独自一人在国外去面试时的那种不安、焦虑，都因为有大舅陪在身边而消失了大半。另外，我的大舅妈也非常关心我的生活，在我初到日本时，知道我还没有适应日本的生活习惯，经常做一些中国菜，喊我去吃饭，每次回家，也都会让我带很多食材回去；她也知道我厨艺不精，也送了我一个高压锅，叫我如何使用、使用之前要再三确认泄压阀等，也经常很仔细的教我如何处理食材。从我 2016 年来日本至今，正因为有他们的关心、帮助与支持，我才能继续我的求学之路，再次拜谢大舅与大舅妈。

最重要的，要感谢我的父母和姐姐。因家境贫寒，在我上小学之后，父亲就独自一人去上海打工挣钱，母亲在家照顾我和姐姐的生活。独自在上海打拼的父亲，非常节省，只为了能够存更多的钱，来改善家庭的生活；而母亲在家照顾我与姐姐的生活的同时，也要辅导我们的学习，另外还要忙于农活，非常的辛苦。但是即使家里经济困难，父母还是尽力的挣钱，供我和姐姐读书。因为他们小时候就是因为家里没钱才放弃读书，他们知道不读书，会吃更多的苦，知道只有读书，才能有更好的出路，才不会重蹈他们的覆辙。如果没有他们的坚持，也许现在的我并不是在这里写博士毕业论文，而是在某个地方辛苦的搬砖。另外，我要谢谢我的姐姐，从小到大她一直都很照顾我。初中的时候，我虽然是住校，但是在我放假时，只要她有时间，都会带我出去玩；在我上大学之后，我回家的次数就少了，但是每次回去，姐姐也都会带着我出去玩，带我去买衣服。现在她已结婚生子，有了一个非常可爱的小孩子丁锦弦。至亲恩情，大爱无疆，此恩无所为报，定当尽心竭力以进取，唯愿亲人久乐长安。

行文至此，我要感谢的人其实还有很多，有朋友、有同学、也有在兼职的时候认识的同事，他们在我人生的某个时刻，以某种身份出现在我的身边，帮助我、照顾我、关心我，让我成长、让我的人生更丰富、更有意义。人生虽有离别日，山水应有相逢时。遇见，是生命的缘分，愿他们能遇心之所爱、一路良人相伴。

最后，要感谢从未放弃、坚持到底的自己。来日留学至今已有 6 年，从最开始懵懵懂懂的少年，在经历过社会洗礼之后，逐渐走向成熟。一路走来虽路途艰辛，但得恩师、贵人相助，收获颇丰，而这段求学经历也将成为我一生中最宝贵的财富。

## 発表論文リスト

### 論文誌論文

- [1] 周家興, 廣瀬幸, 柿崎淑郎, 猪俣敦夫, 寺田真敏. “API グループ間の相関性とフォルダ操作頻度に基づくマルウェア分類手法の提案”. 情報処理学会論文誌 Vol.61 No.12, pp.1792-1801 (2020)

### 国際会議論文

- [1] Jiaxing Zhou, Miyuki Hirose, Yoshio Kakizaki, Atsuo Inomata. “Evaluation to Classify Ransomware Variants based on Correlations between APIs”. The 7th International Conference on Information Systems Security and Privacy, pp.465-472 (Feb.11-13, 2021)

**国内口頭発表**

- [1] 周家興, 寺田真敏. “IoT マルウェア基礎情報の調査”. 2022年暗号と情報セキュリティシンポジウム pp.1-8 (Jan.18-21, 2022)
- [2] 周家興, 廣瀬幸, 柿崎淑郎. “API 間の相関性に基づくランサムウェア亜種を区別する提案”. 情報処理学会 インターネットと運用技術 研究報告 Vol. 2018-IOT-43 No.11 , pp. 1-6 (Dec.6-7, 2018)

## 参考文献

- [DXGUIDE] デジタルトランスフォーメーションを推進するためのガイドライン (DX推進ガイドライン) Ver. 1.0, 経済産業省, Retrieved from: [https://www.meti.go.jp/policy/it\\_policy/dx/dx\\_guideline.pdf](https://www.meti.go.jp/policy/it_policy/dx/dx_guideline.pdf)
- [BCM2006] Bishop, C. M.: Pattern Recognition and Machine Learning, springer (2006)
- [BJE2017] Boughorbel, S., Jarray, F. and El-Anbari, M.: Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric, PLoS One (2017)
- [BL2001] Breiman, L.: Random forests, Machine learning, vol. 45, no. 1, pp. 5-32 (2001)
- [CFVS] Corvus Forensics, “VirusSahre.com,” VirusShare.com, 参照 Nov. 23, 2021
- [CHCK2012] Chu, C., Hsu, A., Chou, K. et al.: Does feature selection improve classification accuracy? Impact of sample size and feature selection on classification using anatomical magnetic resonance images, Neuroimage, vol. 60, no. 1, pp. 59-70(2012)
- [CUCKOO] Cuckoo Sandbox (2012, April 24). Retrieved from <https://cuckoosandbox.org/>
- [CVE17215] NVD, “NVD – CVE-2017-17215,” NVD - CVE-2017-17215 (nist.gov), 参照 Nov. 24, 2021
- [CVE8361] NVD, “NVD – CVE-2014-8361,” NVD - CVE-2014-8361 (nist.gov), 参照 Nov. 26, 2021
- [EPITBASE] Exploit Database, “Exploit Database – Exploits for Penetration Testers, Researchers, and Ethical Hackers,” Exploit Database - Exploits for Penetration Testers, Researchers, and Ethical Hackers (exploit-db.com)
- [EEOSUNA] EE Osuna.: Support Vector Machine: Training and Application. Doctoral Dissertation, Massachusetts Institute of Technology (1998)
- [FFRI16] 株式会社FFRI: FFRI Dataset 2016のご紹介, available from: <http://www.iwsec.org/mws/2016/20160530-ffri-dataset-2016.pdf>

- [GNUOBJ] GNU, “objdump(1) – IoT manual page,” objdump(1) - IoT manual page (man7.org), 参照 Nov. 23, 2021
- [HBANAS] Hybrid-Analysis (2013, May 18). Retrieved from <https://www.hybrid-analysis.com/>
- [HBZ2018] Hampton, N., Baig, Z., & Zeadally, S. (2018). Ransomware behavioural analysis on windows platforms. *Journal of Information Security and Applications*, 40, 44–51. <https://doi.org/10.1016/j.jisa.2018.02.008>
- [IPA2021] 情報セキュリティ10大脅威 2021: IPA 独立行政法人 情報処理推進機構, Retrieved from: <https://www.ipa.go.jp/security/vuln/10threats2021.html>
- [ISLAM13] Islam, R., Tian, R., Batten, L. M., & Versteeg, S. (2013). Classification of malware based on integrated static and dynamic features. *Journal of Network and Computer Applications*, 36(2), 646–656. <https://doi.org/10.1016/j.jnca.2012.10.004>
- [JULERCE] Johannes Ullrich, “Linksys E-series - Remote Code Execution exploit-db.com,” Linksys E-series - Remote Code Execution - Hardware remote Exploit (exploit-db.com), 参照, Nov. 26, 2021
- [KAK19] Kakisim, A. G., Nar, M., Carkaci, N., & Sogukpinar, I. (2019). Analysis and evaluation of dynamic feature-based malware detection methods. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11359 LNCS, 247–258. [https://doi.org/10.1007/978-3-030-12942-2\\_19](https://doi.org/10.1007/978-3-030-12942-2_19)
- [KZWE16] Kolosnjaji, B., Zarras, A., Webster, G., & Eckert, C. (2016). Deep learning for classification of malware system call sequences. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9992 LNAI, 137–149. [https://doi.org/10.1007/978-3-319-50127-7\\_11](https://doi.org/10.1007/978-3-319-50127-7_11)
- [LAC] 株式会社Lac, 相次ぐ「二重脅迫型ランサムウェア」の被害と対策～今、ペネトレーションテストにできること～, Retrieved from: [https://www.lac.co.jp/lacwatch/service/20210827\\_002700.html](https://www.lac.co.jp/lacwatch/service/20210827_002700.html)

- [LG2014] Louppe, G.: Understanding random forests: From theory to practice, arXiv preprint arXiv:1407.7502 (2014)
- [LNA2017] Lemaître, G., Nogueira, F. and Aridas, C. K.: Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning, *J. Mach. Learn. Res.* (2017).
- [LW19] Liu, Y., & Wang, Y. (2019). A robust malware detection system using deep learning on API calls. *Proceedings of 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference, ITNEC 2019*, (Itneec), 1456–1460.  
<https://doi.org/10.1109/ITNEC.2019.8728992>
- [MBW1975] Matthews, B. W.: Comparison of the predicted and observed secondary structure of T4 phage lysozyme, *BBA - Protein Struct.* (1975)
- [MEDHAT18] Medhat, M., Gaber, S., & Abdelbaki, N. (2018). A New Static-Based Framework for Ransomware Detection. *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, 710–715.  
<https://doi.org/10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00124>
- [MSRWC] Microsoft:Ransom:Win32/Crypmod.A!bit, available from: <  
<https://www.microsoft.com/en-us/wdsi/threats/malware-encyclopedia-description?Name=ransom:win32/crypmod.a!bit&ThreatID=2147734432> > (accessed 2020-02-11)
- [NBRRL19] Nunes, M., Burnap, P., Rana, O., Reinecke, P., & Lloyd, K. (2019). Getting to the root of the problem: A detailed comparison of kernel and user level data for dynamic malware analysis. *Journal of Information Security and Applications*, 48, 102365.  
<https://doi.org/10.1016/j.jisa.2019.102365>

- [NPACYBER] サイバー空間をめぐる脅威の情勢等 | 警察庁ウェブサイト,  
Retrieved from:  
<https://www.npa.go.jp/publications/statistics/cybersecurity/index.html>
- [NVD2013] BOOTH, Harold, et al., “The national vulnerability database (nvd): Overview,” 2013
- [NVDs] NVD - Statistics. Retrieved from:  
[https://nvd.nist.gov/vuln/search/statistics?form\\_type=Basic&results\\_type=statistics&search\\_type=all&isCpeNameSearch=false](https://nvd.nist.gov/vuln/search/statistics?form_type=Basic&results_type=statistics&search_type=all&isCpeNameSearch=false)
- [PF2011] Pedregosa F. et al.: Scikit-learn: Machine learning in Python, J. Mach. Learn. Res., (2011)
- [RADARE2] Radare2 Team, “Radare2 Book,” Introduction - The Official Radare2 Book, 参照 Nov. 23, 2021
- [SBD2018] Subedi, K. P., Budhathoki, D. R., & Dasgupta, D. (2018). Forensic analysis of ransomware families using static and dynamic analysis. *Proceedings - 2018 IEEE Symposium on Security and Privacy Workshops, SPW 2018*, 180–185.  
<https://doi.org/10.1109/SPW.2018.00033>
- [SCIPY] Scipy.org (2010, Oct.). Retrieved from  
<https://docs.scipy.org/doc/numpy/>
- [SKLEARN] scikit-learn (2008, Aug. 20). Retrieved from <https://scikit-learn.org/stable/>
- [SMJ2004] Shahin, M., Maier, H. and Jaksa, M.: Data division for developing neural networks applied to geotechnical engineering, *Journal of Computing in Civil Engineering*, vol.18, no. 2, pp. 105-114(2004)
- [SPR18] Stiborek, J., Pevný, T., & Rehák, M. (2018). Multiple instance learning for malware classification. *Expert Systems with Applications*, 93, 346–357.  
<https://doi.org/10.1016/j.eswa.2017.10.036>
- [SRKC2016] Sebastián, M., Rivera, R., Kotzias, P., & Caballero, J. "Avclass: A tool for massive malware labeling," In International symposium on research in attacks, intrusions, and defenses, pp. 230-253. Springer, Cham. September, 2016

- [SRPC16] Sebastián, M., Rivera, R., Kotzias, P. and Caballero, J.: Avclass: A tool for massive malware labeling, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (2016)
- [THEZOO] theZoo (2015, Aug. 5). Retrieved from <https://github.com/ytisf/theZoo>
- [VIRUSARE] Virussshare (2007 March 5). Retrieved from <https://virusshare.com/>
- [VIRUSGN] Virusign (2014 Jan. 6). Retrieved from <https://www.virusign.com/>
- [VS19] Virtanen P. et al., SciPy 1.0--Fundamental Algorithms for Scientific Computing in Python, pp. 1–22 (2019)
- [VSTAL] Virustotal, available from:  
<https://www.virustotal.com/gui/file/17b923990673b325ab31bbe045ba49d4f39a6b9412c36cce136da2ce9fbb8995/detection>