脅威情報とホワイトリストを用いたサイバー攻撃自動対処システムの研究

明治大学大学院先端数理科学研究科
２０１９年度
博士学位請求論文

脅威情報とホワイトリストを用いた サイバー攻撃自動対処システムの研究

A Study on Automated Defense System based on Threat Intelligence and White List

学位請求者　先端メディアサイエンス専攻

重本 倫宏

1. 研究背景と目的

1.1. 研究の背景

近年，民間企業や政府機関，制御システム等の重要インフラを狙ったサイバー攻撃が顕在化しており，個人，企業，国家それぞれの利益や安全性を損なうリスクが高まっている．特に APT（Advanced Persistent Threat）攻撃[1]は，秘密裏に，そして執拗に長期間攻撃を続ける点で従来の脅威とは異なり，マルウェアの侵入を検知して完全に防止することは不可能になりつつある．このような巧妙化する攻撃に対し，組織間でインテリジェンス[1]を共有して攻撃に備える集団防御の概念が浸透してきた．集団防御を実現するべく，ICT-ISAC 等の公益法人や FireEye[2], Threat Connect 等の民間企業がインテリジェンスの共有を進めている．しかし，インテリジェンス共有の仕組みは整いつつあるものの，インテリジェンスの活用が進んでいないのが実態である．例えば，2015 年 6 月には日本年金機構が標的型攻撃を受けて 125 万件もの個人情報を漏えいし[3]，これを端緒として短期間に同様のマルウェアによって東京商工会議所や早稲田大学等，計 44 もの組織の情報漏えい被害が生じた[4]．この標的型攻撃では EMDIVI[5]と呼ばれる遠隔操作型マルウェアが用いられていたが，このマルウェアの特性や対処方法等のインテリジェンスが適切に共有されて且つ対策に迅速に活用されていれば，これらの被害の発生は抑えられたと考える．

著者はこのような状況に鑑み，マルウェアの動的解析結果の情報や，共有されたインテリジェンスを活用することでサイバー攻撃に対して集団防御を実現する自律進化型防御システム AED（Autonomous Evolution of Defense）の研究を進めてきた[6][7]．マルウェアの中には，自身がインターネットと通信可能か否かを判断するために，感染初期に正規のサーバに対して疎通確認（HTTP 通信）を行うものが存在する．このため，マルウェアを動的解析した結果得られたマルウェアの通信先をそのまま HTTP プロキシ（以下，プロキシ）等のブラックリストに加え遮断対象とすると，業務へ悪影響（可用性の低下）が発生する可能性がある．このような不確実性の高い脅威情報を用いて対策を実現するところに AED 研究の難しさがある．そこで，マルウェアの動的解析や共有されたインテリジェンスから得られた不審サイト情報（ホスト名（FQDN）及び IP アドレス）をグレーリストとして管理し，クライアントが当該サイトへアクセスしようとした場合に，プロキシで追加認証を要求する．これにより，例え誤った情報による認証追加であったとしても人間による業務上必要なアクセスは許可しつつ，マルウェア等の機械によるアクセスを遮断することを試みる．しかし，最近は DGA（Domain Generation Algorithm）と呼ばれるドメイン生成アルゴリズム[8]を用いて，次々と新たなドメインを生成するものもあり，すべての接続先を予め把握することは困難になってきている．

[1] 脅威情報や IT 機器の脆弱性情報およびそれらに関する分析や対処支援情報

1.2. 現状の問題

現状のセキュリティ対処には 2 つの問題がある．

問題１：即時性のある脅威情報の入手困難性

従来のマルウェア対策は主に，セキュリティベンダなどが提供しているシグネチャ（マルウェア検査パターン）に基づくアンチウイルスソフトにより行なわれていた．アンチウイルスソフトは既知のマルウェアに対しては有効であるが，次々と発生する未知のマルウェアや，標的型攻撃に利用される高度な検知防止機能を持つマルウェアに対しては，ベンダ側でのマルウェア解析やシグネチャ作成が追いつかず，現状では十分対応しきれていない[9]．このような状況においては，次々と発生するマルウェアの挙動を解明し，脅威情報を入手することが重要となってきている．

また，NIST（National Institute of Standards and Technology）発行の Computer Security Incident Handling Guide[10]においても，インシデントに備えてインテリジェンスを収集することの有用性が情報源の一覧とともに示されている．NVD（National Vulnerability Database）[11]や JVN（Japan Vulnerability Notes）[12]等では構造化された脅威情報が配信されており，効率的な分析や脆弱性管理をはじめとした各種自動化に有用である[13]．一方で，最新の脅威情報は，まずブログ，ニュースサイト，および SNS といった媒体において，非構造データとして配信されることが多く，それらの情報が公開されてから構造化されるまでの間にはタイムラグが存在し，1 ヶ月以上を要する場合もある[14]．このため，最新の脅威情報に追従するには，非構造なデータを分析・活用する必要がある．

問題２：対処適用による業務影響

攻撃者に侵入された後の被害を抑制するためには，機密情報の窃取や感染拡大を目的とした悪意のある Web サイト（以降，悪性サイト）等，外部への通信を遮断することが重要である．悪性サイトへの通信を抑制する方法のひとつに，インテリジェンス（ブラックリスト等）を用いるものがある．しかし，インテリジェンスには，正規サイトが誤って含まれている場合があり，仮に業務遂行に必要な非悪性サイトがブラックリストに誤って含まれていた場合，当該サイトにアクセスできず，業務阻害の要因となってしまう．業務阻害を抑制する方法として，インテリジェンスを事前に精査し，非悪性サイトを人手で除外する方法も考えられるが，Web サイトの精査という別のコストが生じてしまう．

1.3. 従来手法と課題

問題 1 に対して，マルウェアの接続先を収集する方法として，マルウェアを特殊な解析環境で実行して，その振る舞いを観測する動的解析手法が研究されている．著者のグループでも，多種環境を用いて未知マルウェアの挙動を自動的に解明する，マルウェア動的解析システム（M3AS: Multi-modal Malware Analysis System）の研究を進めてきた[15][16]．M3AS を用いることにより，マルウェアの挙動解析を行うオペレータの負荷を大幅に削減しながら，マルウェアの接続

先を抽出することが可能となる．一方で，動的解析において，完全な機械化は難しく，詳細の確認や最終的な判断など，一部は人手に依る部分もある．マルウェアの数が膨大化しているという背景もあり，人手に依る部分をできるだけ削減し，解析をより効率化することが求められている．

マルウェアが膨大化している原因のひとつに，亜種マルウェアの存在がある．亜種マルウェアとは，既知のマルウェアと同様のプログラムを利用していたり，挙動が類似していたりするもののことである．このことから，解析対象を既知のマルウェアファミリに分類することができれば，そのベースとなる挙動が既に解析されており，専門家による手動解析が不要な亜種マルウェアを事前に選別し，解析効率を向上することが期待できる．

Obrst ら[17]は辞書やオントロジを作成することによって，ブログ，ニュースサイト，および SNS で配信される非構造データの構造化を試みている．しかし，セキュリティ分野では，新たなマルウェアの出現や脆弱性の発見・コードネームの付与等により，新しい語（未知語）が生まれやすいことから，継続的な辞書やオントロジのメンテナンスが容易ではない．

問題２に対し，多くの組織で，情報システム部門，あるいは SOC (Security OperationCenter) / CSIRT(Computer Security Incident Response Team)に属する専門家が，発生したインシデントに手動で対応してきた．ところが高度化，頻発化，迅速化するサイバー攻撃に対しては，今いる専門家の人手によるセキュリティ対策では限界があった．この状況においてセキュリティ対策の迅速対処にむけた施策がいくつか進められている．

その一つが SOAR（Security Orchestration Automation and Response）である．SOAR は，複数のセキュリティ機器を単一のシステムで統括コントロールし，事前に対処手順を登録しておくことで，自動的な対処を実現するシステムの総称である．商業的にはすでに製品が存在しており，Splunk 社の Phantom[2]や Paloalto 社の Demisto[3]などが著名である．しかし，これら SOAR 製品は，対策手順を Playbook としてあらかじめ登録しておく必要があり，セキュリティ対策によるリスク低減と，業務を考慮した可用性維持の両面を考慮したセキュリティ対策は行われていない．

淵上ら[18]は，組織内に流入する実行ファイルをサンドボックスで実行し，感染拡大の原因となる活動が観測された時点で，感染拡大に伴う通信を遮断する設定を組織内ネットワークスイッチの ACL に追加することで，感染拡大を抑制するネットワーク型動的解析システムを提案している．しかし，マルウェアが疎通確認のために行う正規サイトへの通信や，サンドボックス環境に導入した OS や，アプリケーションが行う正規の通信などが発生した場合には，誤って遮断してしまうという問題（偽陽性）が発生する．

また，マルウェアに感染したクライアントによる外部サイトへの情報漏えいを防止する技術として，インターネットへアクセスする際に CAPTCHA（Completely Automated Public Turing test to tell Computers and Humans Apart）[4]認証を要求する製品が存在する[19]．CAPTCHA

[2] https://www.splunk.com/en_us/software/splunk-security-orchestration-and-automation.html
[3] https://www.demisto.com/
[4] http://www.captcha.net/

は機械と人とを判別する逆チューリングテストであり，マルウェアのようなプログラム（機械）では CAPTCHA をクリアすることが出来ない特性を利用する．しかし，本製品は，外部サイトに接続する度に CAPTCHA 認証を行っており，日々の業務の妨げとなる．

1.4. 研究の目的

本論文では，標的型攻撃に対する早期対策の重要性に鑑み，業務影響を考慮した自動対処システムを提案する．このため 1.2 節で提起した２つの問題に対し，以下を目的とする．

目的１

即時性のある脅威情報の入手が困難であるという問題１に対して，マルウェア解析の高度化と公開情報の構造化により，攻撃者による遠隔操作や情報搾取を防止するために有用となる脅威情報を抽出すること．

目的２

抽出した脅威情報を用いて対処すると業務に悪影響を与える場合がある問題２に対して，業務への悪影響を抑えつつ，マルウェアによる被害発生リスクを軽減すること．

1.5. 研究の新規性

提案する自動対処システムの概要を図 1.1 に示す．

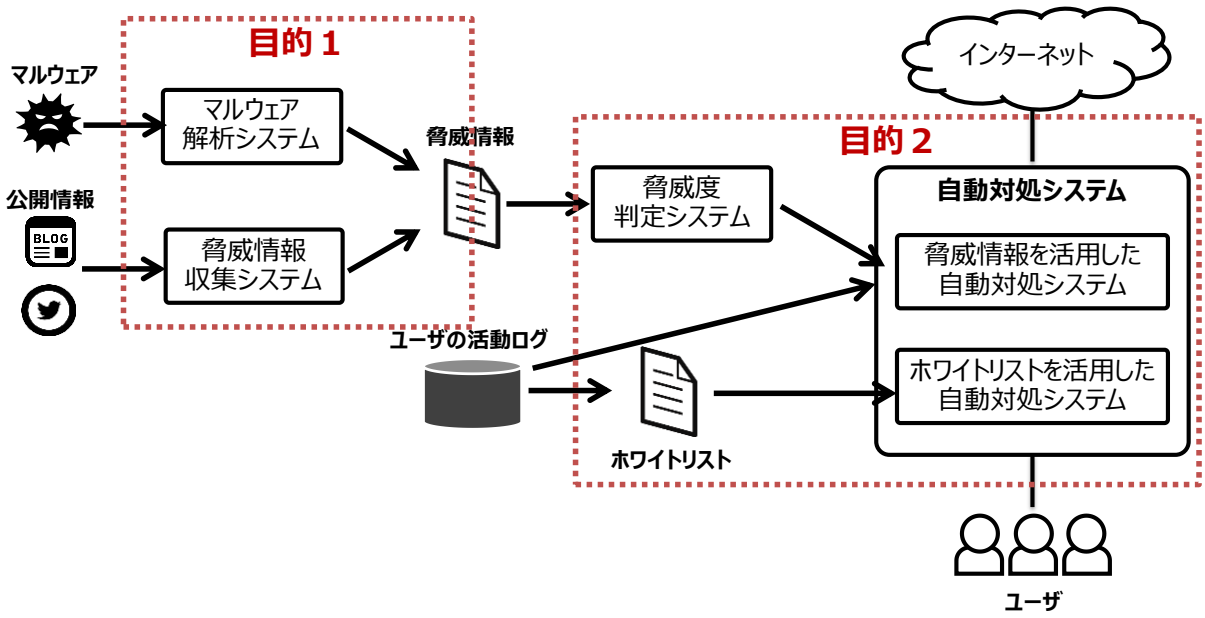


図 1.1　自動対処システムの概要

本論文で提案する自動対処システムは，マルウェアの動的解析結果や公開サイトから脅威情報を収集する（目的１）．解明して得られた結果から，業務影響を考慮した自動対処を実現する（目的２）．しかしながら，目的１を達成するにあたって，以下の課題がある．

課題１：マルウェア解析の高度化
課題２：公開されている脅威情報の構造化

そこで課題１を解決するために，複数の解析環境でのマルウェア解析結果を用いて，解析対象のマルウェアを類似する既存のマルウェアファミリに分類する分類手法を提案する[実績 10].

課題２を解決するために，公開されているセキュリティの脅威情報や脆弱性情報から固有表現を抽出・構造化し，インテリジェンスを自動で構築する技術を提案する[実績 4,5].

これらの技術によって即時性のある脅威情報を入手できる．しかし，これらの脅威情報には正規サイトも含まれる可能性があり，その情報を用いて機械的に対策（遮断）してしまうと 1.2 節で提起した問題２が発生する．このため目的２を達成するには，以下の課題がある．

課題３：脅威情報の高精度化
課題４：業務に影響を与えずにマルウェアの通信を遮断

課題３を解決するために，前述したマルウェアの自動解析によって得られた脅威情報や公開情報から収集した脅威情報の脅威度を推定する技術を提案する[実績 1,11].

課題４を解決するために，脅威情報の確信度と対処した際の影響度から対処の適用可否を決定する技術を提案する[実績 9]．さらに，まだ世の中に知られていない未知のサイトをマルウェアが利用した場合にも対処できるようにするため，ホワイトリストを活用した自動対処技術を提案する[実績 2,3]．また，ホワイトリストを用いた対策では，時間経過とともにホワイトリストを見直す必要があることから，ホワイトリストを最適化する技術を提案する[実績 6].

これらの提案を通じて，ユーザの業務に悪影響を与えづらい自動対処手法を実現する．

1.6. 位置づけ

本節では，提案する自動対処システムの位置づけを表 1.1 を用いて述べる．

比較対象として，手動対処と，CAPTCHA による対策[19]と，サンドボックス解析の結果を用いて通信遮断を行う技術を提案した淵上らの手法[18]とを用いて本論文で提案する自動対処システムの特徴を述べる．

表 1.1　研究の位置づけ

	提案手法	既存手法		
		手動対処	CAPTCHA 製品[19]	淵上ら[18] 提案手法
脅威情報の即応性	○ マルウェア解析の結果や公開情報で得られた脅威に即座に対応可能	× 人手による対処には時間が必要	○ 不審な通信は CAPTCHA により対応可能	△ サンドボックス解析で得られた脅威には対応可能
業務への悪影響	○ 業務影響を考慮した対処により悪影響を最小化	◎ 確実な情報に基づく対策のみ実施するため，副作用の発生は稀	× 毎回 CAPTCHA が出現することから，業務に影響が出やすい	△ 解析結果に正規サイトが含まれている場合に悪影響がでる
未知サイトへの対応	○ ホワイトリスト活用により対応可	× 対応不可	○ 不審な通信は CAPTCHA により対応可	× 対応不可

本論文の提案手法は，マルウェア解析の高度化や公開サイトの脅威情報収集により，最新の脅威に対応できる点で優位である．収集した脅威情報には，マルウェアのインターネット接続先 URL に関する情報も含まれる可能性があるが，脅威情報の脅威度を判定することにより，これら正規サイトを除外することが可能となる．

また，従来技術は，不審サイトへの接続リスクを軽減するか，業務への影響を軽減するか，どちらかに偏っていた．これに対し，提案手法は，脅威の確信度と業務影響を考慮して接続可否を判断したり，ホワイトリストを活用することで業務への悪影響を抑制しながら不審なサイトへのアクセスを遮断したりすることができる点で新規である．

著者が提案した AED は，不審サイトへの接続リスクと業務への影響を軽減するシステムである．しかし，最近は DGA のように，次々と新たなドメインを生成するものもあり，すべての不審サイトを予め把握することは困難になってきている．本論文の提案手法は，この問題を解決するものであり，未知のサイトを利用した攻撃にも対応することが可能となる．

1.7. 論文構成

論文の構成を図 1.2 に示す.

サイバー攻撃の自動対処を実現するために，本論文は脅威情報を収集する技術と，収集した脅威情報に基づいて自動対処を行う技術の 2 つによって構成される.

2 章では，関連研究を述べる．3 章では，マルウェア解析を効率化する技術[実績 10]について述べる．4 章では公開情報から最新の脅威情報を収集する技術[実績 4,5]について述べる．5 章では収集した脅威情報の危険度を推定する技術[実績 1,11]について述べる．6 章では脅威情報を活用した自動対処技術[実績 9]について述べる. 7 章は, ホワイトリストを活用した自動対処技術[実績 2,3]について述べる. 8 章ではホワイトリストの最適化技術[実績 6]について述べる. 9 章では, 本論文をまとめる.

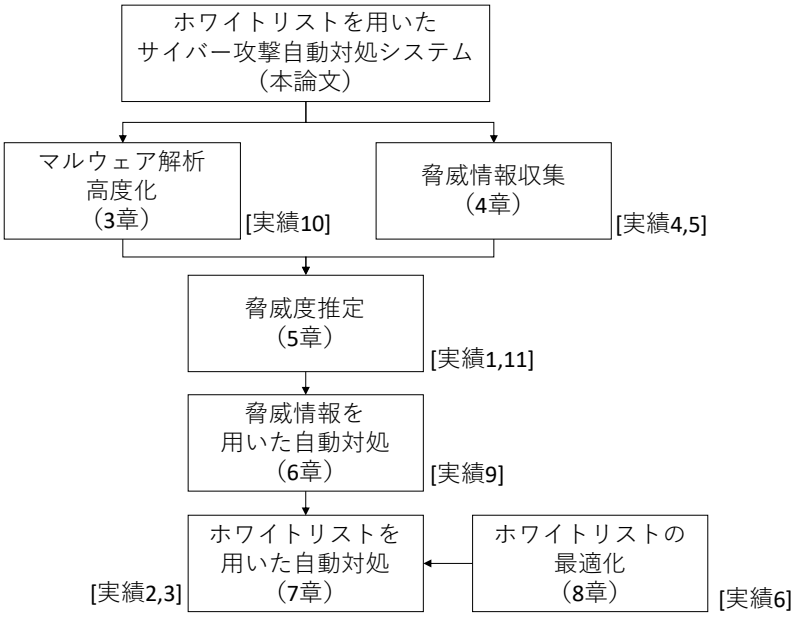


図 1.2　本論文の構成と各章の関係

2. 基本定義と従来研究

2.1. 用語の定義

本論文で用いる用語を以下に定義する．

(ア) マルウェア及び動的解析

マルウェアは，不正や攻撃を達成するために作成され用いられる悪意あるソフトウェアの総称である．本論文で扱うマルウェアには，Portable Executable などの実行ファイル形式に加え，JavaScript などスクリプト形式も含まれる．

動的解析はマルウェアの挙動を解析する技術の一種である．一般的には，サンドボックスと呼ばれる解析環境上でマルウェアを実行し，呼び出す API・読み書きするファイルやレジストリ・ネットワーク通信などの観点で挙動を監視・分析する．動的解析とは反対に，マルウェアを実行せずに挙動を解析する技術を静的解析と呼ぶ．

マルウェアおよび動的解析の具体例や詳細については次節以降で述べる．

(イ) 検知率の評価指標

表 2.1 に検知率を評価する評価指標の基となる混合行列を示す．例えば，マルウェアと予測したファイルが（予測値＝陽性），実際にマルウェアだった（真の結果=陽性）場合，予測結果は真陽性（TP: True Positive）と呼ばれる．同様に，予測値＝陽性，真の結果＝偽性の場合，予測結果は偽陽性（FP: False Positive）となる．また，予測値＝陰性であり真の結果＝陽性または偽性のとき，予測結果はそれぞれ偽陰性（FN: False Negative），真陰性（TN: True Negative）と呼ばれる．FN, FP はそれぞれ「検知見逃し」,「誤検知」とも呼ばれる．

検知率の代表的な評価指標は以下の通り求められる．

- 正解率(Accuracy)=(TP+TN)/(TP+FP+FN+TN)
- 真陽性率(TPR: True Positive Rate)=TP/(TP+FN)
- 検知見逃し率(FNR: False Negative Rate)=FN/(TP+FN)
- 誤検知率(FPR: False Positive Rate)=FP/(FP+TN)
- 再現率(Recall)=TP/(TP+FN)
- 適合率(Precision)=TP/(TP+FP)
- F 値(F-Measure)=2Recall・Precision/(Recall+Precision)

表 2.1　混合行列

	真の結果＝陽性	真の結果＝偽性
予測値=陽性	真陽性(TP)	偽陽性(FP)
予測値＝偽性	偽陰性(FN)	真陰性(TN)

2.2. 従来研究

2.2.1. マルウェア解析

(ア) マルウェア解析環境

サンドボックス環境を用いたマルウェア動的解析については，多くの研究がなされており，Anubis[20]，Ether[21]，CWSandbox[22]，TTAnalyze[23]，Panorama[24]等がある．しかし，近年のマルウェアは高度化が進んでおり，こうした動的解析の環境を検知してその動作を止めるような解析妨害機能を有しているものもある．Dhilung ら[25]は，Anubis は 84.78%，Ether は 78.18%の解析環境検知型のマルウェアに回避されてしまうという報告をしている．このような状況から，解析妨害機能を回避することが重要となっている．

解析妨害機能を回避するアプローチとして，複数の解析環境を用意することによってマルウェアの動作発現確率の向上を図るものがある．本アプローチを採用している手法の 1 つに，著者のグループで研究開発を行っている M3AS がある．M3AS は，マルウェアを多種類の解析環境で実行させることで環境を選ぶマルウェアをも自動的に解析し，その挙動や動作環境を推定してレポートする．また，複数の解析環境を用いるものとしては，BareCloud[25]も挙げられる．BareCloud は，複数の著名な解析環境と非解析環境に近づけた環境の両方を用意し，各環境で同じ検体を動作させた際に，その挙動に一定の差が見られた場合には，解析環境検知型のマルウェアであるとして検出する．

(イ) マルウェア推定手法

マルウェア推定手法とは，解析対象がマルウェアか否かを推定する手法のことであり，大きく静的解析による手法と動的解析による手法の 2 つがある．静的解析による手法の 1 つに，PE-Miner[26]がある．PE-Miner は，PE ヘッダの n-gram を特徴量に，決定木等のアルゴリズムを用いて解析対象がマルウェアか否か推定する手法である．また，Kolter ら[27]は，バイトコードの n-gram を特徴量とし，ナイーブベイズやサポートベクターマシン（SVM）等によって解析対象がマルウェアか否か推定している．動的解析による手法には，Ahmed ら[28]のものがある．Ahmed らの手法は，解析対象を動的解析し，API Monitor[29]を利用して取得した API の引数や呼び出し列を基に，各種機械学習アルゴリズムによって解析対象がマルウェアか否かを判定するものである．また，Tian ら[30]も，API 呼び出し列を用いたマルウェア推定手法を提案している．

マルウェア推定に静的解析を用いる場合，様々な解析妨害機能が施されている場合があり，

そのすべてを回避しつつ解析を実施することは難しく，汎用性にかける部分がある．例えば，マルウェア本体にパッキングが施されている場合には，アンパックによりマルウェア本体を抽出する必要がある．このパッキングには，独自のパッカーが利用されている場合も少なくなく，アンパックは容易ではない．また，動的解析を用いる際，マルウェアの解析妨害機能によってその挙動が発現しなかった場合は，適切に推定できない恐れがある．

(ウ) マルウェア分類手法

マルウェア分類手法とは，解析対象のマルウェアを既知のマルウェアファミリに分類するもの，あるいは類似の特徴を有するマルウェア同士を同じグループにクラスタリングするものである．マルウェア推定手法と同様に，静的解析によるものと動的解析によるものの 2 つに大別できる．静的解析によるものには，Iwamoto ら[31]の手法がある．Iwamoto らは，API 推移依存グラフの類似度を Dice 係数により算出し，階層型クラスタリングでマルウェアの亜種グループを抽出している．動的解析による手法としては，Christodorescu ら[32]の手法がある．Christodorescu らは，システムコールの呼び出しログの引数から，依存グラフを作成するとともに，グラフの類似度を算出することによって，亜種マルウェアを検出している．また，Rafique ら[33]は，http，smtp，udp，および tcp 等の各種ネットワーク挙動の特徴を捉え，解析対象のマルウェアをマルウェアファミリに分類する手法を提案している．

しかし，マルウェア推定手法と同様，静的解析の場合は，汎用性にかける，動的解析の場合は，挙動が発現せずに適切な分類ができない恐れがあるといった課題がある．

2.2.2. インテリジェンスの収集

インテリジェンスの構築は，有用である一方で，そのコストが大きいという課題がある．この課題を緩和すべく，様々な研究がなされている．

非構造データを構造化することにより，分析を効率化する，あるいは各種自動化に活用するというモチベーションの下，辞書やオントロジを作成する研究がある[34][35]．一方で，セキュリティ分野では，新たなマルウェアの出現や脆弱性の発見・コードネームの付与等により，新しい語（未知語）が生まれやすいことから，継続的な辞書やオントロジのメンテナンスが容易ではない．

また，機械学習の手法を用いて NER タスクを解くことにより，固有表現を抽出し，インテリジェンスを構造化する研究もある[36][37][38][39][40]．固有表現とは，組織名や地名といった固有名詞や日時や金額といった数値表現の総称であり，テキスト中から固有表現を認識・抽出することを NER と呼ぶ．この固有表現として，インテリジェンスからマルウェア名や製品名といった分析に有用な情報を抽出し，構造化を図るのが先述の研究である．一方で，これらの多くは教師あり学習によるものであり，大量のラベル付き（アノテーション済み）コーパスを教師として用いることにより高精度を実現する．しかし，セキュリティ分野をはじめ，ドメイン固有の分野

ではアノテーション済みコーパスが存在しておらず，教師データの確保が大きな課題として挙げられている[41]．この場合，自前でアノテーション済みコーパスを用意する必要があるが，このコストが大きいことがインテリジェンス構造化のボトルネックとなっている．具体的には，1 記事へのアノテーションに要する時間は，簡易なニュースでも中央値で約 8 分，場合によっては 1.73 時間を要するうえ，専門的な記事に対してはさらに時間を要するという研究報告[42]もあり，対象の教師データを用意することは容易ではない．IBM 社は本課題に対し，8 つの大学と連携してアノテーションすることによって解決を図っている[43]．

速報性の高い情報を収集することを目的に，SNS からベンダのパッチリリース情報[28]や脅威情報・脆弱性情報の抽出を試みている研究もある[44][45][46]．これらの研究は，速報性の高い情報を抽出可能という点で有用である一方で，情報抽出に留まっているものが多い．

2.2.3. WEB サイトの脅威度推定

Web サイトの性質を判別する研究は，以下の 3 種類に大別できる．

(ア) Web サイトへのアクセスが不要なもの

孫ら[47]は，既知の悪性 URL 群と近い性質を持った URL を未知の URL 群から抽出し，Bayesian sets と呼ばれる類似要素探索アルゴリズムを用いてブラックリストを構築することを目指したものである．Michael ら[48]は，URL 文字列ベースで決定木によって良性/悪性サイト分類を行っている．Sujata ら[49]は，URL のみからフィッシングサイトか否かを判定するものである．Page Rank やドメインにフィッシング特有の文字列が出現しているか否かを特徴量とし，ロジスティック回帰により判定を行っている．各研究の利点として，Web サイトへアクセスしないため，比較的判定が高速なことや攻撃者のアクセスログ等を取ることによる被解析検知を逃れられることが挙げられる．一方で，文字列のみを用いる場合後述する研究のように，Web サイトにアクセスして情報を取得する場合に比べると取得できる情報に限りがあり，精度の面ではやや劣る傾向が見られる．

(イ) 非悪性サイトへのアクセスが必要なもの

PREDETOR[50]は，ドメインの文字列や登録情報を用いて悪性か否かを分類するものである．悪用されるドメインの登録にはバースト性があることや失効したドメインが即時再取得された場合は，攻撃者による取得である可能性が比較的高いことが述べられており，悪性ドメインの分類に寄与することが実証されている．Justin ら[51]は，URL 文字列やホスト情報を用いた悪性サイトの推測器を提案しており，フィッシングに関連する URL は，そうでない URL に比べて URL が長いこと，ドメイン名が長いこと，およびドメインの生存期間が短いことを示している．千葉ら[52]の手法は，クライアント型ハニーポットでの URL の巡回を最適化するために，巡回候補 URL の悪性度を算出し，その高い順に巡回するものである．この悪性度を算出するために，SVM を利用し，特徴量には whois 情報や FQDN 文字列の特徴を

採用している．福島ら[53]は，悪性サイトが属する IP アドレスブロックとドメイン登録に用いたレジストラに着目し，両情報が既知の悪性ドメインのものと類似している場合，信頼性が低いドメインであるとして，ブラックリストに追加する手法を提案している．
EXPOSURE[54]は，主に DNS 情報を用いて決定木で悪性ドメインを見つけるものである．
これらの研究は，Web アクセスが不要な研究よりも推測に利用できる情報が多いため，精度が比較的高い傾向が見られる．一方の欠点としては，Web アクセスが不要な(ア)に比べると，推測に要する時間が比較的長いことや対象の Web サイトにアクセスする(ウ)に比べると取得できる情報に限りがあり，精度が劣る傾向が見られる点が挙げられる．

(ウ) 悪性サイトへのアクセスが必要なもの

EvilSeed[89]は，既知の悪性サイト情報を基に，クライアント型ハニーポットが利用する効率的な巡回クエリを生成し，悪性サイトを収集するものである．既知の悪性サイトリストを基（Seed）にして，リンクや DNS 情報等が類似しているページを検索するクエリを生成する．そのクエリを基に巡回したページの性質を Oracle と呼ばれる Google Safe Browsing 等からなるコンポーネットを用いて判定し，悪性と判断されたものは，再度検索クエリ作成の Seed とすることで効率的に悪性サイトを巡回することが可能となる．本研究のように，実際に悪性サイトへアクセスするものは，比較的時間を要する，クローキングへの対策が必要である等の懸念があるものの，精度の面で優れている傾向にある．

2.2.4. 自律進化型防御システム

著者のグループが提案した AED とは，不審 URL へのアクセス制御を，業務への悪影響を抑えつつ迅速に行うための技術である．AED は，いきなり通信を止めるのではなく，一旦グレーなものとして保留する．そして，グレーなサイトにアクセスしたとき，追加認証を行う．人間はアクセス先が悪性でないと判断した場合，文字を読み取り，追加認証をクリアして外部にアクセスできるが，マルウェアは文字を認識できず追加認証をクリアできないため，アクセスを引き続き防止可能である．なお，アクセス先が悪性か否かの判断を補助するために，ユーザの要求に応じてアクセス先のスクリーンショット等を取得する機能も有する．この追加認証が AED の第一の特徴である．そして，複数の従業員が追加認証をクリアした場合，安全なサイトとみなしてグレーリストを自動でホワイトリストに振り分ける．誰も追加認証をクリアできなかった場合，悪性なサイトだとみなしてブラックリストに振り分ける．このように，従業員の追加認証結果を統計的に分析して，アクセスポリシが自律進化していくことが AED の第二の特徴である．

AED を用いれば，ブラックリストを事前に精査せずとも自動で更新されるため，業務への悪影響がなく，かつ迅速に対処でき，運用を効率化することが可能となる（図 2.1）．

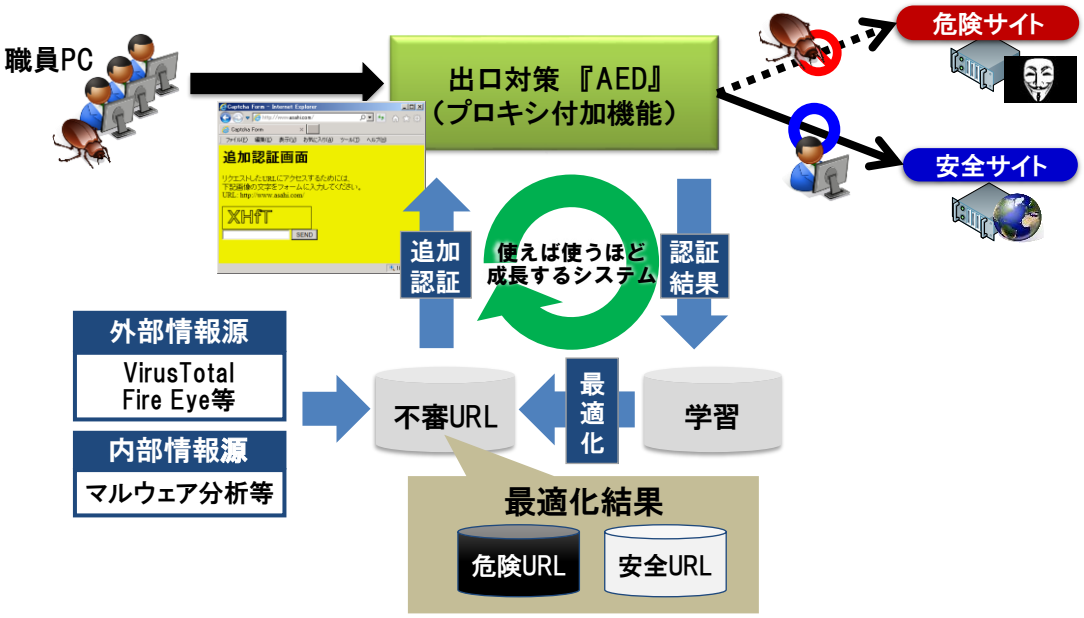


図 2.1　AED の全体像

2.2.5. ブラックリストを用いた対処

マルウェアの動的解析結果等の不確実なインテリジェンスを元に，プロキシ等の出口対策のブラックリストに活用する手法が畑田ら[56]より提案されている．また，悪性な社外サイトへのアクセスを防ぎつつも，安全性の高いサイトへのアクセス時には CAPTCHA 認証を省略することで業務への影響を軽減する手法が角田ら[57]に提案されている．

2.2.6. ホワイトリストを用いた対処

プロセスや通信に基づくホワイトリストを用いて標的型攻撃やマルウェアを検知する研究も進められている．中里ら[58]は，日常で利用しているプロセスか否かをプロセス情報から判断し，不審なプロセスを特定する手法を提案している．Takemori ら[59]は，ユーザが端末を操作していない時間帯に発生した通信をホワイトリストと比較することで，ボットを検知する手法を提案している．著者のグループでも，複数端末で行われる様々な種類の不審活動を分析し，攻撃者の拡散経路をグラフ構造として抽出することで拡散活動を検知する手法を提案している[60]．

時系列データを扱う機械学習の分野では，Sliding Window を用いて定期的にモデルを更新する方式が検討されている[61]．提案手法でも定期的に再学習を実施する点で Sliding Window に近い方法を採用している．

また近年，学習時及び運用時における，機械学習モデル自体に対する攻撃技術が，Adversarial Machine Learning として注目されている[62]．学習時における主な攻撃手段は，学習データに攻撃情報を混入させるものである．

3. マルウェア解析高度化

3.1. 導入

近年サイバー攻撃に用いられるマルウェアが高度化・膨大化しており，その脅威は年々増加している．マルウェア数が膨大化している原因のひとつに，亜種マルウェアの存在がある．このことから，解析対象を既知のマルウェアファミリに分類することによって，専門家による手動解析が不要な亜種マルウェアを事前に選別し，解析効率を向上することが期待できる．また，どのファミリにも属さないことが分かれば，当該マルウェアは新種のマルウェアだと判断でき，対策の優先度をあげるなどの判断を行うことが可能となる．

本章では，多種解析環境での解析結果を基にマルウェアを分類し，人手での解析を補助する手法を提案する．同手法は，解析環境間での動作の違いや API コール列等を特徴量とし，深層学習によって解析対象を既知のマルウェアファミリに分類する．

3.2. マルウェア分類手法

マルウェアの分類手法には，静的解析によるものと動的解析によるものがあるが，それぞれに汎用性に欠ける・解析に長時間を要する，解析環境によってはマルウェアの本来の動作が発現しない場合があるといった課題がある．動的解析の課題を解決する 1 つのアプローチとして，複数種類の解析環境を用意することで，マルウェアの本来動作の発現可能性を上げるという方法がある．本章では，複数解析環境におけるマルウェアの動的解析レポートを活用したマルウェア分類手法について検討する．

3.2.1. マルウェア分類手法の概要

マルウェア分類手法は，複数解析環境におけるマルウェアの解析レポートを活用し，マルウェアを分類するものである. 提案手法の全体像を図 3.1 に示し, 各構成要素について以下で述べる.

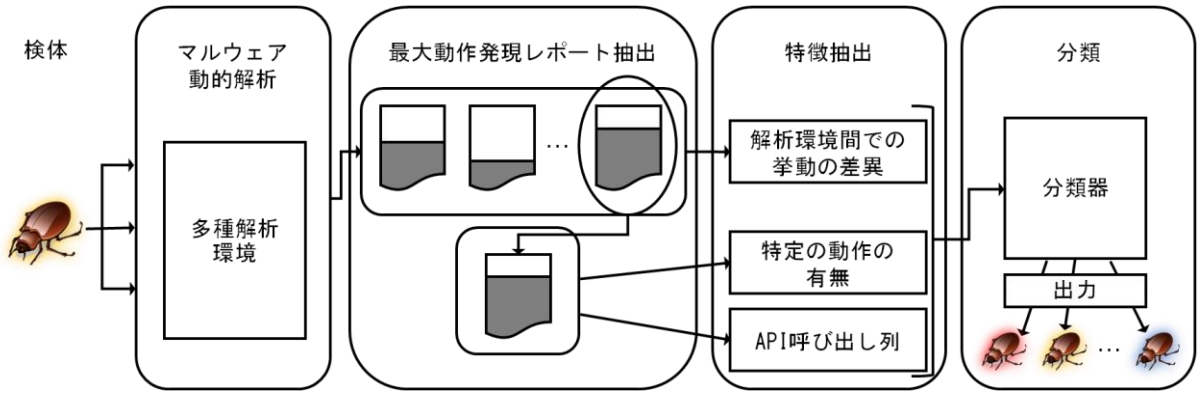


図 3.1　マルウェア分類手法の全体像

(ア) マルウェア動的解析

解析対象のマルウェアを受け取り，動的解析を行った後，その解析レポートを出力する．なお，前述のように，提案手法は多種解析環境における解析レポートを活用して解析対象のマルウェアを分類することから，多種解析環境上で実施されることを前提とする．

(イ) 最大動作発現レポートの抽出

多種解析環境から出力された解析レポートのうち，最も解析対象の動作が発現したものを抽出する．これは，最も動作が発現したということは，解析対象の本来の性質を最も現しているため，解析対象の適切な分類に有用であると考えられるためである．

(ウ) 特徴抽出

マルウェアの解析レポート，および抽出した最大動作発現レポートから特徴を抽出し，(エ)のマルウェア分類器に投入できる形にする．

(エ) 分類

投入された特徴量を基に，解析対象を分類し，その結果を出力する．

(ア)のマルウェア動的解析に用いる多種解析環境には，著者のグループが開発した M3AS のような既存の多種解析環境を利用するものとし，残る(イ)〜(エ)について，以下で詳述する．

3.2.2. 最大レポートの抽出

解析対象の本来の性質を可能な限り捉えるために，多種解析環境で動作させた中で，最も動作の発現したレポートを抽出する．マルウェアを含むプログラムの動作には，API の呼び出しが伴うことから，多くの動作を示す検体は，そうでない検体に比べて API の呼び出し数が多いと考えられる．このため，各解析レポートから，最も多くの API を呼び出したものを最大動作発現レポートとして抽出する．

3.2.3. 特徴抽出

本項では, 提案手法において, マルウェアを分類するために利用する特徴量について検討する．

(ア) API 呼び出し列

前述のとおり，マルウェアを含むプログラムの動作には，API が伴う．このため，API の出現頻度や出現順序がマルウェア毎の特徴を表していると考えられる．なお，2 章で述べたように，API の呼び出し列をマルウェアの推定・分類に利用する研究は多く，本手法においても，採用する．また，各 API は，呼び出されたユニークな API 次元数のベクトルで，表現したい

API に対応する 1 つの要素だけが 1 で他の要素は全て 0 である 1-hot ベクトル（1-of-K ベクトル）として表現する．

(イ) 解析環境間での挙動の差異

解析環境検知型のマルウェアは，自身の動作環境が解析環境であると判断した場合，例えば，自身のプロセスの終了など，真の挙動とは異なる動作を行う．多種解析環境において，そのようなマルウェアを動作させた際，解析環境間で，真の挙動と解析妨害挙動が混在する可能性がある．この現象を捉えることにより，解析環境検知型のマルウェアを検出できる．実際に，Lindorfer ら[63]は，動作環境間での挙動の差異が環境検知型のマルウェア検出に有用なことを実証している．この解析環境間での挙動の差異が，マルウェアの分類にも寄与するのではないかと考え，特徴量として採用した．具体的な特徴量を表 3.1 に示す．

表 3.1　解析環境間での挙動の差異に着目した特徴量

通番	特徴量	値
1	生成されたファイルの差異	0-1
2	tcp 通信先の差異	0-1
3	udp 通信先の差異	0-1
4	http 通信先の差異	0-1
5	https 通信先の差異	0-1
6	irc 通信先の差異	0-1
7	dns リクエストの差異	0-1
8	参照した hosts の差異	0-1
9	操作した mutex の差異	0-1
10	操作したレジストリの差異	0-1
11	user-Agent の差異	0-1
12	API 呼び出し数の差異	0-1
13	プロセス数の差異	0-1

これらは，ファイル生成や外部との通信など，マルウェアの動作として現れ得る部分を検討し，選定した．なお，通番 1～11 の値は，解析環境ごとの解析レポート A が NR 個存在する場合，以下の式 3.1 で算出する．

$$\mathrm{dif}(A_1, A_2, \cdots, A_{NR}) = 1 - \frac{|A_1 \cap A_2 \cap \ldots \cap A_{NR}|}{|A_1 \cup A_2 \cup \ldots \cup A_{NR}|} \qquad (式\ 3.1)$$

式 3.1 は，解析レポート間の共通項を全要素で割ることで導出される解析レポート間の類似度を 1 から減算することによって，解析レポート間の差異を算出している．また，解析環境間での挙動の差異が大きいほど 1 に，小さいほど 0 に近づく．

また，通番 12，13 の値は以下の式 3.2 で算出する．

$$dif(A_1, A_2, \cdots, A_{NR}) = 1 - \frac{\min(A_1, A_2, \ldots, A_{NR})}{\mathrm{Max}(A_1, A_2, \ldots, A_{NR})} \qquad (式\ 3.2)$$

式 3.2 は，算出対象の全解析レポートにおける最小数を最大数で割ることで導出される解析レポート間の類似度を 1 から減算することによって，解析レポート間の差異を算出している．また，式 1 と同様に，解析環境間での挙動の差異が大きいほど 1 に，小さいほど 0 に近づく．

(ウ) 解析環境間での挙動の差異

マルウェアごとの特徴を捉えるために，特定動作の有無を特徴量として利用する．利用する特徴量の一覧を表 3.2 に示す．なお，これらの特徴量は，Rafique ら[33]の手法を参考に，マルウェアに現れ得る動作を選定している．

表 3.2　特定動作の有無に着目した特徴量

通番	特徴量	値
1	自身の削除有無	0/1
2	自身のコピー有無	0/1
3	他プロセスへのコードインジェクション有無	0/1
4	自身の自動起動登録有無	0/1
5	実行可能ファイルの生成有無	0/1
6	メールの送信有無	0/1
7	tcp 通信の有無	0/1
8	udp 通信の有無	0/1
9	http 通信の有無	0/1
10	https 通信の有無	0/1
11	irc 通信の有無	0/1
12	hosts ファイルの書き換え有無	0/1

3.2.4. 分類

上述した特徴量を用いてマルウェアを分類する．API の呼び出し順序には，プログラムごとの特徴が反映されており，時系列を保持したまま利用するのが望ましい．そこで，API 呼び出し列は，時系列データとして利用し，その分類には，系列データを高精度で学習可能な深層学習手法

の 1 つである RNN（Recurrent Neural Network）を利用する．API 呼び出し列を時系列データとして利用することから，特徴量は，時系列のもの（API 呼び出し列）と非時系列のもの（解析環境間での挙動の差異，特定動作の有無）に大別できる．提案手法において，マルウェアを分類するために利用するネットワークを図 3.2 に示し，以降で説明する．

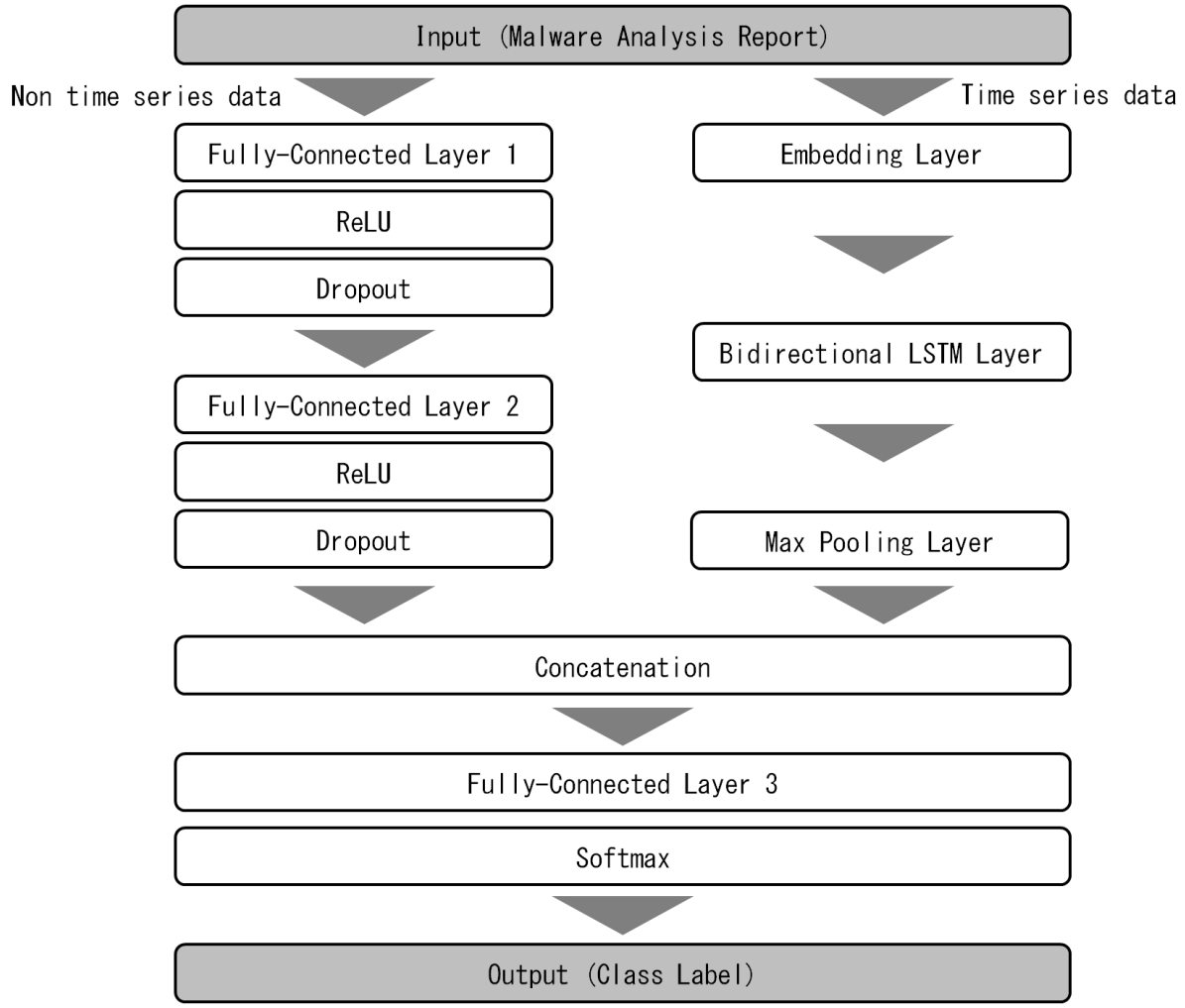


図 3.2　ネットワークの構成

時系列のデータである API 呼び出し列に関しては，まず，1-hot ベクトルを Embedding 層で分散表現にする．その後，Bidirectional LSTM（Long Short-Term Memory）層へ分散表現にした API 呼び出しの系列データを投入する．LSTM[64]は， RNN の一種であり，基本的な RNN よりも長期の記憶が可能な手法である．また，Bidirectional RNN[65]は，基本的な順向きの RNN に逆向きの RNN を加え，両者の出力を結合したものであり，順向きのみの場合よりも性能が良いとされている[66]．提案手法では，Bidirectional LSTM とし，LSTM を両方向の形で利用している．その後，各時刻における Bidirectional LSTM の全出力（系列）を max pooling している．

系列の max pooling は，テキスト分類でよく用いられる手法であり[67]，系列中の局所的な特徴が大きな値を持つように学習できた際に，max pooling によって，その値を最終的な出力に反映することができる．API 系列を用いてその API 群からマルウェアを分類する本タスクは，単語群からテキストを分類するタスクに類似していると考え，max pooling を採用した．

非時系列のデータには，全結合層を複数持つ順伝搬型のネットワークを利用する．また，それぞれにおいて，活性化関数には ReLU（Rectified Linear Unit）[68]を利用し，過学習を抑制するための Dropout 層を利用する．

上述の時系列・非時系列データを Concatenation 層で結合し，最終的に Softmax 関数によって出力を得る．Softmax 関数の出力は，各クラスへの所属確率となっている．つまり，解析対象のマルウェアが既存のマルウェアファミリのいずれに近いかを確率として出力する．

3.2.5. 期待される効果

提案手法は，上述の手法によって，課題であった複数解析環境におけるマルウェアの動的解析レポートを活用したマルウェア分類を実現する．また，これによって以下の効果が期待できる．

(効果 1)亜種マルウェアの解析効率化

解析対象が既知のマルウェアをベースに作成されたような亜種マルウェアであれば，本手法によって，両者を結び付けることができる．これにより，人手での詳細な解析の省略や解析の効率化が期待できる．

(効果 2)新種マルウェアの抽出

解析対象が新種のマルウェアであれば，本手法では，既知のどのマルウェアにも結び付かず，それによって新種ではないかということが推定できる．このように，本手法によって，より重点的に解析すべき新種マルウェアを抽出することが期待される．

3.3. 評価

3.3.1. データセット

評価では，FFRI Dataset 2016[69]を利用した．同データセットは，マルウェア対策研究人材育成ワークショップ（MWS）において提供されている研究用データセットの 1 つであり，Cuckoo sandbox[70]を用いたマルウェア 8,243 検体の解析結果である．同データセットには，同一検体を Windows 10，Windows 8.1 それぞれで解析した結果が収録されていることから，多種解析環境での解析結果として利用できると考え利用した．

また，訓練・評価する際の教師ラベルとしては，Cuckoo sandbox によって標準化された ESET-NOD32 の検知名を用いた．これは，検知漏れや特定の検知名への偏りが比較的少なかったためである．なお，データセットのうち，ESET-NOD32 が検知したものについてのみ利用しており，

その総数は 6,513，ユニークなラベル数は 315 である．

3.3.2. 評価項目

上述した 2 つの効果を検証するために，以下の評価を行う．

(評価 1)亜種マルウェアの分類精度

(評価 2)新種マルウェアの抽出可能性

利用するデータセットには，ユニークなラベル 315 のうち，2 検体以上の検体が含まれるものが 134，1 検体しか含まれないものが 181 ある．そこで，まずは前者の 134 を用いて分類精度を評価する．その後，1 検体しかいないものを擬似的な新種マルウェアとして新種マルウェアの抽出可能性を評価する．

評価に用いたネットワークのパラメータを表 3.3 に示す．

表 3.3　評価に用いたネットワークのパラメータ

層種	入力サイズ	出力サイズ	関数
Fully-Connected 1	25	25	ReLU
Fully-Connected 2	25	25	ReLU
Embedding	294*系列長	128*系列長	-
Bidirectional LSTM	128*系列長*2	128*系列長*2	-
Max pooling	128*系列長*2	128*2	-
Concatenation	25+256	281	-
Fully-Connected 3	281	134	Softmax

評価に用いたデータセットには，294 種類の API が含まれていたことから，Embedding 層への入力は 294 次元に，分類先の既知マルウェアは 134 種類であることから出力は 134 次元になっている．このとき，optimizer に AdaGrad[71]，Dropout 層におけるドロップアウト率に 50%，エポック数に 50 を用いている．ミニバッチサイズには，32 を利用し，可変長である API 呼び出し列をバッチ処理するために，パディングを挿入し，長さを統一している．また，多クラス分類であるため，損失関数には，交差エントロピーを用いている．さらに，過学習抑制のために，early stopping を利用している．なお，評価環境は，表 3.4 に示すとおりである．

表 3.4　評価環境

項目	スペック
CPU	Intel Core i7-6700
GPU	GeForce GTX 1080
メモリ	8,192 MB
OS	Ubuntu 16.04 LTS

3.3.3. 亜種マルウェアの分類精度

本評価では．ESET-NOD32 において同じラベルが付けられた検体を同じクラスに分類できるか検証した．データセット内に 2 検体以上含まれる 134 種類の検体のうち，ランダムに 8 割を訓練データ，2 割を評価データとした．前述のとおり，出力は各クラスへ所属する確率となることから，出力のうち，最も確率の高いクラスと正解クラスが一致した場合に，分類に成功したとして，評価データを分類した際の正解率を算出する．エポック毎の正解率の推移を図 3.3 に示す．

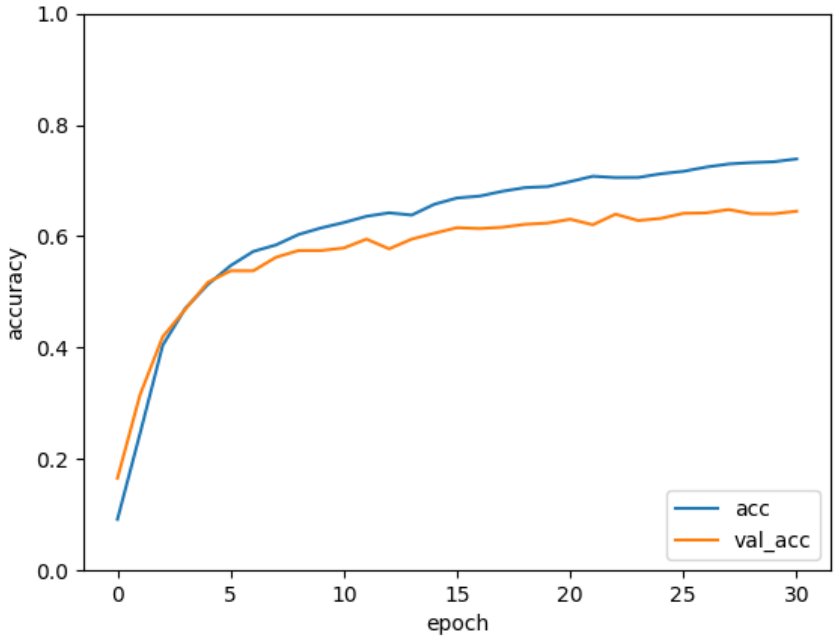


図 3.3　エポック毎の正解率の推移

訓練データに対する値をオレンジ，評価データに対する値を青で示している．なお，early stopping によって，エポック数 31 の時点で学習が打ち切られている．また，評価データに対する性能は，エポック数 28 の時点で，正解率が約 64.80%となっている．正解率約 64.80%という値は，既存研究に比べると改善の余地がある．例えば，Rafique ら[33]は，20 ファミリ 6,000 検

体を複数のアルゴリズムを用いて分類しており，いずれのアルゴリズムにおいても約 85%以上の正解率で分類している．データセットやファミリ数が異なるため，単純な比較はできないものの，正解率に劣る結果となっている．

3.3.4. 新種マルウェアの抽出可能性

本評価では，提案手法において，訓練データに含まれない未知の検体を新種のマルウェアであるとして検出できるか検証した．新種マルウェアは，既知マルウェアのいずれにも分類されないことから，各ファミリへ所属する確率が既知マルウェアに比べて小さいのではないかという仮説を立て，確率の違いによって新種マルウェアの抽出を試みた．本仮説を検証すべく，(評価 1)で構築したモデルに，既知マルウェアのデータセットとして同じく(評価 1)で用いた評価用のデータを，新種マルウェアのデータセットとして，FFRI Dataset 2016 に 1 検体だけ含まれる 181 種類の検体を投入し，各ファミリへ所属する確率を出力した．なお，同評価を実施する上でのノイズを抑制するため，既知マルウェアのデータセットとして用いた評価用データには，(評価 1)において正しく分類できたもののみを利用した．

上述の実験によって出力された値のうち，最も確率の高いファミリの確率を 10%単位で区切り，新種マルウェア・既知マルウェアそれぞれをプロットした分布を図 3.4 に示す．

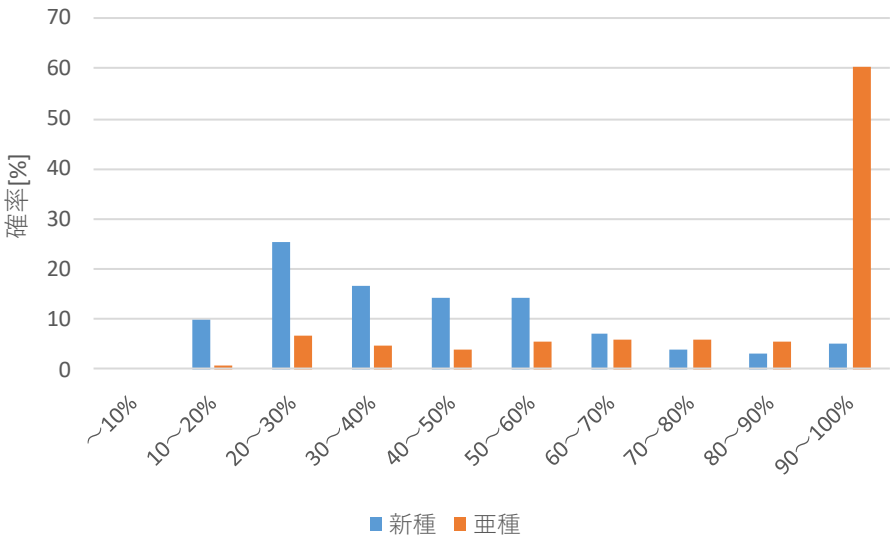


図 3.4　確率が最も高いファミリの確率分布

図 3.4 から分かるように，既知マルウェアの確率は，90～100%の範囲が最も高く，60～70%，70～80%の順に続く（それぞれ全体の 60.19%，6.04%，5.92%）．これに対し，新種マルウェアは 20～30%の範囲が最も高く，30～40%，40～50%の順に続く（それぞれ全体の 25.41%，16.57%，14.36%）．このように，新種マルウェアは比較的確率が低いのに対し，既知マルウェアの確率は

高いことが確認できた.

3.4. 考察

(ア) 解析環境

解析妨害機能の 1 つに，マルウェア解析が仮想環境で行われることを利用し，仮想環境では動作を行わないものがある．このような解析妨害機能をもつ検体は，仮想環境と物理環境での動作の差異を見ることによって，検出することが可能である．一方で，評価に用いたデータセットの解析環境は，OS は異なる（Windows 10，Windows 8.1）ものの，ともに仮想環境である．このため，上述のような解析妨害機能をもつ検体は，異なる動作を取らず，特徴量の 1 つとして用いた解析環境間での違いがあまり分類に寄与してないと考えられる. FFRI Dataset を用いた実験によって, 提案手法の利用可能性を検証できたため, 今後は, M3AS などを利用し，仮想・物理環境の混在を含む，より複数の環境における環境での解析結果を用いて，より詳細な評価を実施する.

(イ) ラベル付与方法

実験に用いたデータのラベルとして単一のアンチウイルスソフトでの検知名を利用した．また，多くの先行研究においてもラベル付けに同様のアプローチがとられている．一方で，単一のアンチウイルスソフトでの検知名を用いた場合，ベンダのバイアスが混入しうることや検知名が ground truth と一致しない場合があることが他の先行研究によって指摘されている[72][73][74]．これらの問題を緩和し，ground truth なラベルを付与するべく，複数のアンチウイルスソフトでの検出結果を組み合わせる手法が用いられている[75][76][77]．厳密な訓練・評価のために，上述の手法を取り入れ，訓練・評価データに対してより高精度なラベル付けを行うことが望ましいと考えられる.

(ウ) 最大動作レポートの抽出

呼び出し API 数が最も多いものを最も動作した解析レポートとして抽出し，学習・分類に利用した．一方で，解析環境を検知すると無意味な API 呼び出しを繰り返すような解析環境妨害機能も存在する．このような場合，呼び出し API 数は多くなり，最も動作した解析レポートとして抽出されるものの，解析のために本来の挙動を最も発現させた解析レポートを抽出したいという意図には沿っておらず，適切な分類ができない恐れがある．このような現象の防止策として，例えば，マルウェアらしい挙動に利用される API には大きい重みを，そうではない API には小さい重みを割り振り，前者をより重用して呼び出し API 数をカウントすること等が考えられる.

(エ) ネットワーク選定

時系列データを扱うものとして，Bidirectional LSTM を用いたが，LSTM を多段に利用することによって，より中長期的な特徴を掴む Stacked LSTM や CNN（Convolutional Neural Network）を活用し，複数のタスクで LSTM より高精度を出している QRNN（Quasi-Recurrent Neural Networks）[78]もあり，これらを取り入れることによって正解率に改善が見られる可能性もある．また，非時系列側のネットワークに関しても，ネットワーク構造や各種パラメータを決めうちで構築した面があり，これらを洗練していくことでより正解率を向上できる可能性がある．

(オ) 提案手法の運用

提案手法を運用する際，良性のプログラムは，既知マルウェアのいずれにも分類されず，新種マルウェア抽出の妨げとなる可能性がある．このため，良性プログラムとマルウェアを分別する手法と組み合わせ，マルウェアと判断されたもののみを提案手法に投入するといった運用が望ましいと考えられる．

(カ) 自動対処システムとの連携

提案手法を用いることにより，解析したマルウェアが既知のマルウェアなのか，新種のマルウェアなのか判断することが可能となる．提案手法により得られた，新種，亜種の情報を活用することにより，自動対処システムの精度向上が期待できる．具体的には，新種マルウェアだと判断された場合には，当該マルウェアの接続先の危険度が高いと判断する．また，頻繁に亜種が出現する場合には，今後も亜種が出現する可能性が高いため，脅威情報からは削除しないように運用することなどが考えられる．

3.5. まとめ

本章では，マルウェア解析の自動化・効率化を目的に，従来手法における解析環境ではマルウェアが動作しない場合があるという課題を緩和するために，多種解析環境におけるマルウェアの動的解析結果を用いた分類手法を提案した．本手法は，解析対象のマルウェアを類似する既存のマルウェアファミリに分類するものであり，これにより，亜種マルウェアの選別や既存のマルウェアファミリに分類されない新種のマルウェア抽出に応用できること，およびそれらを通した解析全体の効率化が期待できる．また，プロトタイプを実装し，評価実験によって，8,243 検体のマルウェア解析結果を 134 種類のファミリに分類できること，既知のマルウェアを約 64.80%の正解率で分類できることを確認した．また，各マルウェアファミリへ所属する確率を利用することによって，新種マルウェアを抽出でき得ることが確認できた．

4. 脅威情報の収集

4.1. 導入

本章では，固有表現認識（Named Entity Recognition: NER）を用いたセキュリティインテリジェンスの構造化手法を提案する．本手法は，製品名やマルウェア名のように，インテリジェンスとして着目すべき固有表現を抽出・構造化することにより，分析の効率化を狙う．この際，各語の条件付確率に着目することで，未知語の見逃し率を抑制する．

4.2. インテリジェンス構造化の課題

増加・高度化する攻撃に追従し，適切に対処するために，セキュリティ監視業務を担う SOC や CSIRT では，脆弱性情報，攻撃の早期警戒情報，および感染のインジケータ情報（Indicator of Compromise: IOC）等を収集する．SOC/CSIRT では，大きく以下のような手順でインテリジェンスを活用した業務が行われる（図 4.1）．

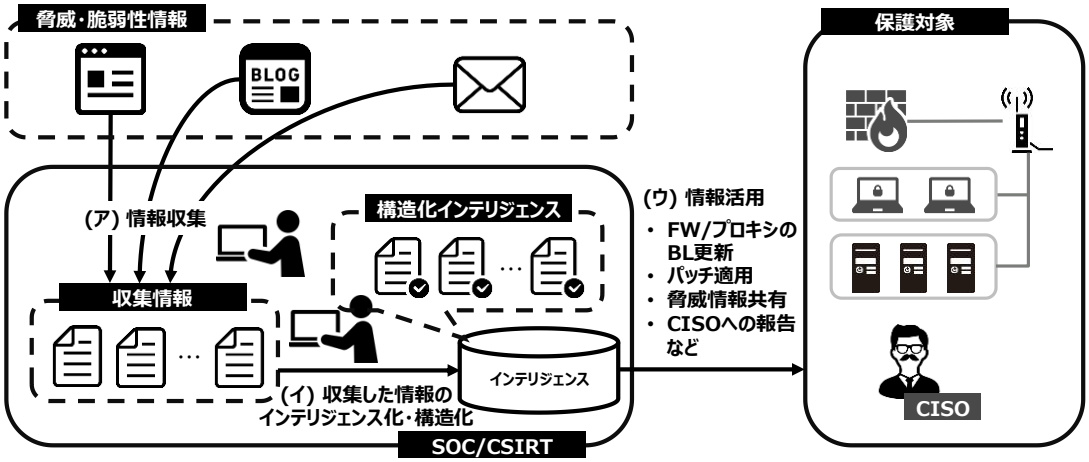


図 4.1　インテリジェンスに関する SOC/CSIRT 業務の全体像

(ア) 情報収集

サイバーセキュリティに関する情報を外部機関等から収集する．この情報には，例えば，悪性 URL，新規の脆弱性，および流行している攻撃の情報が含まれる．

(イ) 収集した情報のインテリジェンス化・構造化

収集した情報を分析し，セキュリティインテリジェンスとしてまとめるとともに，再利用・参照可能な形で保存する．

(ウ) 情報活用

(イ)で収集・構造化したインテリジェンスを各 SOC/CSIRT 業務に活用する．たとえば，悪性 URL リストを利用したファイアウォール・プロキシのブラックリスト更新や脆弱性情報を利用したパッチ適用，および攻撃トレンドの共有等が挙げられる．これにより，自組織の資産に関係する脆弱性への対応や流行している脅威への事前対応が可能となる．

従来の SOC/CSIRT では，これら一連の作業は，その多くが人手によるものを占めており，業務効率化のボトルネックとなっていた．特に，(ア)(イ)で挙げた情報収集とそのインテリジェンス化・構造化は，NVD において 75,000 を超える脆弱性が報告されている上，毎月 60,000 を超えるセキュリティブログ記事が公開されている[43]現状においては，多大なコストとなっている．

インテリジェンス構造化に関する課題は，以下の 2 つである．

(課題 1) インテリジェンス構築におけるコストが大きい

辞書やオントロジに関しては，初期作成コストに加え，継続的なメンテナンスコストが，機械学習に関しても教師データの用意コストが大きい．

(課題 2) 未知語を見落としてしまう可能性がある

辞書やオントロジに関しては，事前に定義していない未知語は認識することが出来ない．機械学習に関しては，教師データにない単語であっても，文脈情報等から未知語を認識することも可能である．しかし，未知語の認識精度は，教師データに出現する固有表現の認識精度に比べ大きく劣ることが示されている[79]．本課題は，未知語が頻出するセキュリティ分野では，その影響が顕著であると言える．

4.3. インテリジェンス構造化手法の提案

4.3.1. インテリジェンス構造化手法の要件

上で述べた課題を解決するための要件として，以下の 2 つが挙げられる．

(要件 1) NER の実施とその際のコスト抑制

インテリジェンスの構築・構造化を実現するために，既存研究においてその有用性が示されている NER を利用する．ただし，辞書・オントロジによるアプローチでは，それぞれの初期作成コストとメンテナンスコストが，教師ありな機械学習のアプローチでは，教師データの用意コストが大きいという課題がある（課題 1）．インテリジェンス分析コストの低減を狙う本研究において，その手段にあたる NER でもコストを抑制することが要件として挙げられる．

(要件 2) 未知語の見落とし率抑制

辞書・オントロジによるアプローチ，機械学習によるアプローチのいずれにおいても未知語の認識精度が比較的低いという課題がある（課題 2）．この課題は，特にセキュリティ分野での悪影響が大きく，未知語の見落とし率を抑制することが要件として挙げられる．

4.3.2. 設計方針

（要件 1）を満足するにあたり，NER を用いてインテリジェンス構築・構造化を実現し，SOC/CSIRT 業務におけるインテリジェンス構築を支援する．また，辞書ベースのアプローチでは，（要件 2）でも挙げている未知語への対応が難しいため，機械学習によるアプローチを採る．この際，機械学習によるアプローチにおける教師データの用意コストが大きいという課題は，半教師あり学習による固有表現認識モデルを採用することにより緩和する．半教師あり学習のモデルでは，教師データのうち一部のデータにのみアノテーションを実施した上で，アノテーションされていないものも含め全体を教師データとして利用する．これにより，全ての教師データにアノテーションが必要な教師ありモデルよりもアノテーションコストを抑制できる．具体的に利用するモデルについては，4.4 節で詳述する．また，インテリジェンスを構築・構造化するにあたり，NER でどういった固有表現を抽出すべきかを導出する必要がある．これについても 4.4 節で後述する．

（要件 2）の未知語の見落とし率抑制は，NER において，未知語が見落とされる際の条件付確率に着目し，未知語の可能性がある語を拾得する手法によって満足する．本手法は，未知語をはじめとして，本来であれば何らかの固有表現としてラベル付けされるべきであったにも関わらず，NER では見落とされたものを拾得するものである．見落とされた固有表現を抽出するために，各語の条件付確率を利用する．NER による固有表現の見落とし例を図 4.2 に示す．

コーパス：An Analysis of the Petya Ransomware Outbreak …

NER

認識結果

文字列	予測ラベル		正解ラベル
	第一候補（確率）	第二候補（確率）	
An	O (0.99)	Product (0.01)	O
analysis	O (0.99)	Product (0.01)	O
of	O (0.99)	Product (0.01)	O
the	O (0.99)	Product (0.01)	O
Petya	O (0.50)	B-Malware_Name (0.40)	B-Malware_Name
Ransomware	B-Malware_Type (0.95)	O (0.03)	B-Malware_Type

見逃し

図 4.2　NER による未知語の見逃し例

NER では，各語がどういった確率でどのラベルの固有表現かを算出する．この際，通常であれば最も確率の高い候補が同語のラベルとして採用される．つまり，見逃された語には，第一候補が「どの固有表現でもない」というラベルが付与されている．このとき，真に固有表現でないものは，「どの固有表現でもない」というラベルの条件付確率が比較的高く，固有表現たりうる語は，

同ラベルの条件付確率が比較的低くなる. また, 第二候補として真のラベルが挙がることが多い. これは，多くの場合コーパスは固有表現以外の語の割合が多く，過去のデータからその語の情報が直接得られない未知語や出現頻度の少ない語に対しては，固有表現で無いというラベルを付与することで正解率が高くなりやすいこと，および語の情報を除くとその文脈的な特徴から，正解のラベルが上位候補として現れやすいことに起因する．これらの現象を利用し，第一候補が「固有表現でない（O)」というラベルかつ第二候補の条件付確率が閾値よりも高いものを検出し，第二候補を固有表現の可能性があるとして，拾得する．

本手法の具体的な挙動を図 4.2 で示した従来の NER であれば見逃していた「Petya」を用いて説明する．図中では，第一候補が「O（固有表現ではない)」，第二候補が「Malware_Name（正解ラベル)」と続いている．第一候補が「固有表現ではない」こと，および他の真に固有表現ではない語の第二候補に比べ，その条件付確率が比較的高いことから，「Petya」は，固有表現の候補として拾得される．本手法により，固有表現の見落とし率を抑制する．

4.3.3. 抽出する固有表現

既存研究，STIX[80]，および実務者へのヒアリングを基に，インテリジェンスを構築すべき情報は何かという観点から, 表 4.1 に示す固有表現を導出した. なお, 表 4.1 で挙げた固有表現は，CVE 番号やハッシュ値のように，フォーマットが定まっており，正規表現で抽出できるものとそうでないものに大別できる．提案手法では，正規表現で抽出できるものは正規表現で抽出し，そうでないものを NER で抽出する．

表 4.1　抽出する固有表現の一覧

固有表現	意味	正規表現抽出
Alert_ID	アラートの識別子	○
Attack_Method	攻撃手法	×
Attacker	攻撃者・攻撃グループ名	×
Campaign_Name	攻撃のキャンペーン名	×
CVE_ID	CVE 番号	○
File_Hash	ハッシュ値	○
File_Name	ファイル名	○
Ip_v4/Ip_v6	ipv4/ipv6 アドレス	○
Mail_Address	メールアドレス	○
Malware_Name	マルウェア名	×
Malware_Type	マルウェアの種類	×
Mitigation_Word	Mitigation のための語	×
MS_ID	MS 更新プログラムの ID	○
Organization_Name	組織の名称	×
Port	ポート番号	○
Product	製品，システム	×
Reference	URL，FQDN	○
Region	国・地域名	×
Segment	業界区分	×
Site_Type	悪性 Web サイトの性質	×
Time	時間・時期	○
Version	バージョン名	○
Vulnerability	脆弱性	×

4.3.4. インテリジェンス構造化手法の全体像

これまで述べてきた設計方針に基づいたインテリジェンス構造化手法の全体像を図 4.3に示す.

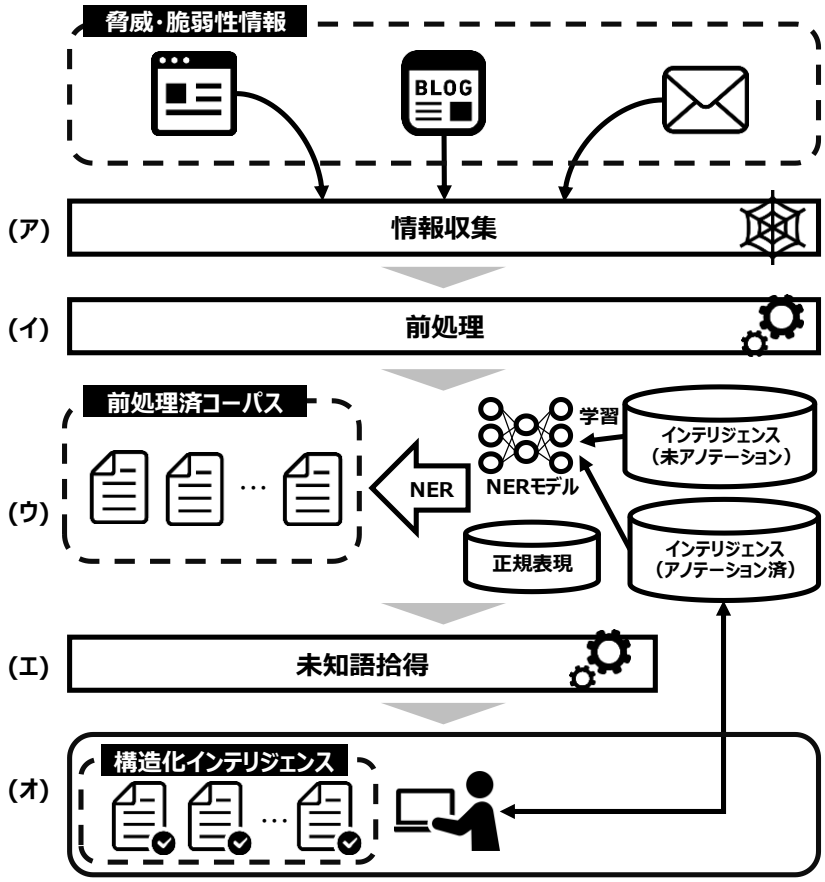


図 4.3　インテリジェンス構造化手法の全体像

(ア) 情報収集

外部機関が公開している脅威情報や脆弱性情報を収集する．

(イ) 前処理

収集した情報に対し，ノイズの除去や NER で利用できる形への変換といった前処理を実施する．

(ウ) 固有表現認識

収集した情報から固有表現を抽出する．この際，前述したとおり，正規表現で抽出できるものは正規表現で，そうでないものは NER で抽出する．

また，半教師ありの NER モデルを構築するための教師データのうち，アノテートされた教師データは，構造化された（アノテートされた）インテリジェンスと同等である．そこで，人手でアノテートした教師データに加え，インテリジェンス構造化手法によって構造化されたインテリジェンスも同様に教師データとして利用する．

(エ) 未知語拾得

(ウ)において，固有表現として認識されなかったものから，未知語の可能性があるものを抽出する．

(オ) 分析者への提示

ここまでの処理フローで構築されたインテリジェンスを SOC/CSIRT の分析者に提示し，分析に活用する．

4.4. 実現方式

4.4.1. 情報収集

インテリジェンス構造化手法で扱う情報としては，US-CERT[81]，ICS-CERT[82]をはじめとする公的組織や各種セキュリティブログ[83]によって Web サイトで公開されているものを想定している．日々の SOC/CSIRT 業務における情報収集では，前回収集時から更新された新規の情報のみを取得する．RSS で配信されている各 Web ページの更新情報を利用して新規の情報を取得した．

4.4.2. 前処理

インテリジェンス構造化手法で扱う Web ページでの公開情報には，本文以外のヘッダ・フッダ等に加え，本文の中にも html タグ等のノイズが含まれる．また，ストップワード等，後段の NER においてノイズとなる情報が含まれている．そこで，それらのノイズを除去するために，下記の前処理を実施する．

(ア) 本文抽出

Web ページの内容をテキスト化し，ヘッダとフッダを取り除くことにより，インテリジェンスとなり得る本文部分を抽出するとともに，NER でノイズとなる html タグ等を除去する．

(イ) 文/単語分割

文ごとの系列ラベリングにより固有表現を抽出する．また，単語単位でラベル付けを行う．そこで，(ア)で抽出した本文を文ごとに分割した上で，それぞれ単語ごとに分割する．

(ウ) 正規化

表記ゆれや不要な情報による NER への悪影響を抑制するため，正規化を行う．全ての文字を小文字に統一した上で，語の語幹を取り出すステミングと見出し語へ変換するレンマ化を行った．また，URL は，ある程度自由に設定できるものの，その表記の多様性自体はラベル付与に寄与しない．そこで，全 URL に対し，URL を意味するタグへの置換を行った．

(エ) ストップワード除去

ストップワードとは，冠詞や前置詞のように，多くの文に高頻度で出現するために，ラベル付与に寄与しない語のことである．提案手法では，Natural Language Toolkit[84]で定義されている英語のストップワード 179 個を除去した．

4.4.3. 固有表現認識

ここまでの処理で自然言語処理に適した形となったコーパスに対し，NER を実施する．正規表現で抽出可能な CVE 番号や各種ハッシュ値等を正規表現で抽出した後，それ以外の固有表現を半教師ありの NER モデルによって抽出する．具体的には，Ma ら[85]が提案している文字の Convolutional Neural Network（CNN）による表現と単語の埋め込み表現を結合したベクトルを入力とし，Bi-directional Long Short-Term Memory（Bi-LSTM）層，結合層，および Conditional Random Field（CRF）層によって NER を行うモデルを用いる（図 4.4）．本モデルは，NER タスクにおいて state-of-the-art を達成しており，同等のモデルは多くの研究で用いられている[86]ことから，インテリジェンス構造化手法においても採用した．この際，文字表現と単語埋め込み表現をアノテーションされていないコーパスも用いて学習することで，アノテーションコストを抑制しつつ，固有表現の認識精度向上を図る．

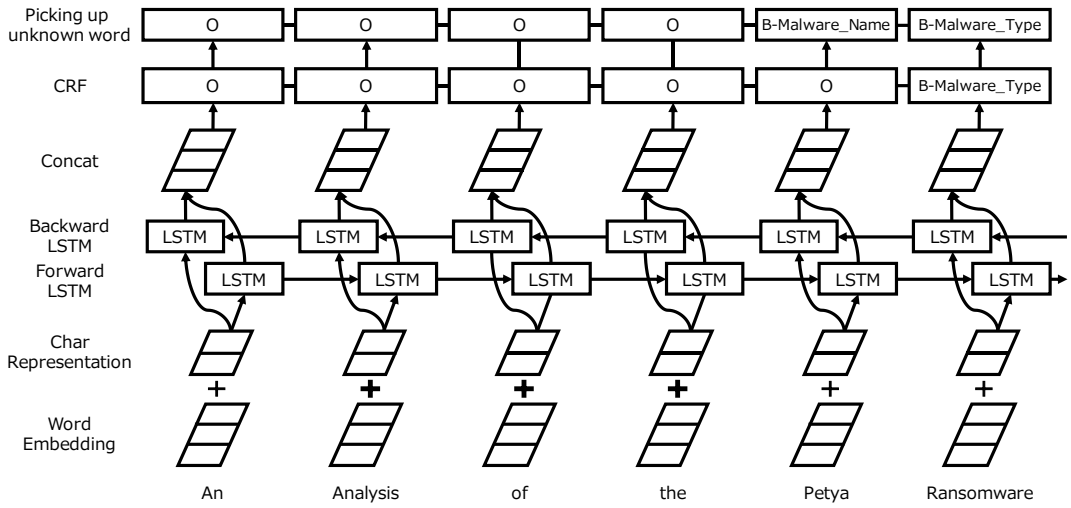


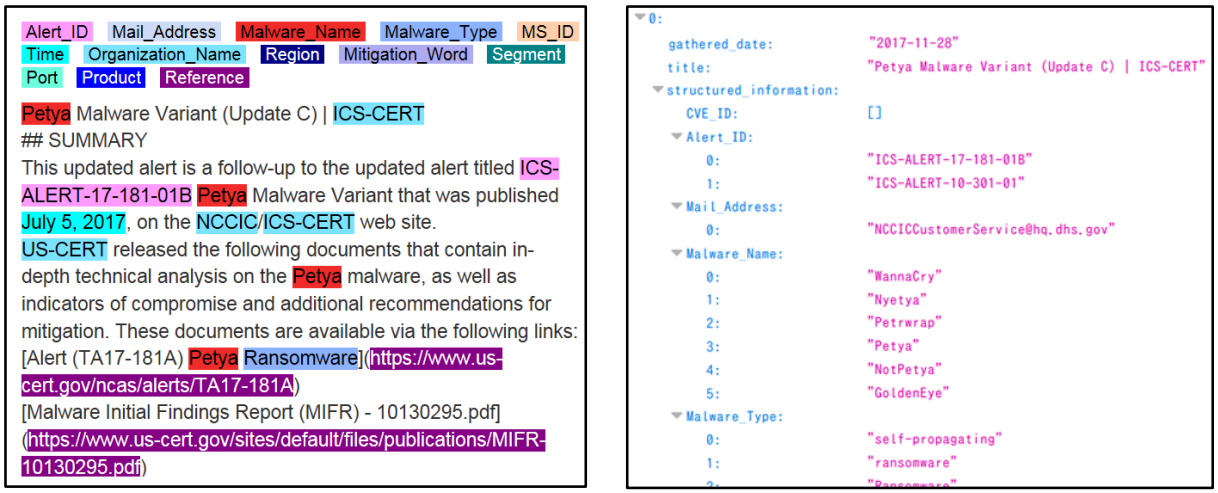
図 4.4　NER モデルの全体像[Ma ら[85]]

4.4.4. 未知語拾得

未知語を拾得するために，上で述べたとおり，固有表現認識結果における条件付確率を参照し，第一候補のラベルが O かつ第二候補の条件付確率が閾値以上のものを拾得する．提案手法においては，CRF 層の結果を受け取り，未知語か否かを判定する層を NER モデルの終端に追加する形で実現した（図 4.4）．

4.4.5. 分析者への提示

分析者は最終的にタグ付けされた文章や構造化された情報を確認する．本手法によって，ICS-CERT による Petya の Alert[87]に(A)タグ付けと(B)構造化を実施した画面の一部を図 4.5 に示す．(A)では，そのタグから ICS-CERT（Organization_Name）によって発行されたランサムウェア（Malware_Type）の Petya（Malware_Name）に関するレポートであることが推察できる．このように，タグ付けされた文章により分析を効率化することが期待できる．また，構造化されたインテリジェンスを機械処理等に活用することが期待される．



(A)タグ付けされた文章　　(B)構造化されたインテリジェンス

図 4.5　提案手法の出力

4.5. 評価

4.5.1. 評価項目と評価環境

評価項目は，以下の 3 つである．

(ア) 固有表現の認識精度

インテリジェンス構造化手法は，Ma ら[85]の手法をベースラインとして NER を実施し，それに加えて各語の条件付確率を用いて未知語を拾得と認識精度の向上を図るものである．そこで，提案手法がどの程度固有表現を認識できるかの精度を適合率，再現率，および F 値によって検証する．この際，ベースライン手法とインテリジェンス構造化手法を比較する．

(イ) 見落としの抑制度

提案手法は，第二候補の条件付確率を用いることにより，従来の NER では見落としていた固有表現の抽出を図る．そこで，本手法によって，従来の NER であれば見落としていた固有

表現をどの程度抽出できるか検証する．

(ウ) 処理時間

本手法は，日々の分析業務での活用を想定しているため，処理時間がその範囲内に収まっている必要がある．なお，本手法の処理としては，情報更新後のモデルの再学習とそのモデルを利用した記事に対する NER がある．そこで，本手法におけるモデルの再学習時間と 1 記事に対する NER の時間を測定し，実用に耐えうるか検証した．

なお，評価用のデータセットとしては，インテリジェンスのひとつである ICS-CERT の Alert に対して，著者のグループが IOB2 によりアノテーションしたコーパス（123 記事，4,779 文）を利用した．また，このうち 2011 年 7 月 13 日から 2013 年 10 月 29 日までの 83 記事（全体の約 70%）を訓練用，以降の 40 記事（約 30%）を評価用のデータとして利用した．また，未アノテーションな教師データとしては US-CERT の Alert179 記事と ICS-CERT の Adversary950 記事の合計 1,129 記事を使用した．ハイパパラメータは，ランダムサーチで探索し，文字表現の次元数を 25，文字埋め込み表現の次元数を 100，LSTM の次元数を 100 に設定した．また，第二候補から固有表現候補を抽出するか否かの閾値として 0.20 を用いた．なお，評価環境は，表 4.2 に示す通りである．

表 4.2　評価環境

項目	スペック
CPU	Intel Core i7-7700K 4.20GHz
GPU	GeForce GTX 1080
メモリ	64GB
OS	Ubuntu 16.04.3 LTS

4.5.2. 評価結果

(ア) 固有表現の認識精度

前述のデータセットに対するベースライン手法，提案手法それぞれの固有表現の認識精度を表 4.3 に示す．

表 4.3　固有表現の認識精度

	precision	recall	F-measure
ベースライン	0.75	0.74	0.73
提案手法	0.77	0.80	0.78

いずれの指標においても，提案手法がベースラインの手法を上回っていることから，その差分である未知語拾得の部分が認識精度向上に貢献したといえる．

また，データセット中には現れなかった「Attacker」と「Campaign_Name」を除く 10 の固有表現について，提案手法による固有表現ごとの認識精度を表 4.4 に，混同行列を図 4.6 に示す．

表 4.4　固有表現ごとの認識精度

	precision	recall	F-measure
Attack_Method	0.30	0.26	0.28
Malware_Name	1.00	0.57	0.72
Malware_Type	1.00	0.88	0.93
Mitigation_Word	0.71	0.85	0.78
Organization_Name	0.92	0.92	0.92
Product	0.67	0.66	0.66
Region	0.67	0.53	0.59
Segment	0.70	0.83	0.76
Site_Type	1.00	1.00	1.00
Vulnerability	0.63	0.62	0.62

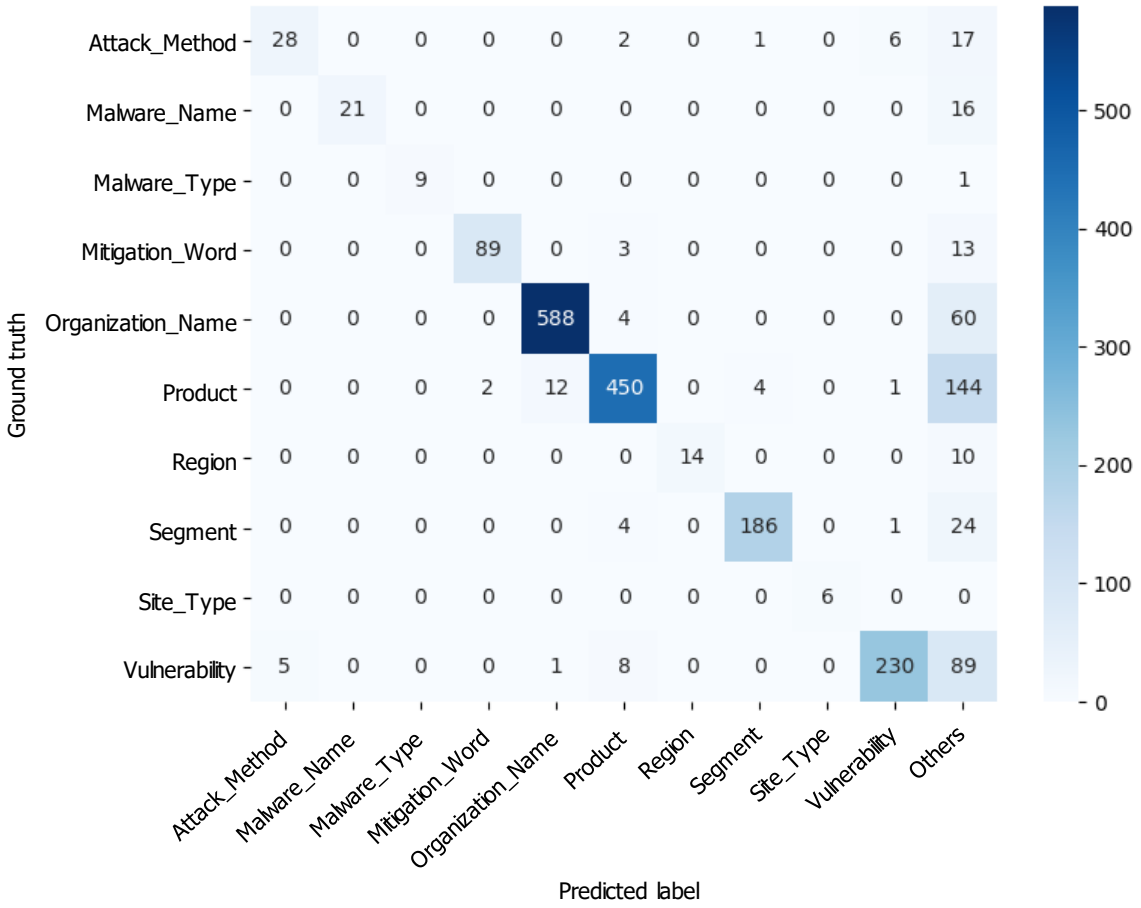


図 4.6　固有表現ごとの混同行列

表 4.4 と図 4.6 から，「Attack_Method」，「Malware_Name」，および「Region」が他の固有表現に比べて見逃しが多く，低精度となっていることが見て取れる．これは，アノテーション済み教師データにおけるこれらの固有表現の出現数が他に比べて少ないことが原因の一つであると推察される．本事象に関しては，今後より大規模なデータセットを用いて評価し，改善が見られるか検証する．

(イ) 見落としの抑制度

上で述べたとおり，第二候補から固有表現候補を抽出するか否かの閾値として 0.20 を用いた．つまり，第一候補のラベルが「いずれの固有表現でもない」かつ第二候補の条件付確率が 0.20 以上の場合に，見落された固有表現として第二候補を抽出する．評価したところ，上述の条件に当てはまる 108 の単語が新たに固有表現として抽出され，そのうち 95 個が真に固有表現である単語だった．なお，データセット中に，第二候補が正解の固有表現は 203 個含まれていた．すなわち，偽陽性を約 12%（13/108）に抑えつつ，従来手法では見落とされてい

た固有表現のうち，第二候補が正解であったものを約 47%（98/203）抽出できた．この結果が表 4.3 に示した性能向上に寄与している．以上のように，提案手法によって，固有表現の見落としを抑制可能なことが確認できた．

(ウ) 処理時間

1,000 エポック実施した際の学習曲線を図 4.7 に示す．この際，1 エポックに 54 秒，合計で約 15 時間を要した．

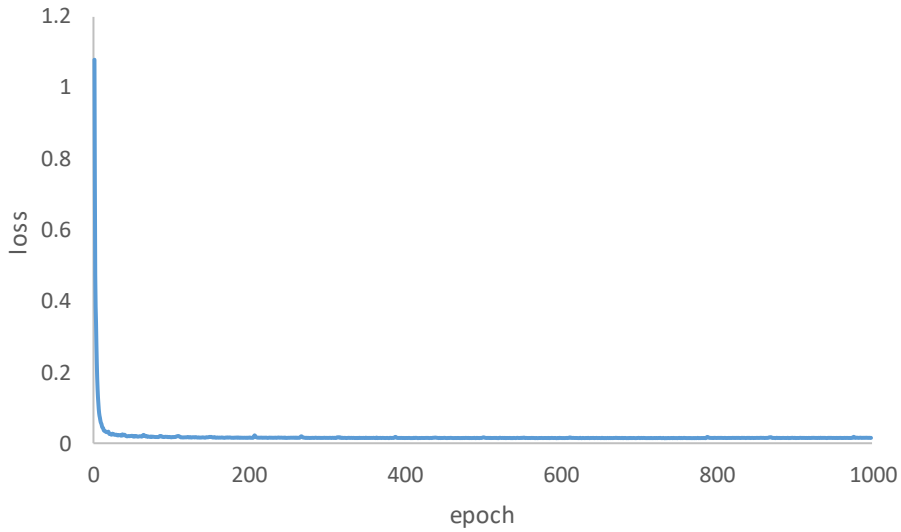


図 4.7　学習曲線

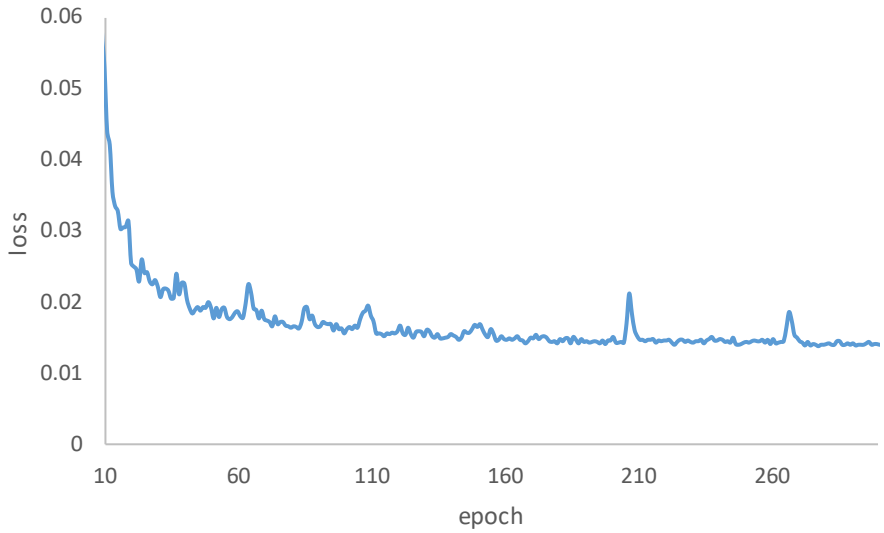


図 4.8　学習曲線（10～300 エポック）

図 4.8 に，10〜300 エポック時の学習曲線を示す．図 4.8 から，120 エポック付近で学習がほぼ収束していることが分かる．このため，Early Stopping を利用し，この時点で学習を打ち切ることにより，同等の精度を確保しつつ，1.8 時間程度でモデル再学習を実施できる．

また，評価用の 40 記事に対する NER には，約 6.37 秒を要した．このことから，1 記事平均は，約 0.16 秒である．仮に，1 日に 2,000 記事のセキュリティブログ（毎月 60,000 件を超えるセキュリティブログが発信されている[43]状況から，1 月を 30 日として試算）を処理するとしても，約 318.5 秒（約 5.3 分）で処理可能である．

以上より，モデル再学習に約 1.8 時間，NER に 1 記事平均約 0.16 秒，想定される 1 日の記事量 2,000 に対しても約 5.3 分と，実用の範囲内であると言える．

4.6. 考察

(ア) 正規表現による固有表現の抽出

フォーマットが定まっている固有表現は，正規表現により抽出する．正規表現による抽出は，ほぼ 100%の精度だったものの，幾つかの誤認識が見られた．例えば，1.1.1.1 のようなソフトウェアのバージョンが IPv4 アドレスとして抽出される現象が見られた．誤認識数は多くなかったため，個別に除外ルールを作成する，人手で修正する等の手法により，対応可能であると推察される．また，URL を抽出した際，IOC として記載されている悪性 URL なのかベンダ等が公開している Web ページへのリンクであるかを正規表現では判定できない．これについては，ホワイトリストや悪性 URL 判定手法の導入が必要である．

(イ) 見落とした固有表現を抽出する際の閾値

閾値に 0.20 を用いて評価を行い，NER の性能向上に寄与することを示した．本閾値は，値を増加することにより，固有表現として拾得するか否かの判定が厳しくなるため，recall が低下する代わりに，precision が向上する．実際に本手法を運用する際には，どちらの値を重視するかによって閾値を調整するのが望ましいと推察される．

(ウ) 処理時間

提案手法におけるモデル再学習や NER の処理時間は，全体の記事量や記事ごとの内容量に依存する．データセットに対しては，実用の範囲内であることを示したが，運用の中でデータ数は単純増加していくと思われるため，その中でも実用に耐えうるか，より大規模なデータセットを用いて今後検証していく．

4.7. まとめ

本章では，未知語を考慮した NER によるインテリジェンス構造化手法について述べた．提案手法は，ベースラインとして，文字の CNN による表現と単語の埋め込み表現を結合したベクトルを入力とし，Bi-LSTM 層，結合層，および CRF 層によって NER を行うモデルを用いることにより，インテリジェンスとして着目すべき固有表現を抽出する．この際，CRF 層で算出した各語の条件付確率に着目することにより，セキュリティ分野では特に頻出な未知語に対する見逃し率を抑制し，認識精度の向上を図る．これらの手法により，インテリジェンスの構築・構造化を支援し，分析の効率化を狙う．

評価では，提案手法のプロトタイプを実装し，ベースラインの手法よりも高精度で固有表現を認識できることと従来手法では見逃していた未知語を認識できていることを確認した．さらに，モデルの再学習時間や 1 記事に対する NER に要する時間を計測し，実運用にも耐えうることを示した．

5. 脅威度の推定

5.1. 導入

悪性サイトへの通信を抑制する方法に，インテリジェンス（ブラックリスト等）を用いるものがある．しかし，インテリジェンスには偽陽性が含まれている場合があり，仮に業務遂行に必要な非悪性サイトがブラックリストに誤って含まれていると，当該サイトにアクセスできず，業務阻害の要因となってしまう．業務阻害を抑制する方法として，インテリジェンスを事前に精査し，非悪性サイトを人手で除外する方法も考えられるが，Web サイトの精査という別のコストが生じてしまう．そこで，本章では機械学習を用いた Web サイト脅威度の推定手法を提案する．

5.2. Web サイト脅威度推定システムの提案

Web サイト脅威度推定システムは以下の 2 つの機構から構成される．

(ア) 外部情報取得機構

本機構は，後段の悪性度推定に必要な情報を取得する．具体的には，推定対象 Web サイトの URL をベースとして外部のサーバに問い合わせを行い，WHOIS 情報，DNS 情報，Alexa ランク，および地理情報を取得する．

また，一度取得した情報は，データベース（以降，DB）に保管することにより，以降外部アクセスなしに参照する．このため，同じドメインに対する 2 回目以降のアクセスでは，DB を参照して外部情報を取得することにより，外部アクセスを伴うことによる処理時間の長大化や同一情報提供元に対するアクセス過多を抑制できる．

(イ) 悪性度推定機構

本機構は，機械学習によってアクセス先の悪性度を推定する．ブラックリストを悪性サイトの，ホワイトリストを良性サイトの教師データとして学習し，推定モデルを構築する．特徴量としては URL 文字列から取得できる情報に加えて，外部情報取得機構を用いて取得した情報を利用する．なお，各特徴量は，良性サイトと悪性サイトの間に違いとして現れやすい点に着目して選出しており，各値は，0〜1 の間に正規化して利用する．具体的には，最小値を 0，最大値を 1 とし，その間を比例配分する．また，後述の URL 文字列のように上限の無い特徴量に関しては，特徴量化時点での最大値が 1 になるように正規化を行う．特徴量一覧については，値域と共に表 5.1 に示し，以降で詳述する．

表 5.1　各推定器における特徴量と値域

推定器	カテゴリ	通番	特徴量	値
1	URL 文字列	1	URL 文字列長	0-
		2	ドメイン文字列長	0-
		3	パス文字列長	0-
		4	URL 文字列に含まれる数字の数	0-
		5	ドメイン文字列に含まれる数字の数	0-
		6	パス文字列に含まれる数字の数	0-
		7	パス文字列に含まれるトークン数	0-
		8	パス文字列に含まれる平均トークンス数	0-
		9	ドメイン文字列に全体に対する最長英単語の長さの割合	0-1
		10	FQDN のジニ係数	0-1
		11	FQDN が”.”を含むか	0/1
		12	FQDN が”-“を含むか	0/1
		13	URL が拡張子で終わるか否か	0/1
		14	URL が拡張子で終わるか否か	0/1
		15	URL 文字列が”exe”を含むか否か	0/1
		16	URL 文字列が”php”を含むか否か	0/1
		17	ドメインが IP アドレスか否か	0/1
2	WHOIS 情報	18	ドメイン登録期間（年数）	0-10
		19	ドメイン性質（初登録 or 更新有）	0/1
		20	ドメイン初登録からの年数	0-
		21	ドメイン登録時間（0-23）	0/1
		22	ドメイン登録曜日（月火水木金土日）	0/1
		23	レジストラ（該当レジストラのうち，悪性データの割合）	0-1
3	DNS 情報	24	A レコード数	0-
		25	AAAA レコード数	0-
		26	CNAME レコード数	0-
		27	MX レコード数	0-
		28	NS レコード数	0-
		29	PTR レコード数	0-
		30	TXT レコード数	0-
		31	逆引きが設定されているか否か	0/1
		32	ネガティブ TTL（SOA レコードの minimum 値）	0-
	Alexa ランク	33	Alexa ランク（訪問者数ランク）	0-
		34	Alexa ランクの差分(アクセス数ランク-訪問者数ランク)	0-
	地理情報	35	国 (IP アドレスに対応する国のうち，悪性データの割合)	0-1

URL 文字列では，悪性 URL は，良性 URL よりも，複雑である[50][47][48]（通番 1～11），特定の文字・拡張子の出現頻度が高い[50][48][49]（通番 12～16），およびドメインと紐付けられていない[47][48]（通番 17）という傾向を掴むために特徴量を選定した．なお，通番 11 のジニ係数とは，集合の複雑性を測る指標の 1 つであり，集合の複雑性が低いほど 0 に，高いほど 1 に近づく．このため，FQDN をアルファベットや記号 1 文字ごとの集合とみなし，ジニ係数を算出することにより，その複雑性を測ることができる．

WHOIS 情報では，攻撃に利用されるドメインは使い捨てであり，生存期間が短い場合がある[51][88][89][52]（通番 18～20），攻撃者の登録作業コストや登録の金額面でのコストを抑制するために，まとめて登録される場合がある[3][8] （通番 21，22），および，攻撃者がドメインを取得する際のレジストラには偏りがある[51][88][90][53]という傾向を掴むために，特徴量（通番 23）を選定した．

DNS 情報では，各種レコード（通番 24～31）に加え，悪性サイトのネガティブ TTL は短い傾向にある[91][54]という特徴を掴むために，特徴量（通番 32）を選定した．

また，悪性サイトは，アクセス数が正規サイトより少ないという仮定の下，アクセス数ランキングである Alexa ランク（通番 33，34）を利用した．さらに．攻撃に利用されるドメインは特定の国に偏っているとの報告[49]から，通番 35 を選出した．

アンサンブル学習によって複数の推定器を組み合わせることによって，汎化能力が向上することが知られているため，ここまでに述べてきた特徴量をまとめて単一の推定器に利用するのではなく，3 つの推定器を用意し，推定器 1 で URL 文字列，推定器 2 で WHOIS 情報，および推定器 3 でその他の情報（DNS 情報，地理情報，および Alexa ランク）を利用して，各推定器の加重平均を取ることによりアクセス先サイトの脅威度を判定する．各推定器に与える重みには，学習時に教師データの分類精度を記録しておき，その分類精度の推定器間での比率を正規化した値を利用する．例えば，同様の教師データを推定器 1 が 90%，推定器 2 が 85%，推定器 3 が 75%の精度で分類できた場合，その比率は，90:85:75 であり，この比率の合計が 1 になるよう正規化する．つまり，それぞれの推定器に対して，0.36，0.34，0.30 の重みを与える．

また，本機構は，対象 Web サイトの脅威度を 0（良性寄り）～100（悪性寄り）の連続値で算出する．この際，対象 Web サイトが悪性か否かを分類する閾値を定めておき，脅威度が閾値未満の場合は良性，閾値以上の場合は悪性と判断する．

5.3. 評価実験

5.3.1. 実装

提案手法は，上述した通り，機械学習によってサイトの脅威度を推定するが，この部分の実装には機械学習ライブラリである scikit-learn [92]を利用した．また，WHOIS，DNS，Alexa ラン

ク，および地理情報といった各種外部情報の取得には，それぞれ Python-whois[93]，dnspython[94]，Alexa API[95]，および GeoIP[96]を利用した．

5.3.2. データセット

良性サイトのサンプルとして，オープン Web ディレクトリである DMOZ[97]から収集したサイト群, 悪性サイトのサンプルとして hpHosts[98], spamhaus[99], Malware Domain List[100], および aguse[101]から収集したサイト群を利用した．この際，良性/悪性サイト情報それぞれ 50,000 件ずつ，合計 100,000 件をランダムに取得した．これらの情報を以降の評価に利用する．

5.3.3. 評価項目

評価項目は，以下の通りである．

(ア) 脅威度推定精度

Web サイト脅威度推定機能は，複数の推定器を利用して，アクセス先サイトの脅威度を推定する．良性データ/悪性データを用いてこの推定精度を検証する．

(イ) 処理性能

脅威度推定にかかる時間が実用範囲内か検証する．なお，処理性能は，表 5.2 に示す環境で測定した．

表 5.2　評価環境

項目	スペック
CPU	IIntel Core i7-2600 （4 コア，8 スレッド）
メモリ	2,048MB
OS	Ubuntu 14.04 LTS

5.3.4. 評価結果

(ア) 脅威度推定精度

本評価では，5.3.2 項で収集した良性サンプル 50,000 件と悪性サンプル 50,000 件を用いて，提案手法の脅威度推定精度を検証する．まず，URL 文字列のみを用いた場合，WHOIS 情報のみを用いた場合，および DNS 情報+Alexa ランク+地理情報を用いた場合の 3 パターンにおいて，各種アルゴリズム（線形 SVM，ロジスティック回帰，決定木，K 近傍法，ランダムフォレスト，3 層ニューラルネット，およびアダブースト）を用いて評価を実施した．その後，全情報をまとめて 1 つの推定器に利用した場合と各パターンで最も高精度であったアルゴリズムのものを組み合わせた場合の提案手法で比較評価を行った．各アルゴリズムでは，対象 Web サイトの脅威度を 0（良性寄り）～100（悪性寄り）の連続値で算出するが，この際，悪性サイト/良性サイトの分類閾値には，50 を用いた（脅威度が 50 未満の場合：良性と判

断，脅威度が 50 以上の場合：悪性と判断）．なお，提案手法において，各推定器の重みには，同データセットに対する各推定器の正解率の比率を利用した．さらに，提案手法に関しては，閾値をどの値に設定すれば良いか，FP，FN，および正解率の観点から評価した．なお，これらの評価には，10-分割交差検証を用いた．

各パターンにおける脅威度推定精度の測定結果を表 5.3 に示す．

表 5.3　各パターンにおける脅威度推定精度の測定結果（%）

	URL	WHOIS	DNS	すべて利用	提案手法
線形 SVM	66.11	81.72	74.32	86.72	-
ロジスティック回帰	66.11	81.73	74.38	86.76	-
決定木	68.81	82.06	84.30	88.67	-
K 近傍法	67.75	80.75	88.79	83.80	-
ランダムフォレスト	68.92	83.07	85.53	91.26	-
ニューラルネット	67.07	82.27	80.00	88.72	-
アダブースト	67.17	81.98	81.17	89.64	-
最高精度	68.92	83.07	88.79	91.26	92.74

表 5.3 から，URL 文字列のみ，WHOIS 情報のみを用いた場合はランダムフォレスト，DNS 情報+Alexa ランク+地理情報を用いた場合は K 近傍法がそれぞれ 68.92%，83.07%，および 88.79%と最高精度であることが確認できる．さらに，上述のアルゴリズムを用いた 3 つの推定器を組み合わせて利用したところ，各推定器単体のいずれ（68.92%，83.07%，および 88.79%）よりも高く，かつ全ての情報を単純にまとめて利用した場合（91.26%）よりも高い 92.74%の精度で Web サイトを分類できた．また，図 5.1 は，各パターンでの最高精度を出した分類器の Receiver Operating Characteristic（以降，ROC）曲線である．ROC 曲線において，曲線下の面積を Area Under the Curve （以降，AUC）と呼び，AUC が 1 に近いほど，識別性能が高いことを示す．

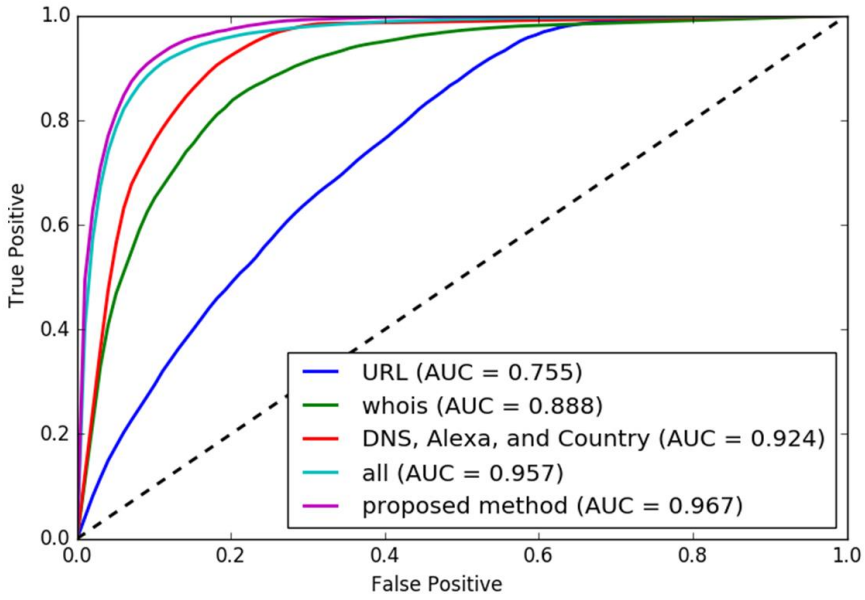


図 5.1　各パターンにおける ROC 曲線

図 5.1 から，提案手法の AUC が 0.976 と正解率と同様に他のいずれのパターンにおける値（0.754，0.903，0.949，および 0.966）よりも高いことが分かる．

以上のように，提案手法が各推定器単体の場合や全情報をまとめて 1 つの推定器で利用した場合のいずれよりも高精度で Web サイトの脅威を推定できた．また，複数の研究において，URL 文字列のみで推定することの問題点が示唆されている（精度の確保が難しい[47]，短縮 URL を誤判定してしまう[48]等）が，本実験においても，URL 文字列のみを使ったものは，最高でも 68.92%と他に比べて精度が低く，各研究での示唆内容を裏付けるものとなった．

次に，提案手法において，閾値を 0～100 の間で変動させ，それぞれの値における FP，FN，および正解率を算出した．この結果を図 5.2 に示す．

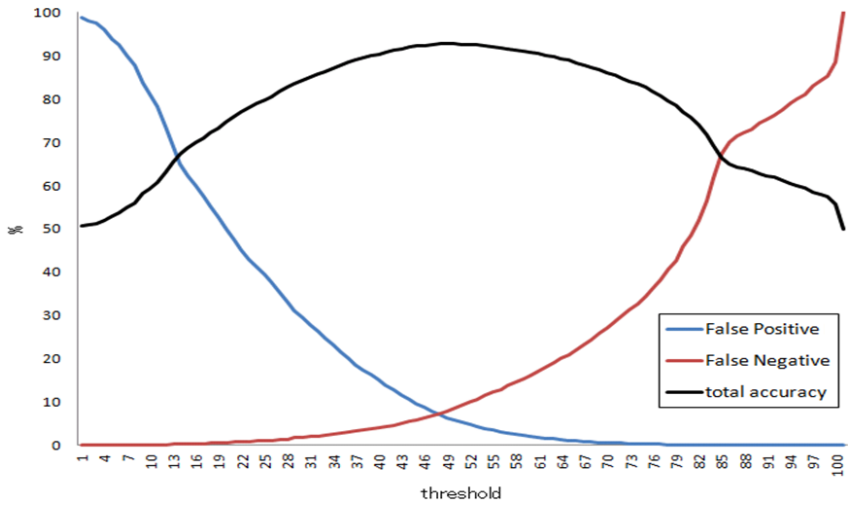


図 5.2　閾値毎の FP，　FN，および正解率（%）

図 5.2 から分かるように，閾値を 48 に設定することで，92.82%と最も高い正解率を得られている．また，閾値を下げると FP が高まり，反対に閾値を上げると FN が高まる．悪性サイトを見逃してしまうと，同サイトへのアクセスが発生して被害が生じうることや良性サイトを不審サイトとして誤検知してしまった場合でも，後述する自動対処と組み合わせれば業務効率への影響は最小限に抑制できることから，FP の増大はある程度許容できるとともに，FN を抑制することが望ましい．最高精度を出せる閾値 48 の場合は FN が 7.36%だが，これを半分以下に抑えようとした場合，閾値を 37 にすることにより，正解率を 89.96%に留めつつ達成できる（3.518%）．

以上の評価結果から，複数の推定器を用いることによって，単体の場合よりも精度が出せること，正解率の視点からは閾値を 48 にすれば良いことが分かった．

(イ) 処理性能

本評価では，提案システムにおいて，情報取得と脅威度判定に要する時間の合計を測定した．なお，測定は，マルチユーザモード下において time コマンドを用いて行った．

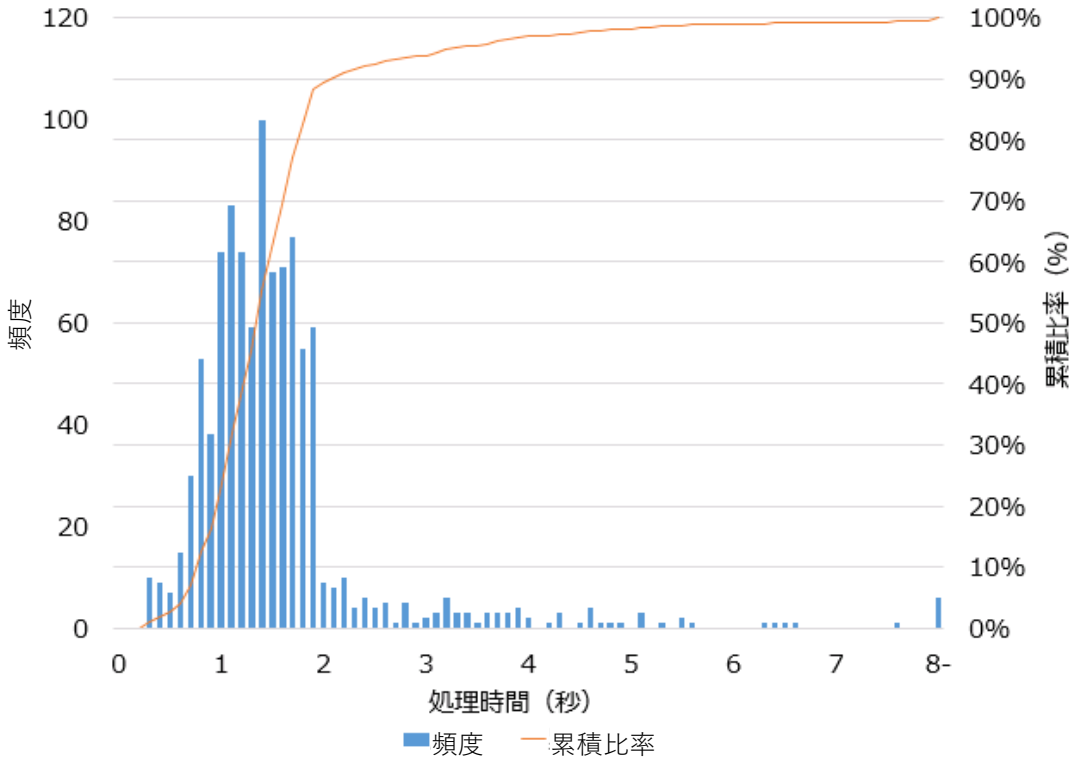


図 5.3　処理時間毎の頻度と累積比率

測定結果の処理時間毎の頻度と累積比率を図 5.3 に示す．図 5.3 から，93.8%の処理時間が 3 秒以内に収まっていることが分かる．多くのユーザがページのロードが 3 秒以内であることを期待しているという調査結果[102]があり，提案システムの処理時間は 93.8%がその範囲内に収まっていることから，処理性能の面でも実用範囲内であると考える．

5.4. 考察

提案手法は，悪性サイトにアクセスすることなく得られる特徴量を用いて脅威度を推定する．2 章で述べたように，悪性サイトにアクセスすることにより得られる情報もある．これらの情報も用いた脅威度推定を行うことでより精度が向上すると考えられる．

5.5. まとめ

本章では，Web サイト脅威度推定機能の検討，開発，および評価を行った．評価によって，サイトの性質を最大 92.74%の精度で推定できることを確認した．

6. 脅威情報を用いた自動対処

6.1. 導入

本章では，脅威情報を活用した自動対処を実現するにあたっての課題を分析し，自動対処システムに求められる要件を導出する. さらに, 導出した要件に基づき自動対処システムを提案する. また，提案した自動対処システムを用いた評価実験を行い，提案システムの有効性を評価する.

6.2. 自動対処システムの提案

6.2.1. 自動対処における課題

マルウェアを動的解析して得られた解析結果に基づき自動対処を実現する際の課題を以下に示す.

【課題 1】解析結果にノイズが含まれている

マルウェア動的解析システムは，解析環境においてマルウェアを実行した際の通信先等の挙動を観測し解析結果として出力する．しかし，マルウェアの中には，ネットワークの疎通確認を行うために，正規サイトへアクセスを行うものが存在する．また，マルウェアの実行中に解析環境にインストールされた OS やアプリケーションが行う正規の挙動が解析結果に含まれる場合もある. このような解析結果のノイズにより正規サイトへの通信を誤って遮断してしまう場合がある.

【課題 2】対処すべき機器や対処方法が分からない

解析結果に含まれる挙動から，どの機器でどのような対処をすれば良いか分からない．また，たとえ対処方法が分かっていたとしても，対処の適用がバッチ処理で行われることにより，通信遮断までのタイムラグが発生してしまう.

【課題 3】業務を止められない

セキュリティ対処を適用することにより，業務にどのような影響がでるか分からない．このため，業務影響を調査しなければならず，この調査に時間を要する.

これらの課題から，マルウェア自動対処に求められる要件を整理した．以下に，自動対処システムに求められる要件を示す.

【要件 1】解析結果のノイズを除去すること

解析結果に含まれる正規サイトの情報や，OS やアプリケーションが行う正規の挙動を除去すること.

【要件 2】機器に適用可能な脅威を抽出すること

セキュリティ機器ごとに適用可能な脅威を抽出し，自動で対処すること.

【要件 3】対処した際の影響を評価すること

対処した際に発生する業務への影響を見積もり，影響度に応じて自動対処の可否を設定できること．

6.2.2. 自動対処システムの提案

6.2.1 項で整理した要件に基づき，自動対処システムを提案する．以下に各要件に対する対応方針を述べる．

【要件 1】への対応として，脅威情報の評価を行う機能を開発する．脅威情報の評価では，外部のセキュリティベンダの情報等を参考に，脅威情報の確信度（当該脅威情報がどれだけ信頼できるかの指標）を算出する．なお，確信度の算出方法は後述する．

【要件 2】への対応として，マルウェア解析結果からセキュリティ機器に適用可能な脅威情報（例えば，マルウェアが接続を行うドメインや IP アドレス等）を抽出する機能と，組織に配備されているセキュリティ機器ごとに脅威情報を受け取り，対処を自動適用する機能（以下，アダプタ）を開発する．

【要件 3】への対応として，対処影響の評価を行う機能を開発する．対処影響の評価では，これまでの業務活動の情報を参考に，業務への影響度（当該対処の適用によってどれだけ業務が阻害されるかの指標）を算出する．なお，影響度の算出方法は後述する．

提案する自動対処システムの概要を図 6.1 に示す．

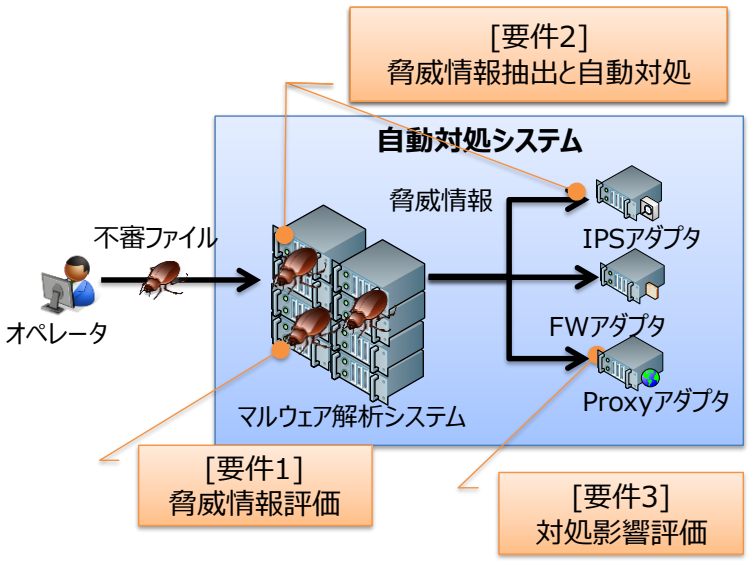


図 6.1　自動対処システムの概要

自動対処システムの具体的な処理の流れを説明する．

1. オペレータは，不審検体をマルウェア解析システムに投入する．
2. マルウェア解析システムは，投入された不審検体を動的解析し，対処に適用可能な脅威情報を抽出する．
3. マルウェア解析システムは，抽出した脅威情報の確信度を算出する．
4. マルウェア解析システムは，確信度が付与された脅威情報を，影響度評価のため，セキュリティ機器と共有する．
5. セキュリティ機器は，マルウェア解析システムから共有された脅威情報を受信し，対処を適用した場合の業務への影響度を算出する．
6. セキュリティ機器は，脅威情報に付与された確信度と，対処を適用した場合の影響度とから，対処の適用可否を判断し，適用が可能な場合は対処する．

このように，マルウェアの解析結果から脅威情報を抽出し，脅威情報の確信度と，対処した場合の影響度を用いることで，業務影響を考慮した自動対処を行うことが可能となる．以降では，自動対処システムの詳細について述べる．

6.3. 自動対処システムの設計

6.3.1. 自動対処システムの概要

提案する自動対処システムの構成を図 6.2 に示す．

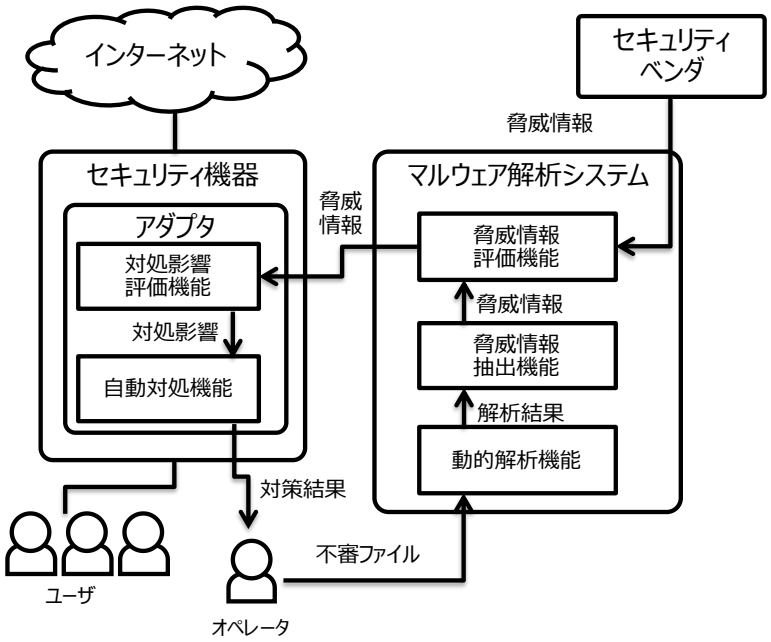


図 6.2　自動対処システムの構成

提案する自動対処システムは，以下 5 つの機能から構成される．

(ア) 動的解析機能

投入された不審検体を動的解析する機能.

(イ) 脅威情報抽出機能

動的解析の解析結果から脅威情報を抽出する機能．セキュリティ機器に適用可能な脅威情報を抽出する．

(ウ) 脅威情報評価機能

抽出した脅威情報の確信度を算出し，確信度を付与した脅威情報を共有する機能．なお，マルウェア解析システムと各セキュリティ機器での脅威情報の共有は，サイバー攻撃活動を記述するための仕様である STIX を用いて行う．

(エ) 対処影響評価機能

脅威情報を受け取り，対処を実行した際の影響度を評価する機能．セキュリティ機器ごとに影響度を評価するアダプタを開発する．

(オ) 自動対処機能

確信度と，影響度を基に自動で対処を適用する機能．なお，セキュリティ機器ごとに対処を適用するアダプタを開発する．

6.3.2. 自動対処システムの詳細

本項では，提案システムを構成する機能の詳細を説明する．

(ア) 動的解析機能

動的解析機能では，マルウェアをサンドボックス環境で実行し，マルウェアの挙動を明らかにする．なお，マルウェアの挙動を解析する動的解析製品として，FireEye[103]や，ThreatAnalyzer[104]などが存在する．また，OSS では Cuckoo Sandbox が入手可能である．提案する自動対処システムでは，著者のグループが開発している，多種環境でマルウェアの解析を行う M3AS を活用する．

(イ) 脅威情報抽出機能

脅威情報抽出機能では，動的解析機能が出力した解析結果から，セキュリティ機器に適用可能な脅威情報を抽出する．組織で活用されている主なセキュリティ機器で対処に活用できる脅威情報を表 6.1 に示す．なお，括弧内には，セキュリティ機器を実現する代表的なオープンソースのソフトウェアを示す．

表 6.1　セキュリティ機器で利用可能な脅威情報

	プロキシ (Squid[5])	IPS (Snort[6])	FW (iptables[7])
IP アドレス	○	○	○
ドメイン	○	○	×
URL	○	○	×
User-Agent	○	○	×
コンテンツ	×	○	×

例えば，プロキシで対処を行う場合には，IP アドレスや，ドメイン，URL，ユーザエージェントの脅威情報を抽出し，共有する．

(ウ) 脅威情報評価機能

上述したように，解析結果から抽出した脅威情報には正規サイトや正規の挙動情報が含まれる場合がある．脅威情報評価機能では，脅威情報に確信度を付与することで，これら正規サイトや正規の挙動を誤って遮断してしまうことを抑制する．脅威情報評価機能では，抽出した脅威情報に対して，セキュリティベンダ等の外部のデータベースを参照し確信度を算出する．提案する自動対処システムでは，VirusTotal[8]を参照し，確信度として High，Middle，Low，None，Unknown のいずれかを出力する．VirusTotal とはユーザが投稿された検体や，ドメイン，URL を複数のウイルス対策ソフトや，Web サイトスキャナなどを用いて解析した結果を返すサービスである．提案手法では，悪性と判断した Web サイトスキャナの数に応じて確信度を算出する．なお，評価対象の脅威情報が VirusTotal に登録されていない場合は，確信度 Unknown を出力する．VirusTotal を用いた確信度の設定例を表 6.2 に示す．

[5] http://www.squid-cache.org/
[6] https://www.snort.org/
[7] https://www.netfilter.org/index.html
[8] https://www.virustotal.com/

表 6.2　VirusTotal を用いた確信度設定

確信度	悪性と判断した Web スキャナの数
None	0
Low	1
Medium	2
High	3 以上
Unknown	-

さらに，脅威情報評価機能は，脅威情報を STIX として出力する．STIX で出力することで，提案システム以外の STIX を活用したセキュリティ機器との連携も可能となる．図 6.3 に脅威情報の出力例を示す．

```
<?xml version="1.0"?>
- <stix:STIX_Package version="1.2" id="example:Package-023b27dd-
083f-4563-9258-4efa0056e1a9"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:cyboxCommon="http://cybox.mitre.org/common-2"
xmlns:example="http://example.com"
xmlns:URIObj="http://cybox.mitre.org/objects#URIObject-2"
xmlns:stixCommon="http://stix.mitre.org/common-1"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:indicator="http://stix.mitre.org/Indicator-2"
xmlns:cybox="http://cybox.mitre.org/cybox-2"
xmlns:stixVocabs="http://stix.mitre.org/default_vocabularies-1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:stix="http://stix.mitre.org/stix-1">
  - <stix:STIX_Header>
      <stix:Description>M3AS:STIX</stix:Description>
    </stix:STIX_Header>
  - <stix:Indicators>
      - <stix:Indicator id="example:indicator-d8e915a9-b215-42ba-
      96e4-f81e6455fe8e" xsi:type="indicator:IndicatorType"
      timestamp="2017-07-20T04:29:15.688158Z">
          <indicator:Type
              xsi:type="stixVocabs:IndicatorTypeVocab-1.1">Domain
              Watchlist</indicator:Type>
        - <indicator:Observable id="example:Observable-4878bc33-
          9f54-4f6f-8a76-26fc87895824">
            - <cybox:Object id="example:URI-3d155515-d634-48e1-
            9d3f-9c2bc976bad6">
              - <cybox:Properties type="Domain Name"
                xsi:type="URIObj:URIObjectType">
                    <URIObj:Value>apartments-
                        gurgaon.com</URIObj:Value>
                </cybox:Properties>
              </cybox:Object>
          </indicator:Observable>
        - <indicator:Confidence timestamp="2017-07-
          20T04:29:16.606258Z">
              <stixCommon:Value>High</stixCommon:Value>
          </indicator:Confidence>
      </stix:Indicator>
    </stix:Indicators>
</stix:STIX_Package>
```

図 6.3　脅威情報の例

(エ) 対処影響調査機能

対処影響調査機能では，脅威情報を受け取り，対処を適用した際の影響度を評価する．対処影響調査機能はセキュリティ機器ごとに実装する．例えば，Proxy（Squid）の場合は，アクセスログを参照し，脅威情報として出力されたドメインへのアクセス割合を算出し，影響度とする．なお，アクセスログを SIEM や DB に格納している場合は，そこから影響度を算出してもよい．

また，FW で対処を適用した場合の影響度を算出するには，FW を通過するログを記録する設定で運用しておく必要がある．IPS は，通常シグネチャにマッチしたログのみを記録する．このため，IPS で対処を適用した場合の影響度を算出するには，脅威情報に含まれるシグネチャを生成し，その後，一定期間運用した後に算出しなければならない．ただ，表 6.1 で示したように，他のセキュリティ機器から影響度を算出することもできる．例えば，Proxy を運用していれば，IP アドレス，ドメイン，URL，ユーザエージェントの情報をログに記録することができ，そこから FW や，IPS で対処を適用した場合の影響度を算出してもよい．

(オ) 自動対処機能

自動対処機能では，脅威情報，確信度，対処影響度を受け取り，対処の適用可否を決定し，セキュリティ対処を実行する．対処適用の設定例（自動対処を行う確信度と対処影響度の組合せ）を表 6.3 に示す．なお，この設定値は運用に応じて変化させてもよい．

表 6.3　自動対処の設定

確信度	対処影響度
None	0%
Low	1%以下
Medium	5%以下
High	10%以下
Unknown	3%以下

確信度が高ければ，対処による影響がありそうな場合でも対処を自動適用するが，確信度が低ければ対処による影響が出そうな場合に自動適用しないように設定することができる．上記設定例は，確信度が High の場合は対処により 10%の通信に影響が出る場合でも対処を自動適用することを表している．

6.4. 評価実験

提案手法の有効性評価のため，自動対処システムのプロトタイプを実装し，評価した．本章では，開発した自動対処システムの評価実験について述べる．なお，プロトタイプでは，プロキシ

のアダプタを実装し，マルウェアの接続先ホストを脅威情報として抽出した．

6.4.1. 評価目的

開発した自動対処システムを以下の観点で評価する．

(ア) 自動対処速度について

検体の解析から自動対処までの時間を評価する．

(イ) 自動対処精度について

マルウェア解析結果から得られた脅威情報のうちどの程度が自動対処されるかを評価する．

(ウ) 確信度のみと影響度のみを用いた自動対処について

マルウェア解析結果から得られた脅威情報を確信度のみを用いて対処した場合と，影響度のみを用いて対処した場合，確信度と影響度を用いて対処した場合（提案手法）の 3 パターンについて，どの程度が自動対処されるかを評価する．

(エ) 影響度評価に用いるユーザ数と期間について

対処影響調査機能において，業務情報として利用するユーザ数を n_u 人，利用する期間を N_u 日とし，n_u と N_u および FP の関係を評価する．

6.4.2. 評価方法

(ア) 自動対処速度について

提案システムのプロトタイプを用いて 100 検体を解析し，各機能の処理にかかる時間を計測する．

(イ) 自動対処精度について

提案システムのプロトタイプを用いて 732 検体を解析し，抽出した脅威情報の確信度を評価する．また，ある組織におけるプロキシのアクセスログを用いて脅威情報のうちどの程度が自動対処されるかを机上検討する．表 6.4 に評価に用いたプロキシログの情報を示す．

表 6.4　プロキシログの概要

項目	値
ユーザ数	34 ユーザ
期間	2016/2/25-3/5
アクセス総数	543,914 アクセス
接続先ホスト数（ユニーク）	5,110 ドメイン

(ウ) 確信度のみと影響度のみを用いた自動対処について

「(イ)自動対処精度について」で用いた検体およびプロキシアクセスログを用いて，確信度 None の脅威情報を自動対処の対象外した場合と，影響度が 0 より大きい（一度でもアクセス

がある）脅威情報を自動対処の対象外とした場合，提案手法の 3 パターンについて，自動対処される脅威情報の数と，業務影響が存在する脅威情報の数を評価する．

(エ) 影響度評価に用いるユーザ数と期間について

業務情報として利用するユーザ数（n_u=1,17,34）と期間（N_u=1,3,7）について FP を評価する．評価には，「(イ)自動対処精度について」で用いた検体およびプロキシアクセスログを用いる．また，自動対処した脅威情報のうち，評価対象期間（34 ユーザのプロキシログ 3 日分）にアクセスが観測された脅威情報の割合を FP として評価する．

6.4.3. 評価結果

(ア) 自動対処速度について

各機能の 1 検体あたりの平均処理時間を表 6.5 に示す．

表 6.5　自動対処システムの処理時間

項目	処理時間[秒]
動的解析機能	460
脅威抽出機能	0.14
脅威評価機能	2.6
対処影響評価機能	1.0
自動対処機能	0.54
計	464.28

1 検体あたりの平均処理時間は約 464 秒となった．

(イ) 自動対処精度について

732 検体から脅威情報（不審ホスト）は 3,739 ホスト抽出された．なお，ユニークな脅威情報は 508 ホスト存在し，そのうちの 491 ホストが自動対処の対象となった．確信度別の脅威情報抽出数及び，対処影響数，自動対処対象数を表 6.6 に示す．ここで，対処影響数とは，影響度が 0%以上の（評価に用いたプロキシログで，1 度でも接続が確認された）脅威情報の数を表す．

表 6.6　評価結果

確信度	脅威情報数	対処影響数	自動対処数
None	132	17	115
Low	118	2	118
Medium	80	0	80
High	159	0	159
Unknown	19	0	19
計	508	19	491

表 6.6 より，確信度 None の脅威情報のうち，対処影響が確認された 17 件が自動対処の対象外となったことが分かる．また，確信度 Low の脅威情報のうち，2 件は対処影響が確認されたが，影響度が少なかったため，自動対処の対象となったことが分かった．なお，確信度 Low で対処影響が確認された接続先は，短縮 URL サイトと，CDN（Content Delivery Network）で利用されているドメインであった．

(ウ) 確信度のみと影響度のみを用いた自動対処について

確信度 None の脅威情報を自動対処の対象外した場合と，影響度が 0 より大きい（一度でもアクセスがある）脅威情報を自動対処の対象外とした場合，提案手法の 3 パターンについて自動対処数と，自動対処によって業務へ影響が出る脅威情報の数（対処影響数）の関係を表 6.7 に示す．

表 6.7　自動対処数と影響数

	自動対処数	対処影響数
確信度のみ	376	2
影響度のみ	489	0
提案手法	491	2

表 6.7 より，提案手法は 3 パターンのうちで最も多く自動対処を実施していることが分かる．対処影響度のみを用いて自動対処を行う場合，影響数は 0 となるが，確信度 Low の脅威情報 2 件を対処することができない．確信度と影響度を用いて柔軟に対処の設定を変更できる提案手法はこの点で確信度のみを用いた自動対処や影響度のみを用いた自動対処よりも優れていると考える．

(エ) 影響度評価に用いるユーザ数と期間について

影響度評価時の業務情報として利用するユーザ数（n_u=1,17,34）と期間（N_u=1,3,7）を変化させた際の FP を図 6.4 に示す.

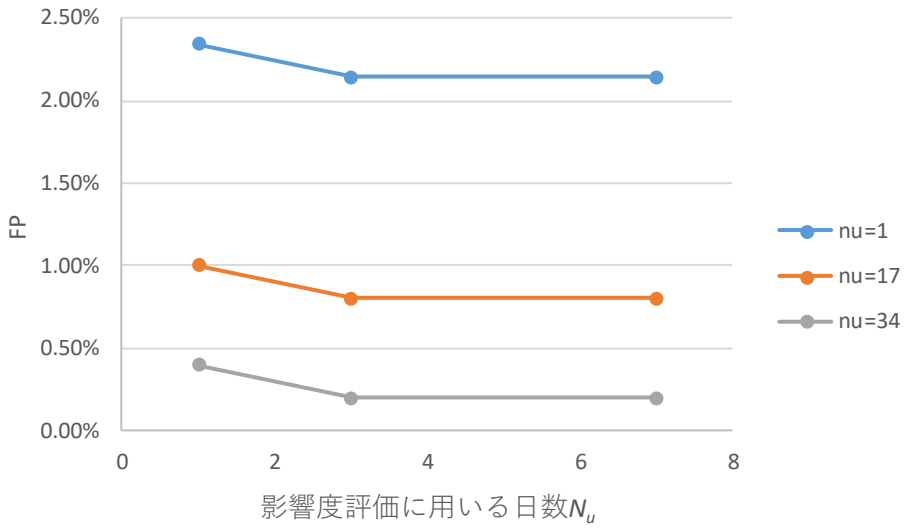


図 6.4　ユーザ数および期間と FP

図 6.4 より，影響度評価に用いるユーザ数 n_u が増えるほど，FP の値が低下することが分かる．また，影響度評価に用いる日数は 3 日程度あれば十分であることも確認できた.

6.5. 考察

(ア) 対処漏れの可能性について

提案した自動対処システムでは，マルウェアを動的解析して得られた脅威情報を用いて対処を行う．例えば，DGA（Domain Generation Algorithm）を用いて解析の度に接続先を生成するマルウェアや，ランダムな接続先へ拡散を試みるマルウェアなどは，動的解析で接続先全てを抽出することができず，対処漏れを起こしてしまう可能性が発生する．本問題については 7 章で解決方法を述べる.

(イ) 外部インテリジェンス情報との連携について

近年，組織間でインテリジェンス（脅威情報や IT 機器の脆弱性情報およびそれらに関する分析や対処支援情報）を共有して攻撃に備える集団防御の概念が浸透しつつある．提案した自動対処システムでは，マルウェアの動的解析結果から得られた脅威情報を自動対処の対象としたが，これらのインテリジェンスから得られた脅威情報を用いて自動対処を実現することで，より多くの攻撃に対処できるようになると考えられる.

6.6. まとめ

本章では，組織における過去の接続ログから対処を適用した際の影響を算出し，算出した影響度に基づき自動対処を行うシステムを提案した．さらに，プロキシと連携したプロトタイプを用いた評価実験により，464 秒でマルウェアの動的解析から対処適用までが完了すること，実マルウェアから 508 件の脅威情報を抽出し，そのうちの 491 件を自動対処できることを確認した．

7. ホワイトリストを用いた自動対処

7.1. 導入

本章では不審サイトの情報（グレーリスト）を用いずに遠隔操作型マルウェアの通信を遮断する，ドメインホワイトリスト（以下，ホワイトリスト）を用いた自律進化型防御システム（以下，ホワイトリスト型 AED）を提案する．ホワイトリスト型 AED では，安全なサイト（ホスト名（FQDN）及び IP アドレス）をホワイトリストとして管理しており，ホワイトリストに合致しない接続先への HTTP 通信が発生した際に追加認証を要求する．これにより，未知のサイトを利用した攻撃にも対応することが可能となる．

7.2. ホワイトリスト型 AED の提案

本章では，ホワイトリストに定められた接続先以外には追加認証を要求することにより，人間による意図的な通信は許可するとともに，認証結果を用いてホワイトリストの精度を高めていく，ホワイトリスト型 AED を提案する．

7.2.1. サイバー攻撃の流れ

遠隔操作型マルウェアを用いたサイバー攻撃の流れを図 7.1 に示す．

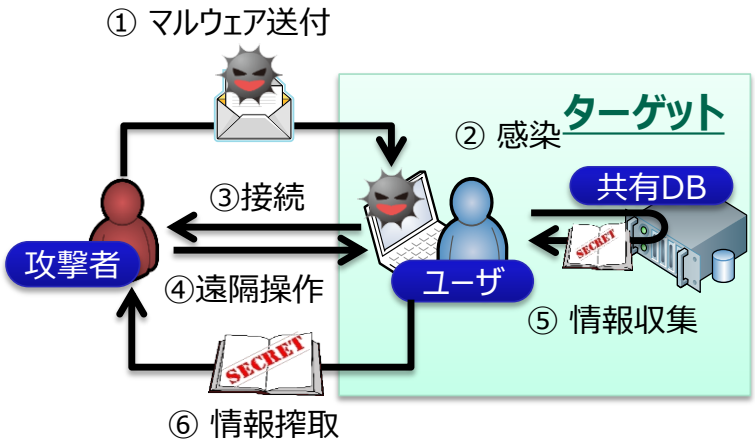


図 7.1　サイバー攻撃の流れ

まず，攻撃者は標的としている組織に，遠隔操作型マルウェアを添付したメールを送付（図 7.1①）する．組織のユーザが誤って添付ファイルを実行すると，遠隔操作型マルウェアに感染（図 7.1②）してしまう．組織に侵入した遠隔操作型マルウェアは，攻撃者との HTTP 通信を確立するため，外部にいる攻撃者に接続（図 7.1③）する．攻撃者は遠隔操作型マルウェアとの HTTP 通信を確立し，感染した端末を遠隔操作（図 7.1④）して内部情報の収集（図 7.1⑤）を行い，最

終的に情報を搾取（図 7.1⑥）する．

提案するホワイトリスト型 AED では，遠隔操作型マルウェアが攻撃者と接続を行う点（図 7.1③）に着目し，ホワイトリストに存在しない接続先に HTTP 通信が発生した際に，マルウェアには解決困難な認証手段（提案システムでは CAPTCHA を用いるが，他の手段でも構わない）を要求することで，遠隔操作型マルウェアによる HTTP 通信を遮断する．これにより，たとえ遠隔操作型マルウェアの組織侵入を許したとしても，マルウェアと攻撃者の通信を遮断し，情報漏えい等の事故を防止することが可能となる．

7.2.2. 提案システムの概要

提案するシステムの概要を図 7.2 に示す．

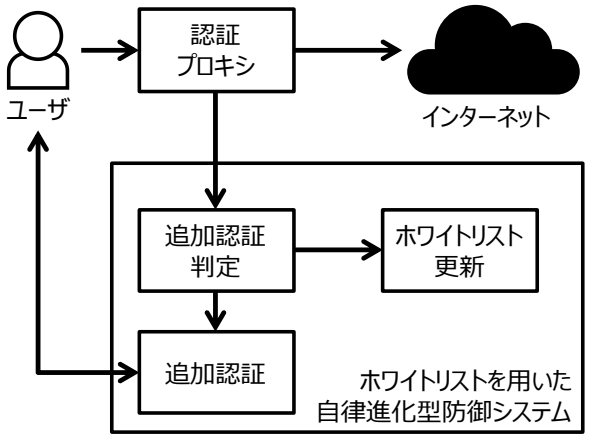


図 7.2　ホワイトリスト型 AED

提案するホワイトリスト型 AED は，認証プロキシ（認証機能付きプロキシ）と連携し，遠隔操作型マルウェアの通信を遮断する．ホワイトリスト型 AED が具備する 3 つの機能を以下に示す．なお，これらの機能の詳細については，7.2.3 項で述べる．

(ア) 追加認証判定

接続先をホワイトリストや過去の追加認証判定結果と比較し，追加認証を要求するか否かの判定を行う機能

(イ) 追加認証

ユーザに対する追加認証を生成し，追加認証に対する応答を受け取る機能

(ウ) ホワイトリスト更新

ユーザの追加認証結果を集計し，ホワイトリストの更新を行う機能

ホワイトリスト型 AED は，ユーザからの接続要求を受け取ると，接続先をホワイトリストや当該ユーザの過去の追加認証判定結果と比較し，追加認証を行うか否かの判定を行う．追加認証が必要と判定された場合は，追加認証（CAPTCHA）を要求する．ユーザが CAPTCHA を解釈し，正しく入力を行えば，当該接続は人による接続（遠隔操作型マルウェアによる C&C サーバへの通信ではない）と判断し，インターネットへの接続を許可する．

さらに，ある一定数のユーザが追加認証に成功している接続先は業務でよく利用される接続先とみなし，ホワイトリストに追加する．

以上のように，ホワイトリストと CAPTCHA を用いた接続制御により，遠隔操作型マルウェアの通信を遮断し，さらに，ホワイトリストの更新により，業務へ与える影響を抑制する．

7.2.3. 提案システムの詳細

提案システムを構成する 3 つの機能の詳細を説明する．

(ア) 追加認証判定

追加認証判定機能の処理フローを図 7.3 に示す．

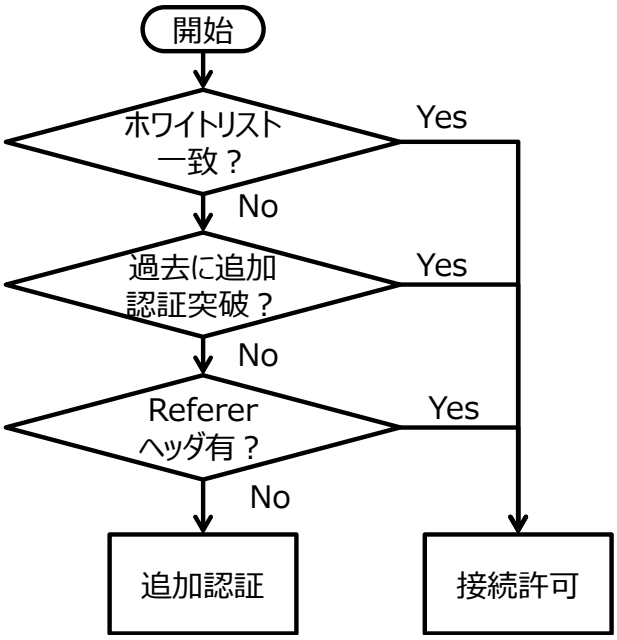


図 7.3　追加認証判定フロー

追加認証判定機能では，プロキシサーバに届いたユーザからのリクエスト情報を受信し，接続先のドメインがホワイトリストに登録されているか否かを判断する．接続先のドメインがホワイトリストに登録されている場合には，ユーザの接続を許可し，登録されていない場合には，当該ユーザが，過去に当該接続先に接続を実施したか否かを判断する．当該ユーザが，過

去に当該接続先に接続を実施していた場合には，ユーザの接続を許可し，接続を実施していない場合には，追加認証機能へリクエスト情報を送信する．

なお，HTML ファイルが外部サーバに格納された CSS ファイル等を参照していた場合にレイアウトが崩れるなどの問題が発生しないようにするため，ユーザからのリクエスト情報に Referer ヘッダが含まれている場合は接続を許可することとした．

また追加認証判定機能では，過去の接続状況を管理するため，以下の情報をアクセスログとして記録する．

・アクセス日時
・ユーザ識別用 ID（認証プロキシのユーザ名）
・アクセス元 IP アドレス
・アクセス先 URL
・Referer ヘッダの情報

(イ) 追加認証

追加認証機能では，CAPTCHA 認証に必要な歪み画像および認証フォームの生成を行う．また，CAPTCHA に対するユーザ入力の正当性を判定する．追加認証機能が生成する認証フォームの例を図 7.4 に示す．追加認証フォームには，「イメージ取得」ボタンも表示する．本ボタンを押下すると，ユーザが接続しようとしているサイトのスクリーンショットを取得し，ユーザに表示する．これは，ユーザの接続可否判断を補助するための措置である．

図 7.4　追加認証フォーム

(ウ) ホワイトリスト更新

ホワイトリスト更新機能は，事前に定めたある一定以上のユーザが CAPTCHA をクリアした場合には業務で利用する接続先とみなし，当該接続先をホワイトリストに登録する．

7.3. 実装

本節では，ホワイトリスト型 AED の実装について述べる．ホワイトリスト型 AED の各機能の構成を図 7.5 に，これらの実装に用いたソフトウェアを表 7.1 に示す．

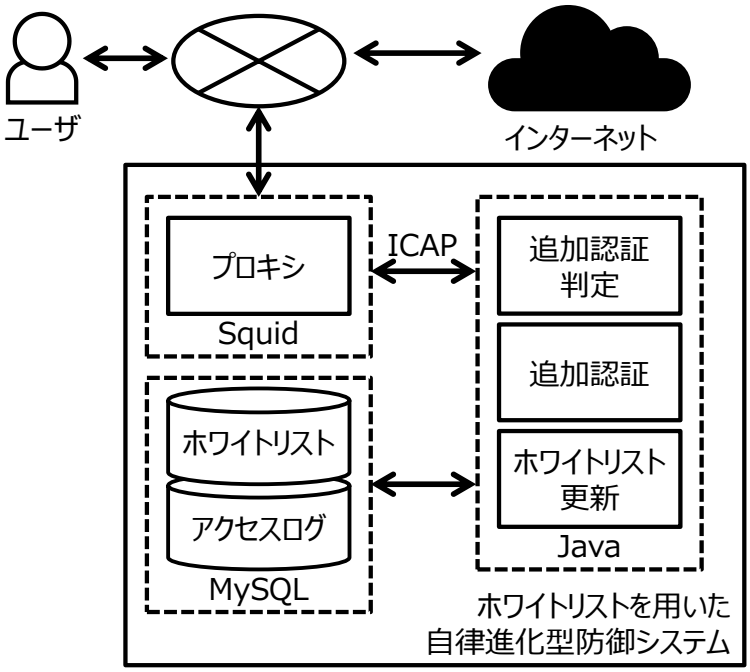


図 7.5　提案システムの実装

表 7.1　実装に用いたソフトウェア

機能	実装に用いたソフトウェア
追加認証判定機能，ホワイトリスト更新機能	Java 1.8.0
追加認証機能	Java Servlet 3.0
データベース（ホワイトリスト，アクセスログ）	MySQL 14.14
プロキシ	squid 3.3.8

なお，プロキシと Java の通信には ICAP（Internet Content Adaptation Protocol）[9]を用いた．

7.4. 評価実験

本節では，開発したホワイトリスト型 AED の評価結果について述べる．

7.4.1. 評価目的

ホワイトリスト型 AED は，業務への悪影響を抑えつつ，遠隔操作型マルウェアの通信を遮断するシステムである．遠隔操作型マルウェアの通信遮断精度と，業務への影響について，以下の観点で評価する．

[9] https://tools.ietf.org/html/rfc3507

(ア) 遠隔操作型マルウェアの遮断精度について

開発したプロトタイプを用いた評価実験を行い，遠隔操作型マルウェアの通信遮断精度を評価する．

(イ) 追加認証の要求率について

開発したプロトタイプを用いた評価実験を行い，業務時の追加認証要求率（総接続数に対する追加認証要求数の割合）を評価する．

(ウ) 業務への影響に関するアンケートについて

被験者へのアンケート調査により業務への影響度を評価する．

(エ) ユーザ数について

ホワイトリスト型 AED は利用するユーザが多いほど，業務への影響が抑制される．ホワイトリスト型 AED を利用するユーザ数を n とし，n と追加認証要求率の関係を評価する．

(オ) ホワイトリスト更新の閾値について

ホワイトリスト型 AED はホワイトリスト更新の閾値が少ないほど，業務への影響が抑制される．ホワイトリストへ移行させるユーザ数を k とし，k と追加認証要求率の関係を評価する．

7.4.2. 評価方法

(ア) 遠隔操作型マルウェアの遮断精度について

世の中に存在する遠隔操作型マルウェア（50 検体）を入手し，開発したプロトタイプを用いて，当該遠隔操作型マルウェアの通信を遮断できるか評価する．なお，評価に用いた遠隔操作型マルウェアの例を表 7.2 に示す．なお，評価に用いたマルウェアの詳細は，付録 A に示す．

表 7.2　評価用遠隔操作型マルウェアの例

種類	ハッシュ値（MD5）
Emdivi	e5653a4bca1239b095509438a3040244
PlugX	5a22e5aee4da2fe363b77f1351265a00
ChChes	8a93859e5f7079d6746832a3a22ff65c

(イ) 追加認証の要求率について

提案システムのプロトタイプを用いて業務時の追加認証要求率を評価する．具体的には，著者を含む 50 名のユーザに提案システムの利用を依頼し，初期ホワイトリストを空にした状態で実験を開始した．なお，標的型攻撃メール訓練の開封率が 10%である[105]ことから，組織内にはセキュリティ意識の低いユーザが 10%存在すると考えられる．本実験でもセキュリテ

ィ意識の低いユーザが 10%存在する（著者を除く実験参加者 49 名の 10%は CAPTCHA が表示された際に内容を確認せずに入力してしまう）との仮説をたて，実験参加者の 10%を超えるユーザが CAPTCHA 認証に成功した際にホワイトリストに追加することとした（k=5）．

実験期間中は，ユーザ自身が普段利用するブラウザのプロキシ設定を変更し，WEB アクセスに際してホワイトリスト型 AED を経由するように設定する．なお，2016 年 2 月 8 日～2016 年 2 月 24 日まで実験を行った．

(ウ) 業務への影響に関するアンケートについて

「(イ)業務への悪影響について」の被験者に対し，実験終了後にアンケートを実施する．なお，アンケートはプライバシーを考慮し，無記名で回答してもらう．

(エ) ユーザ数について

ホワイトリスト型 AED を利用するユーザ数（n=10, 50, 100, 500, 1000）について，追加認証の要求率を評価する．評価には，ある組織のプロキシアクセスログ（15 日分）を用い，ホワイトリスト型 AED を活用した場合の追加認証の要求率を机上検討する．具体的には，15 日分プロキシアクセスログの中から n ユーザ分のアクセスログを抽出し，アクセスごとに図 7.3 に示した処理フローに従い，追加認証を要求するか否かを判定する．なお，追加認証を要求すると判定された際には，当該ユーザが追加認証をクリアするものと仮定し，追加認証をクリアしたユーザ数が 5 ユーザ（k=5）になった接続先はホワイトリストに追加する．これらの処理をおこない，n ユーザ分の総接続回数に対する追加認証要求数を追加認証要求率として評価する．なお，評価は選択するユーザをランダムに変化させながら 10 回行い，その平均追加認証要求率を用いた．

(オ) ホワイトリスト更新の閾値について

ホワイトリストへ移行させるユーザ数の閾値（k=1, 5, 10, 25, 50）について，追加認証の要求率を評価する．評価には，ある組織のプロキシアクセスログ（15 日分）を用い，ホワイトリスト型 AED を活用した場合の追加認証の要求率を机上検討する．具体的には，15 日分プロキシアクセスログの中から 100 ユーザ分（n=100）のアクセスログを抽出し，アクセスごとに図 7.3 に示した処理フローに従い，追加認証を要求するか否かを判定する．なお，追加認証を要求すると判定された際には，当該ユーザが追加認証をクリアするものと仮定し，追加認証をクリアしたユーザ数が k になった接続先はホワイトリストに追加する．これらの処理をおこない，100 ユーザ分の総接続回数に対する追加認証要求数を追加認証要求率として評価する．なお，評価は選択するユーザをランダムに変化させながら 10 回行い，その平均追加認証要求率を用いた．

7.4.3. 評価結果

(ア) 遠隔操作型マルウェアの遮断精度について

評価用マルウェア（50 検体）に対する，遮断精度を表 7.3 に示す．なお，認証プロキシを利用した手法[106]との比較も併せて示す．なお，検体ごとの遮断有無については，付録 A で示す．

表 7.3　遮断精度評価結果

	遮断検体数	遮断精度
提案手法	50	100%
認証プロキシ[106]	41	82%

表 7.3 より, 提案手法は評価に用いた遠隔操作型マルウェア全てを遮断することが確認できた.

(イ) 追加認証の要求率について

実験期間中の総接続数と，追加認証有無の数を表 7.4 に，ホワイトリスト数の推移を図 7.6 に示す．なお，ノイズを除去するため，実験期間を通じて定常的にホワイトリスト型 AED の利用が確認された 24 ユーザ（7 日以上利用していたユーザ）を評価の対象とした（n=24）．また，実験期間中に，検索エンジン（www.google.co.jp）や，総合情報サイト（itpro.nikkeibp.co.jp）等がホワイトリストに登録された．

表 7.4　評価結果

項目	数	割合
追加認証有り	12,115	1.93%
- 回答有り	1,620	0.26%
- 回答無し	10,495	1.67%
追加認証無し	616,718	98.07%
- 追加認証成功済接続先への接続	122,467	19.48%
- Referer ヘッダがついた接続	473,677	75.33%
- ホワイトリストへの接続	20,574	3.27%
計	628,833	100.00%

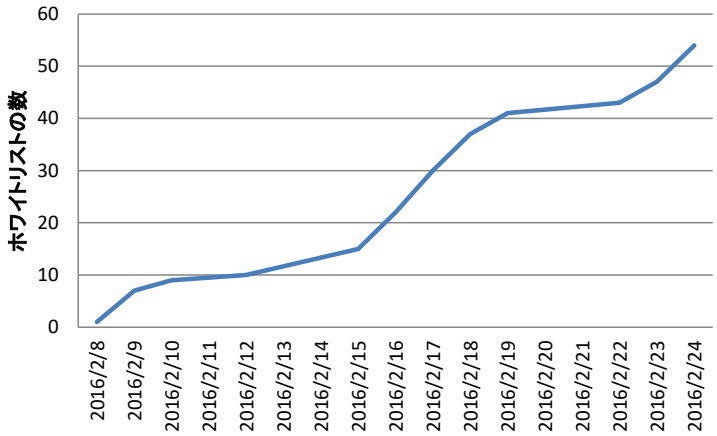


図 7.6　ホワイトリスト数の推移

表 7.4 より，追加認証要求率は 1.93%であることが分かる．なお，実験期間中には遠隔操作型マルウェアによる被害が確認できなかった．これより，1.93%は正常な通信に対して誤って追加認証を要求，すなわち FP が生じたこととなる．また，Referer ヘッダがついた接続に追加認証を発生させないようにする処理は，追加認証の発生頻度削減に大きく貢献していることが確認できた．

また，追加認証の要求数は 12,115 回だったが，追加認証に対する回答は 1,620 回（1 ユーザあたり 1 日平均 3.97 回追加認証へ回答）しか観測されなかった．追加認証に対する回答がなかったサイトには，CRL や OCSP といった電子証明書関連情報を提供するサイトやソフトウェアの更新確認用サイトなどが含まれていた．これらのサイトはブラウザや OS がバックグランドでアクセスするサイトのため，アクセスに際して発生した追加認証画面はユーザに提示されずタイムアウトしてしまう．このような接続先に関しては，予めホワイトリストに登録することにより，追加認証の要求を抑制できると考える．

(ウ) 業務への影響に関するアンケートについて

被験者へのアンケート結果を表 7.5 に示す．アンケートへの回答は 19 名から得られた．

表 7.5　アンケート結果

質問	選択肢	人数
追加認証はどれくらいの頻度で発生しましたか？	発生しなかった	0
	日に 1 回程度	3
	日に数回程度	13
	日に 10 回以上	3
追加認証の頻度についてどう感じましたか？	多くて不便	9
	この程度なら問題ない	10
	もっと多くても問題ない	0
期間中，追加認証の頻度は変化しましたか？	増えた	0
	変わらなかった	9
	減った	10

追加認証の頻度がユーザの利便性に影響を与えるとの仮説をたて，当該仮説を検証するため Fisher の正確確率検定を用いて統計的に有意な差があるかどうかを分析した．検定結果を表 7.6 に示す.

表 7.6　Fisher の正確確率検定

	この程度なら問題ない	多くて不便	p 値
追加認証が日に 1 回程度	3	0	
追加認証が日に数回	6	7	0.3731
追加認証が日に 10 回以上	1	2	
計	10	9	-

利便性に影響しないと考えるユーザ数（10 名）は，不便で影響ありとするユーザ数（9 名）を上回ったが，Fisher の正確確率検定の結果，追加認証の頻度とユーザの利便性の関係に有意な差が認められなかった.

また，アンケート結果より，約半数のユーザが不便と感じていることが分かった．これについては，ホワイトリスト型 AED を利用するユーザ数を増やす，あるいは初期ホワイトリストを活用することで改善できると考える．実験期間中に追加認証の頻度が減ったと感じたユーザ

が存在したが，表 7.4 の結果より，これは，ホワイトリストが更新されたことによる効果ではなく，一度認証に成功したドメインに対して認証を出さないようにする機能の効果だと考えられる．

(エ) ユーザ数について

ホワイトリスト更新の閾値を固定（k=5）した状態で，ホワイトリスト型 AED を利用するユーザ数 n（n=10, 50, 100, 500, 1000）を変化させた際の追加認証要求率を図 7.7 に示す．

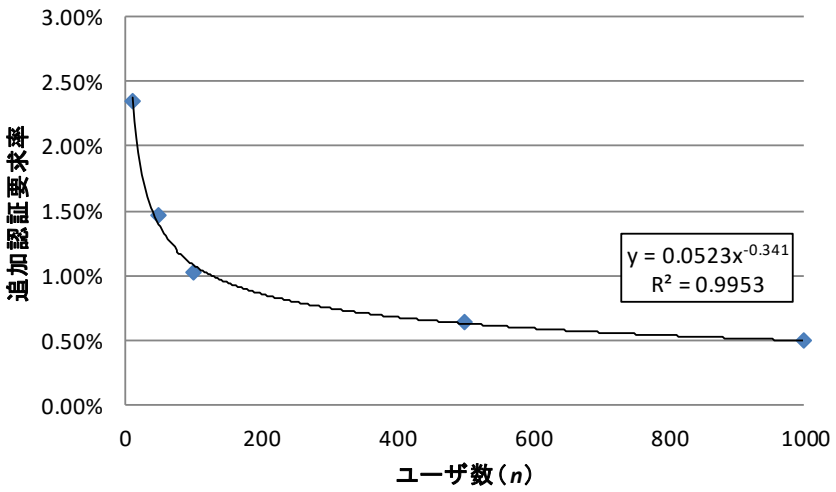


図 7.7　ユーザ数と追加認証要求率

図 7.7 より，ユーザ数が増えるに従い追加認証要求率の減少幅がならだかになることが分かる．

(オ) ホワイトリスト更新の閾値について

ホワイトリスト型 AED を利用するユーザ数を固定（n=100）した状態で，ホワイトリストへ移行させるユーザ数の閾値 k（k=1, 5, 10, 25, 50）を変化させた際の追加認証要求率を図 7.8 に示す．

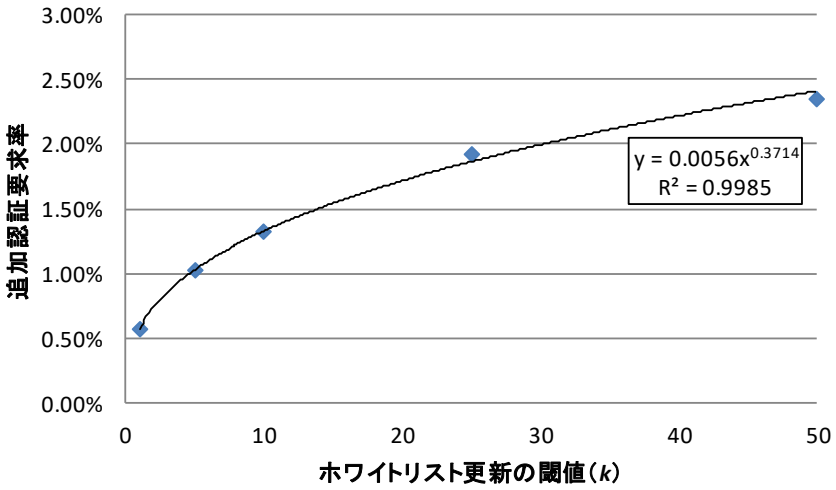


図 7.8　ホワイトリスト更新の閾値と追加認証要求率

図 7.8 より，ホワイトリスト更新の閾値が大きくなるに従い，追加認証要求率の増加幅がならだかになることが分かる．また，k が 5 から 1 へ変化すると，追加認証要求率は約半分になることが確認できた．

7.5. 考察

(ア) 業務に影響を与えないユーザ数について

プロトタイプを用いた評価実験結果から算出した，ユーザ当たりの追加認証要求数（1 日あたり）と追加認証回答数（1 日あたり）の関係を図 7.9 に示す．なお，ノイズを除去するため追加認証要求数の上位 5 ユーザと下位 5 ユーザのデータは除いてある．図 7.9 より，追加認証要求数に対する追加認証回答数はユーザごとにばらつきがあるが，ほぼ線形の関係にあることが分かる．

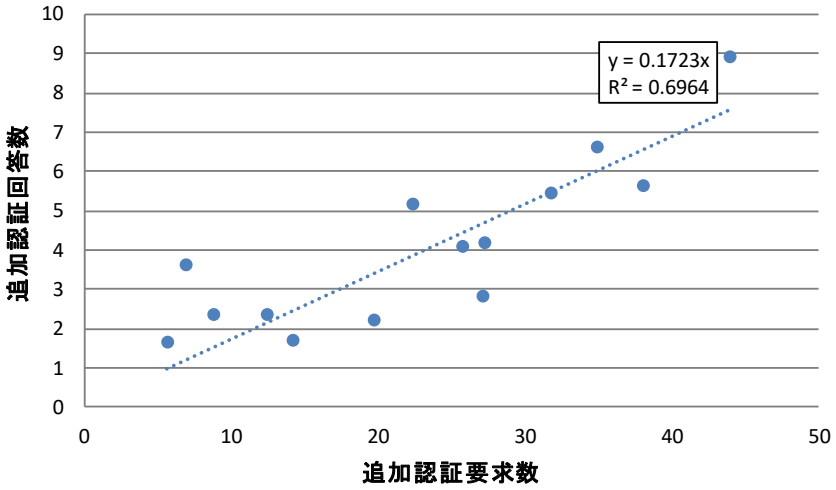


図 7.9　追加認証要求数と追加認証回答数

また，プロキシログを用いた机上検討結果から算出した，ユーザ数とユーザ当たりの追加認証要求数（1 日あたり）の関係を図 7.10 に示す．図 7.10 より，ユーザ数の増加に伴い追加認証要求数は減少することが分かる．

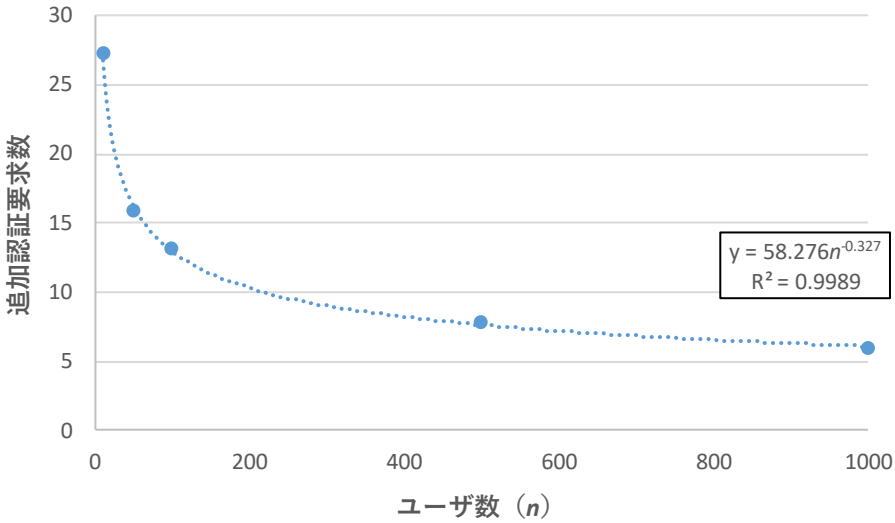


図 7.10　ユーザ数と追加認証要求数

さらに，アンケート結果（表 7.5）より，追加認証頻度が 1 日に 1 回程度であれば不便を感じるユーザは存在しないことが分かる．図 7.9 の近似式より，1 日に 1 回程度の追加認証回答が発生するのは追加認証要求数が 5.8 回の時であり，図 7.10 より，1 日に 5.8 回の追加認証

要求が発生するのはユーザ数が 1,000 人程度であることが確認できる．以上より，ユーザ数が 1,000 人程度になると，1 日の平均追加認証頻度が 1 回程度に抑えられ，業務に影響を与えることなく，遠隔操作型マルウェアの通信を遮断できるようになると言える．

(イ) 初期ホワイトリストの有効性について

評価実験により，電子証明書関連情報を提供するサイトやソフトウェアの更新確認用サイトはホワイトリストに移行されないことが分かった．このような接続先に関しては，予めホワイトリストに登録することにより，追加認証の要求を抑制できると考えられる．ここでは過去 N 日（N=1, 3, 7, 30）にユーザが接続した実績のある接続先を，初期ホワイトリストと定め，初期ホワイトリストを与えることにより，どの程度追加認証の頻度が削減されるかを考察する．具体的には，ホワイトリストに用いる日数 N を変化させながら初期ホワイトリストを生成し，5000 人（n=5000）の組織のプロキシアクセスログ（30 日分）を用い，アクセスごとに図 7.3 に示した処理フローに従い，追加認証を要求するか否かを判定する．なお，追加認証を要求すると判定された際には，当該ユーザが追加認証をクリアするものと仮定し，追加認証をクリアしたユーザ数が k になった接続先はホワイトリストに追加する．これらの処理をおこない，総接続回数に対する追加認証要求数を追加認証要求率として評価する．ホワイトリスト更新の閾値 k（k=1, 5, 10）とホワイトリストに用いる日数 N 日（N=1, 3, 7, 30）を変化させた際の追加認証要求率を図 7.11 に示す．

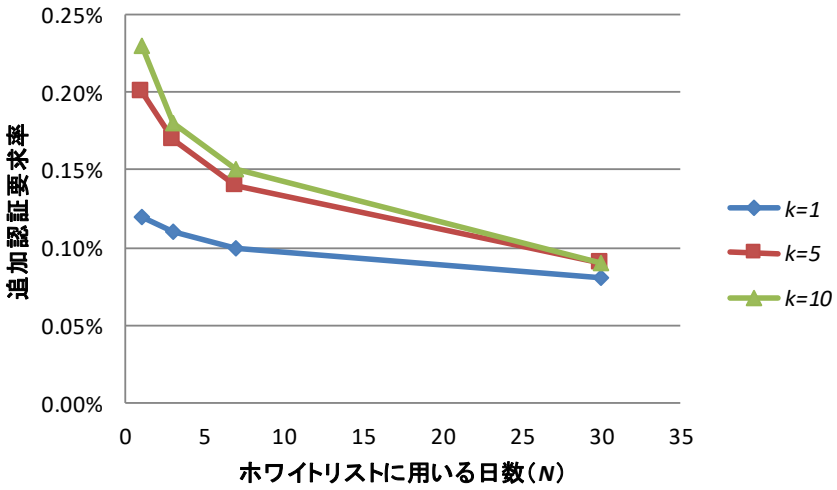


図 7.11　初期ホワイトリストと追加認証要求率

図 7.11 より，ホワイトリストに用いる期間が長いほど，追加認証要求率が低くなることが分かる．

なお，ここでは，過去当該組織がアクセスした接続先を初期ホワイトリストに加えたが，例えば，遠隔操作型マルウェアが既に組織に侵入していた場合には，遠隔操作型マルウェアの接続先が誤ってホワイトリストに追加されてしまう可能性がある．これに関しては，初期ホワイトリストに追加する際に，VirusTotal や，Site Safety Center[10]などのレピュテーションサイトの情報と比較し，安全と判明しているサイトのみを登録することで解決できると考える．

(ウ) Referer ヘッダの有無について

提案システムでは，HTML ファイルが外部サーバに格納された CSS ファイル等を参照していた場合にも，正しく表示させるため，Referer 付きの通信に対する認証を省略することとした．この結果，ほとんどの場合ユーザは問題なく Web 利用可能であることが判明した．ただし，https で取得されたページから http でリソースを取得する場合などにはスキームが変化するため Referer が付かず，通信が遮断されるという問題があることが明らかになった．この問題については，引き続き対策を検討していく．

(エ) 正規サイトを活用した遠隔操作について

遠隔操作型マルウェアの中には，Dropbox[11]のような正規サイトを悪用して攻撃を行うものも存在する[107]．正規サイトが既にホワイトリストに追加されている場合，ドメインレベルのホワイトリスト型 AED では対処することができない．このような問題に対応するためには，URL レベルでホワイトリストを作成する必要がある．しかし，URL レベルでホワイトリストを作成した場合には，追加認証の頻度があがり，業務へ影響を与えてしまう恐れがある．この問題についても引き続き対策を検討していく．

(オ) 提案システムの処理性能について

考察(ア)において，提案システムが実用に耐えうるユーザ数が 1,000 人程度であることを示した．ここでは，提案システムが 1,000 ユーザの処理に耐えられるか否かを確認するため，Apache Jmeter[12]を用いた性能評価を実施する．評価に利用した環境を表 7.7 に示す．なお，ホワイトリストの登録数は，1,000 ユーザのプロキシログを用いた机上検討の際にホワイトリストに移行したドメイン数より決定した．

10 https://global.sitesafety.trendmicro.com/
11 https://www.dropbox.com/
12 http://jmeter.apache.org/

表 7.7　評価環境の性能

項目	値
CPU	Intel Core i5-3470 3.2GHz
メモリ	4GB
ホワイトリストの数	21,000

Jmeter の同時スレッド数を変化させながら計測した，スループットと平均応答時間を図 7.12 に示す．なお，Jmeter のリクエストは，最も性能が要求される（追加認証が発生しない）パターンを用いた．

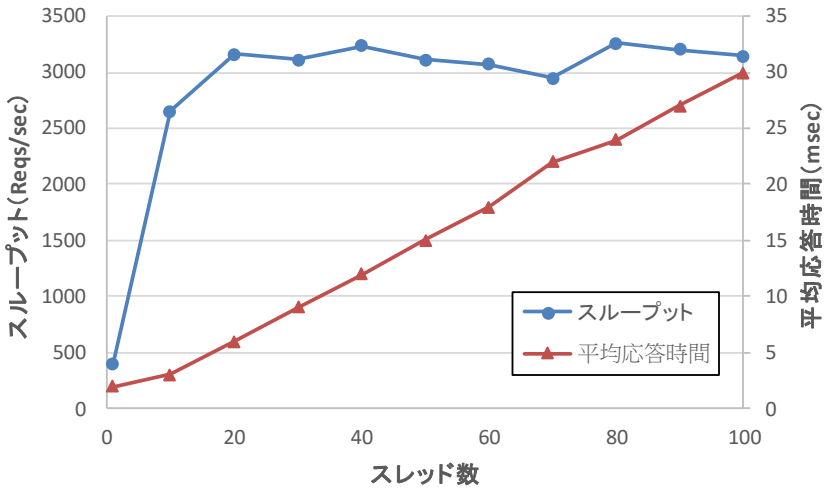


図 7.12　提案システムのスループットと平均応答時間

図 7.12 より，提案システムの処理性能は秒間 3,000 リクエスト程度であることが分かった．また，図 7.12 の平均応答時間が単調に増加していることより，処理性能の限界を超えたリクエストの処理待ちが発生し，応答時間の増大につながったと考えられる．なお，評価実験を通じて応答エラーは発生しなかった．1,000 ユーザのリクエストは，最大 748 リクエスト/秒（平均 22.9 リクエスト/秒）であったことから，処理性能の観点からは，提案システムによって 1,000 人のユーザを処理できることが確認できた．

(カ) 提案システムの限界について

利便性とリスクのバランスを考慮し，ホワイトリストを用いた自動対処システムでは

Referer ヘッダが付いた通信の追加認証を省略する方針とした．これにより，Referer ヘッダを用いてランディングサイトを経由し，マルウェアをダウンロードさせる Drive by download 攻撃や，外部サイトに設置された不正な javascript には対応できない．この問題に対処するためには，6 章で述べた脅威情報を用いた自動対処システムと組み合わせる必要がある．具体的には，脅威が既知（例えばすでに不正な javascript だと分かっている）の場合は，脅威情報を用いた自動対処により遮断することができる．また，脅威が未知の場合でも Referer がついている接続先がスクリプト形式(javascript,php 等)である場合には，3 章で述べたような動的解析環境で実行し，悪性か否かを判断することで対応できるようになる．

また，ホワイトリストを用いた自動対処システムでは一度ホワイトリストに入ってしまった正常サイトが乗っ取られてしまった場合は遠隔操作型マルウェアと攻撃者との通信を許すこととなる．この問題を解決するためには，4 章で述べたように，セキュリティベンダの情報を監視し，正規サイトがクラックされた際には当該正規サイトの情報をホワイトリストから削除する必要がある．これにより，当該サイトへの通信に対して追加認証が発生し，遠隔操作型マルウェアの通信が遮断できるようになる．

(キ) ホワイトリスト更新の閾値 k について

本評価実験では，標的型攻撃訓練メールの開封率が 10%であることを参考にして k の値を定めた．k の値は，接続先によらず一定の値としたが，接続先のリスクに応じて変化させてもよい．例えば，5 章で提案したサイトの脅威度推定システムと組み合わせ，接続先の脅威度が低い場合は k の値を小さく，接続先の脅威度が高い場合は k の値を大きくすることで，リスクに応じた柔軟な対応が可能となる．

7.6. まとめ

本章では，ホワイトリストに定められた接続先以外には追加認証を要求することによって，人間による意図的な通信は許可するとともに，その認証結果を用いてホワイトリストの精度を高めていく，ホワイトリスト型の AED を提案し，プロトタイプを開発した．また，評価実験により，遠隔操作型マルウェアの通信を遮断できることを，ユーザ数が少ないうちは，業務に与える影響が高い（追加認証要求率 1.93%）が，ユーザ数が 1,000 人程度になると実用に耐えうることを確認した．以上の結果より，ホワイトリスト型 AED を用いることで，業務への影響を抑制しつつ，遠隔操作型マルウェアの通信を遮断できることを明らかにした．

8. ホワイトリストの最適化

8.1. 導入

7 章でホワイトリストを用いた自動対処システムを提案した．7 章で提案した自動対処システムで解決できていない問題として，初期ホワイトリストの作成方法や，ホワイトリストに登録されたエントリの削除方法が挙げられる．例えば，過去にユーザが接続した実績のある接続先を初期ホワイトリストと定めた場合，すでに侵入している攻撃者による接続が誤ってホワイトリストに含まれてしまう場合がある．また，一度ホワイトリストに含まれてしまった接続先が攻撃者に乗っ取られた場合に 7 章のシステムでは対処できない．これらの問題に対し，本章ではホワイトリストの最適化手法を提案する．なお，ホワイトリストの最適化は，ホワイトリストを用いた対策において共通の問題である．本章では，評価実験の容易さを考慮し，適用先環境の通常動作や定常業務を学習してリスト化するプロファイル型ホワイトリストを対象として，最適化手法を提案する．

8.2. プロファイル型ホワイトリストの課題

8.2.1. プロファイル型ホワイトリストによる対策

プロファイル型ホワイトリストを用いた対策では，システム導入後の一定期間（たとえば一ヶ月）を学習期間とし，この学習期間中に観測された活動（プロセス起動や，内部通信等）は通常業務での活動とみなされ，プロファイル型ホワイトリストに登録される．運用期間中は，プロファイル型ホワイトリストに登録されていない活動を不審活動とみなし，不審活動を検出した際には，管理者へ通知したり，当該活動をブロックしたりする．プロファイル型ホワイトリストを用いたサイバー攻撃検知の例を図 8.1，図 8.2 を用いて説明する．図 8.1 は，遠隔操作型マルウェアを用いた攻撃の流れを例示したものである．まず攻撃者は，標的としている組織に，遠隔操作型マルウェアを添付したメールを送付する．組織のユーザが誤って添付ファイルを実行すると，遠隔操作型マルウェアに感染してしまう．組織に侵入した遠隔操作型マルウェアは，感染端末を操作し，組織内で感染を拡大し，最終的に情報を窃取する．

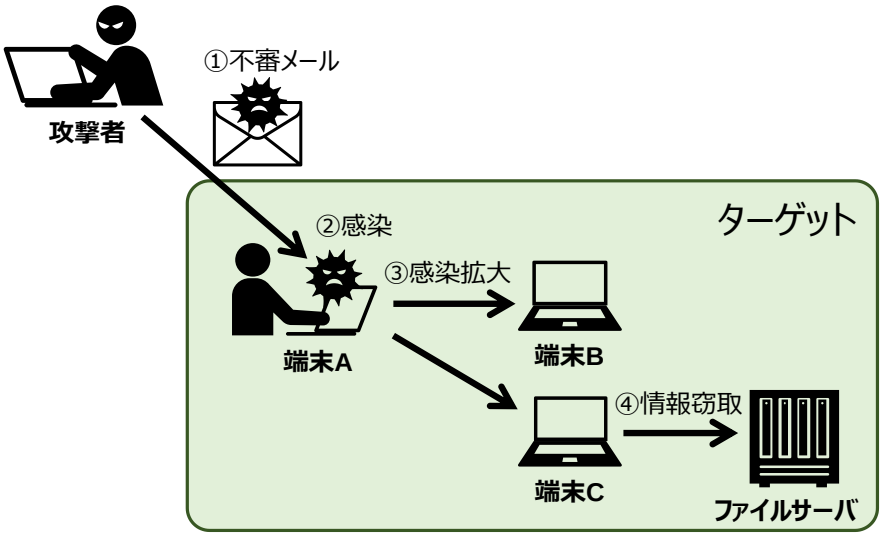


図 8.1　サイバー攻撃の流れ

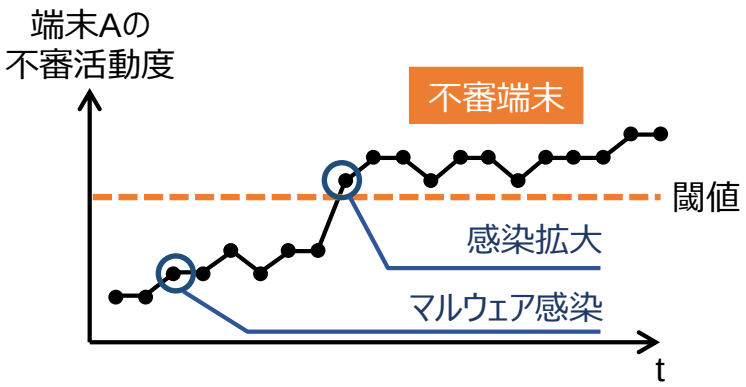


図 8.2　プロファイル型ホワイトリストによる検知

この時，端末 A では，攻撃者の行う活動（マルウェアプロセスの起動や感染拡大によって生じる普段通信を行わない端末への通信）はプロファイル型ホワイトリストに登録されていない活動であるため，端末 A の不審活動度（不審活動の頻度）が上昇する（図 8.2）．不審活動度がある一定の閾値を超えた場合に，当該端末が攻撃者に遠隔操作されているとみなし，管理者へアラートを通知する．

8.2.2. プロファイル型ホワイトリストの課題

プロファイル型ホワイトリストを用いた対策の精度を向上させるためには，当該リストの質が

重要となる．図 8.3 を用いてプロファイル型ホワイトリスト作成の課題を説明する．例えば，既に攻撃者が端末内に侵入していることに気づかずに，学習期間に行われた活動をもとにプロファイル型ホワイトリストを作成することを考える．

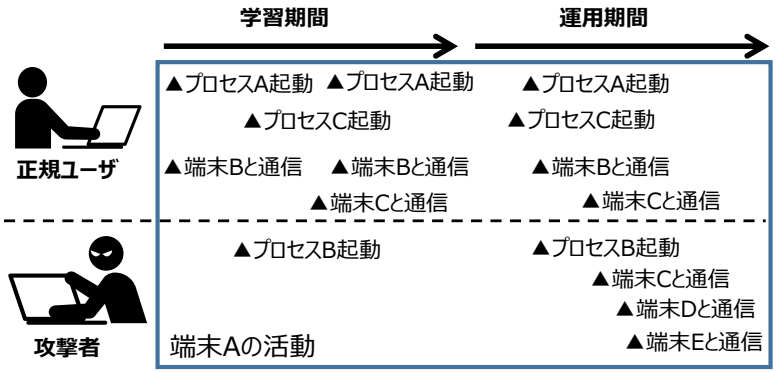


図 8.3　端末 A の活動

プロファイル型ホワイトリストの作成方法として，学習期間に行われた活動すべてを登録した場合，プロファイル型ホワイトリストには，「プロセス A 起動」，「プロセス B 起動」，「プロセス C 起動」，「端末 B と通信」，「端末 C と通信」が登録される．しかし，このプロファイル型ホワイトリストには，学習期間に発生していた攻撃者の活動「プロセス B 起動」が含まれており，当該プロファイル型ホワイトリストを用いて運用を行った場合，運用期間中に発生した攻撃者の活動（「プロセス B 起動」）を不審活動として特定することができない．さらに，運用期間中に発生した攻撃者の活動「端末 C と通信」も，学習期間中の正規ユーザの活動としてプロファイル型ホワイトリストに含まれているため，不審活動として特定することができない．

一方，プロファイル型ホワイトリストに登録する活動を厳選し，例えば「プロセス A 起動」，「端末 B と通信」の 2 つを登録した場合，運用期間中に発生した正規ユーザの活動「プロセス C 起動」を不審活動として特定してしまう．

本章で定義する理想的なプロファイル型ホワイトリストとは，正規ユーザの活動を網羅しつつも，学習期間中の攻撃者の活動や，将来攻撃に利用される活動を排除したものである．図 8.3 の例では，「プロセス A 起動」，「プロセス C 起動」，「端末 B と通信」から構成されるプロファイル型ホワイトリストが，理想的となる．しかし，学習期間中の攻撃者の活動や，将来攻撃に利用される活動を予め特定して排除することは不可能である．

また，プロファイル型ホワイトリスト運用の課題として，プロファイル型ホワイトリストを更新する時期が不明確であることが挙げられる．例えば，正規ユーザが新しいソフトウェアを導入し，利用を開始した場合，プロファイル型ホワイトリストの更新が行われていなければ，当該ソフトウェアを起動するたびに不審活動として特定してしまう．

以上述べたように，プロファイル型ホワイトリストの作成・運用の課題は，大別すると 3 点にま

とめられる.

【課題 1】誤検知が発生する

プロファイル型ホワイトリストに登録する活動を必要以上に制限しすぎた場合，運用期間中に学習に含まれない正規の活動が行われた場合に，不審な活動と検知してしまう.

【課題 2】検知見逃しが発生する

学習期間中に発生した活動を無制限に登録しすぎた場合，運用期間中に発生した攻撃が偶然プロファイル型ホワイトリストに含まれてしまい，検知できない可能性がある（課題 2-1）.

また，学習期間中に攻撃が発生していた場合は，当該攻撃による活動がプロファイル型ホワイトリストに混入してしまい，運用期間中に同様の攻撃が発生した際に，検知できない可能性がある（課題 2-2）.

【課題 3】プロファイル型ホワイトリストの質が劣化する

運用期間中にプロファイル型ホワイトイリストを再学習するべきタイミングがわからず，端末の役割や構成の経年変化に，うまく追随できない.

8.3. プロファイル型ホワイトリスト準最適化手法の提案

8.3.1. 提案手法の概要

上述した課題を解決するため，プロファイル型ホワイトリスト準最適化手法を提案する．各課題へのアプローチ方法を以下に示す.

【課題 1】【課題 2】への対応として，誤検知リスク・検知見逃しリスクの観点からプロファイル型ホワイトリストの質を定量評価する評価指標を策定する（アプローチ 1）．さらに，与えられた活動ログ及び策定した評価指標を基に，最適あるいは準最適なプロファイル型ホワイトリストを，遺伝的アルゴリズムを用いて探索する方式を確立する（アプローチ 2）.

【課題 3】への対応として，策定した評価指標を基に，運用中に再学習が必要なタイミングを判断する手順を確立する（アプローチ 3）.

本手法の全体像を図 8.4 に示す.

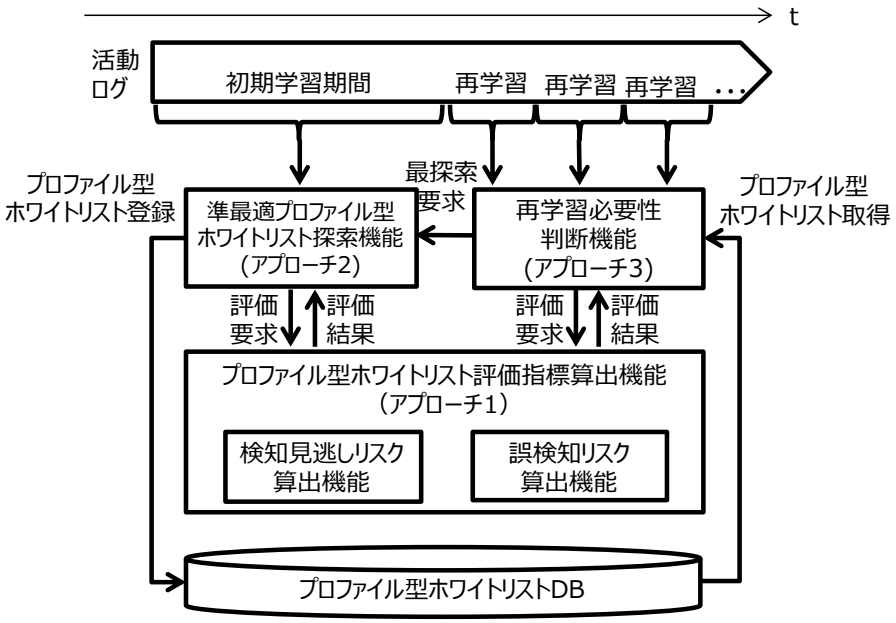


図 8.4　提案手法の全体像

プロファイル型ホワイトリストにおける，誤検知リスクと検知見逃しリスクはトレードオフの関係にある．これは IDS の Precision（適合率）と Recall（再現率）の関係と同様である．このため，IDS の評価では，Precision と Recall の調和平均である F-measure[108]を，両要素を統合した評価指標として用いる．F-measure により，精度・再現率が異なる複数の IDS の中で，どれがより優れたものか判断出来る．（アプローチ 1）においても同様に，統合的な評価指標（WL_{index}）を定めることで，プロファイル型ホワイトリストの質を定量化する．

また，プロファイル型ホワイトリストには数百のエントリが含まれるため，エントリのどのような組合せが，評価指標を最適化するのかは自明ではない．このため，（アプローチ２）では，遺伝的アルゴリズムを用い，ある活動ログが与えられた場合の WL_{index} の値を最適化あるいは準最適化するプロファイル型ホワイトリストを作成する方式を確立する．

理論上，（アプローチ１）（アプローチ２）を用いて運用中に再学習を行うたびに，プロファイル型ホワイトリストは最適化あるいは準最適化される．しかし，機械学習には演算コストが伴うため，不要な再学習の頻発は避けるべきである．このため，（アプローチ３）として，WL_index に基づき，運用中で，再学習が必要なタイミングを判断する手順を確立する．これにより不要な再学習を削減できる．また，多数の端末のプロファイル型ホワイトリストをメンテナンスする必要がある場合は，どの端末のリストから再学習するべきか，優先度付けができる．

8.3.2. 提案手法の詳細

(ア) 評価指標（WL_index）策定

i. 評価指標（WL_index）

ある端末 h のプロファイル型ホワイトリストを作成する時刻を T とし，T までの学習期間中に活動ログ L_h が得られているとする．また，L_h には $C=\{c_1,c_2,\cdots,c_M\}$ の M 種類の活動が記録されている．L_h を基に作成されたプロファイル型ホワイトリスト W_h には，$W_h=\{e_1,e_2,\cdots,e_S\}$ の S 個（$S \leqq M$）のエントリが掲載される．

ここで，Lh からは，学習期間中に発見された攻撃に起因するログは，予め除去されているとする．ただし，潜在的に L_h 内に未発見の攻撃が残存している可能性はある．誤検知リスクを定量化した値を $f(W_h,L_h,T)$，検知見逃しリスクを定量化した値を $g(W_h,L_h,T)$とする．このとき，Wh の WL_index を，トレードオフの関係にある $1\text{-}f(W_h,L_h,T)$と $1\text{-}g(W_h,L_h,T)$の調和平均として表現する．WL_index を定式化すると，

$$\mathrm{index}(W_h, L_h, T) = 1 - \frac{(1+\theta)\cdot\{1-f(W_h, L_h, T)\}\cdot\{1-g(W_h, L_h, T)\}}{\theta\cdot\{1-g(W_h, L_h, T)\}+\{1-f(W_h, L_h, T)\}} \quad (8.1)$$

となる．Index が低い程，プロファイル型ホワイトリストの質が高い．$0 \leq index(),f(),g() \leq 1$ である．なお，θ は，検知見逃しリスクに対して誤検知リスクをどの程度重要視するかを示すパラメータであり，$\theta \geq 0$ である．

ii. 誤検知発生リスク

$f(W_h,L_h,T)$は，エントリが無い空のプロファイル型ホワイトリスト W_{null} と W_h を比較したときの誤検知発生スコアの比として算出する．誤検知発生スコアは，プロファイル型ホワイトリストをある活動ログに適用した場合に発生する誤検知頻度を定量化したものである．即ち，$f(W_h,L_h,T)$は，

$$f(W_h, L_h, T) = \frac{f_s(W_h, L_h, T)}{f_s(W_{null}, L_h, T)} \quad (8.2)$$

として求められる．$f_s(W_h,L_h,T)$は，W_h を L_h に適用した場合の誤検知発生スコアである．

誤検知発生スコアは，拡散活動検知システム[60]のアルゴリズムに習い，ある window 時間（たとえば一時間）の間に発生した，プロファイル型ホワイトリストに含まれない活動のユニーク数に基づくペナルティとして求める．よって，$f_s(W_h,L_h,T)$ を定式化すると

$$f_s(W_h, L_h, T) = \int_0^T p(u(W_h, L_h, t-\mathrm{window}, t), TH_{max}, d)dt \quad (8.3)$$

となる．$u(W_h,L_h,t\text{-}window,t)$は，期間[t-window, t]で発生する活動のうち，W_h に含まれないもののユニーク数を返す関数である．

$p(u,TH_{max},d)$は，関数 u が返すユニーク数に応じたペナルティを求める関数であり，以下のとおり定義される．

$$p(u,TH_{max},d)=\begin{cases} u^d, & u<TH_{max} \\ TH_{max}^d, & u\geq TH_{max} \end{cases} \quad (8.4)$$

d は，ユニーク数に対するペナルティの重み付けであり d≥0 である．TH_{max} は，ペナルティが極端に高い値をとるのを防ぐためのガイドの役割をする閾値である．

ここで，L_h に攻撃に起因する活動群 C_a が含まれており，かつ Wh には登録されていない場合，$fs(W_h,L_h,T)$ は真の値 $fs(W_h,L_h/C_a,T)$ よりも大きな値となる．$fs(W_h,L_h/C_a,T)$を実際に求めるのは不可能である．ここで，

A) L_h と比較して C_a が十分小さい
B) C_a のうち W_h に登録されている要素の発生頻度の比率と，L_h のうち W_h に登録されている要素の発生頻度の比率に大差が無い

の何れかが成り立つのであれば，$f(W_h,L_h/C_a,T)\fallingdotseq f(W_h,L_h,T)$となり，$f(W_h,L_h,T)$は真の値とほぼ等しくなると考えてよい．一般に攻撃の活動量が多い程，事前に発見される機会は多くなる．このため，攻撃が含まれる場合でも(A)が成り立つ蓋然性は高い．また，プロファイル型ホワイトリスト作成方式は C_a 内の要素を区別なく選択するため，(B)が成り立つ蓋然性も高い．本研究では(A)または(B)が成り立つことを前提とする．

iii. 検知見逃しリスク

$g(W_h,L_h,T)$は，すべての C の全要素を含むプロファイル型ホワイトリスト W_{full} と比較したときの，W_h の見逃しスコア $gs(W_h,L_h,T)$の相対値として算出する．即ち，$g(W_h,L_h,T)$は，

$$g(W_h,L_h,T)=\frac{g_s(W_h,L_h,T)}{g_s(W_{full},L_h,T)} \quad (8.5)$$

となる．

W_h の見逃しスコア $g_s(W_h,L_h,T)$は，Wh を適用した場合に発生しうる誤検知頻度を定量化した値である．$g_s(W_h,L_h,T)$はプロファイル型ホワイトリスト内の各エントリ e の見逃しスコア gse(e,T)の総和として算出する．すなわち，

$$g_s(W_h,L_h,T)=\sum_{i=1}^{i=S} g_{se}(e_i,T) \quad (8.6)$$

となる．

ここで，課題 2-1 より，どのようなエントリであっても，運用中に発生した攻撃の一部が偶然一致するリスク（検知見逃しリスク I）は必ず存在する．よってリスク値は一様に 1 と設定する．

ただし，本来は悪意ある活動でなく正常業務でも発生するが，標的型サイバー攻撃に悪用されやすい活動も存在する．例えば JPCERT/CC では，標的型攻撃に悪用されやすい Windows コマンドを公表している[109]．そこで，外部のナレッジを基に，標的型攻撃に悪用されやすい活動群 Bset を収集し，これに含まれる活動に対しては例外的なリスク値を付与する．

一方で，課題 2-2 に対しては，一般に，作成日時（即ち T）の直前に発生した活動をプロファイル型ホワイトリストに加えるのは望ましくない．これは，未だ発見されていない攻撃に起因する活動がリストに混入する恐れがあるためである．例えば，Verizon のレポート[110][111]では，標的型攻撃が発生してから発見されるまでに，ワーストケースでは数ヶ月程度を要するとされている．そこで，ある活動が実際には攻撃に起因するものであるリスク（検知見逃しリスク II）を，活動の初観測日時と T の時間差に半比例すると仮定する．これは，攻撃発生から時間が経過するほど，何らかの手段で攻撃を発見する機会が多くなるためである．リスト作成前に攻撃が発見されれば，攻撃に起因する活動はリストの候補から外すことができる．

WL_{index} では，課題 2 に起因するリスクを，Bset 及び T と初観測日時の差を引数する減衰関数で表現する．1 日前に新規観測された活動をプロファイル型ホワイトリストに登録するリスク値を R, 30 日後のリスク値を 1，Bset に含まれる活動へのリスク値を RB とすると，検知見逃しリスク I 及び II を統合したリスク値 gse (ei,T)は

$$g_{se}(e_i,T)=\begin{cases} R^B, & e_i \in B_{set} \\ R^{\frac{30-\left(T-e_i^{first}\right)}{29}}, & e_i \notin B_{set}, T-e_i^{first}<30\text{days} \\ 1, & e_i \notin B_{set}, T-e_i^{first}\geq 30\text{days} \end{cases} \quad (8.7)$$

として算出できる．なお，e_i^{first} は，エントリ内の活動が最初に発生した時刻を示す．以上より，任意のプロファイル型ホワイトリストについて WL_{index} を計算できる．

(イ) 準最適プロファイル型ホワイトリスト作成

L_h からプロファイル型ホワイトリストを作成する場合，考えられる組み合わせは 2M-1 通りある．通信・プロセス起動を監視対象の活動とした場合，実際には M>100 となるため，全ての組み合わせを試行して，WL_{index} が最小となるプロファイル型ホワイトリストを特定するのは計算量の点で困難である．

そこで遺伝的アルゴリズム[112]を用いて WL_{index} を最小化するプロファイル型ホワイトリストの近似解を探索する方式を検討する．遺伝的アルゴリズムは，ナップサック問題・巡回セールスマン問題など，多項式時間で最適解を算出するのが難しい NP 困難な問題に対して，近似解（準最適解）を導くことができる．

遺伝的アルゴリズムでは，データ（解の候補）を，複数の「遺伝子」で構成される「染色

体」として表現する．そして，複数のランダムな染色体をベースに，任意の評価関数において評価値が高い個体を優先的に選択して，交叉（組み換え）・突然変異などの遺伝子操作を繰り返し世代交代しながら，解（＝評価値が高い染色体）を探索する．

提案手法では，ある活動 c_i をプロファイル型ホワイトリストに含めるか否かを，遺伝子 di で表現する．d_i=1 なら c_i はリストに含まれ，d_i=0 なら ci はリストに含まれない．プロファイル型ホワイトリストに含まれる活動の組合せは，染色体 D={d_1,d_2,…,d_M}として扱う．染色体の遺伝子操作手順としては，選択・交差・突然変異を含む一般的なアプローチを用いる．

図 8.5 に，プロファイル型ホワイトリスト作成方法の擬似コード（generateWhiteList）を示す．まず，関数 initialize は，乱数を基に一定数（例えば 500）の初期染色体のセットを作成する．関数 G は D_n^g,L_h で指定されたプロファイル型ホワイトリストを作成する．次に，現世代で最も WL_{index} の評価値が低い染色体を選択する．P 世代前と比べて WL_{index} に大きな差が無い場合，これ以上の進化は無いと考え，アルゴリズムを打ち切る．アルゴリズムが打ち切られない場合，関数 computeNextGeneration は，現世代を基に，選択・交叉・突然変異により次世代の染色体を作成し，探索を継続する．

```
1   generateWhiteList(L_h, T){
2     //initialize the variable for current generation
3     g=0
4     //the number of seeds
5     N=500
6     //generate N initial seeds for 0^th generation
7     D^0={D_1^0, D_2^0, ..., D_N^0}=initialize(L_h, N)
8
9     While( true ){
10      //function G returns a white list based on D_n^g, L_h
11      //The following two lines of codes find the best
12      //WL_index among the current population
13      best^g = min_{1≤n≤N} index(G(D_n^g, L_h), L_h, T)
14      S = arg min_{1≤n≤N} index(G(D_n^g, L_h), L_h, T)
15
16      //if the index is not improved compared with
17      //the past P generation, return the best value
18      if(best^{g-P} − best^g ≤ δ)
19        return G(D_s^g, L_h) as W_h
20
21      //compute the next generation from the current
22      //population by using selection, cross over
23      //and mutation
24      D^{g+1}=computeNextGeneration(D^g)
25      g++
26    }
27  }
```

図 8.5　プロファイル型ホワイトリスト作成方法の擬似コード

(ウ) 再学習必要性判断

ここで，W_h 作成後の再学習手順について述べる．本手順では，プロファイル型ホワイトリストを作成した時刻 T 以後，インターバル I(例えば I=7days)ごとに，経年変化に伴う W_h の質の変化 Q を評価する．Q がある閾値より高い場合，プロファイル型ホワイトリスト作成後，端末 h の挙動に大きな変化があったと判断して，W_h を再学習対象とする．

時刻 T から，k 回のインターバル経った時刻 T^k (=T+kI)までの期間に発生したログを L_h^k と表記する．このとき，$Q(W_h,L_h^k,T^k)$は

$$Q\left(W_h, L_h^k, T^K\right) = \frac{\mathrm{index}(W_h, L_h^k, T^k)}{\mathrm{index}(W_h, L_h^1, T^1)} \quad (8.8)$$

と評価される．T1 はプロファイル型ホワイトリストを作成してから最初のインターバルである．当然，$Q(W_h,L_h^1,T^1)=1$ となる．Q の値域は，Q>0 である．Q は時間軸で必ずしも単調増加ではない．たとえば，運用中，全く誤検知が発生しない場合，$T^k>T^{k-1}$ より，

$$\mathrm{index}\left(W_h, L_h^k, T^k\right) < \mathrm{index}\left(W_h, L_h^{k-1}, T^{k-1}\right) \quad (8.9)$$

となる．

そして，閾値 Q_{TH} に対して

$$Q\left(W_h, L_h^k, T^K\right) \geq Q_{TH} \quad (8.10)$$

が成り立つとき，プロファイル型ホワイトリスト Wh を更新する．

8.4. 評価実験

8.4.1. 実験概要

提案方式の評価実験について述べる．実験では，ある組織内ネットワークに設置された同一事業部に属する PC から活動ログを取得し，評価指標及びプロファイル型ホワイトリスト作成方式について評価した．学習用のログは，実験に参加している 26 台の端末の通信・プロセス起動ログであり，学習期間は 2017 年 6 月 16 日から 2017 年 7 月 10 日までである．またプロファイル型ホワイトリスト作成日時は 2017 年 7 月 11 日の 00:00:00 とした．

評価では，2017 年 7 月 11 日から 2017 年 8 月 7 日までの運用期間に際して，誤検知が発生する期間（誤検知期間）を測定する．誤検知期間は，運用期間中で，プロファイル型ホワイトリストに含まれないユニークな活動が window 時間当たり閾値 TH 個以上発生する期間の長さである．学習期間・評価期間を合わせ，活動ログの総レコード数は 3,652,514 件である．

表 8.1 に，評価に用いたパラメータのデフォルト値を，表 8.2 に遺伝的アルゴリズムのパラメータを示す．

表 8.1　パラメータのデフォルト値

パラメータ	デフォルト値
TH	4
θ	10
d	1
Bset	φ
THmax	10
R	10
Window	1hour
I	7days

表 8.2　遺伝的アルゴリズムのパラメータ

パラメータ	値
交叉率	0.65
突然変異率	0.1
染色体の数	500
種を保存する割合	0.1

WL_{index} およびプロファイル型ホワイトリスト作成方式の性能評価のために，それぞれについて，比較対象方式を設定する．

WL_{index} の比較対象とする評価指標(alt_{index})では，個々の活動の発生回数と発生時刻を基にプロファイル型ホワイトリストの質を評価する．具体的には，誤検知発生リスク $f(W_h,L_h,T)$の代わりに，W_h 内の活動が L_h で出現する頻度 $r(W_h,L_h)$を用いる．定式化すると以下のとおりとなる．関数 count は，引数で与えられたエントリの出現回数を求める．

$$alt_{index(W_h,L_h,T)} = 1 - \frac{(1+\delta)\cdot r(W_h,L_h)\cdot\{1-g(W_h,L_h,T)\}}{\delta\cdot\{1-g(W_h,L_h,T)\}+r(W_h,L_h)} \quad (8.11)$$

$$r(W_h,L_h) = \frac{\sum_{e\in W_h} count(e,L_h)}{|L_h|} \quad (8.12)$$

WL_{index} と alt_{index} の比較を通し，プロファイル型ホワイトリストに含まれない活動の発生頻度を考慮することで，WL_{index} が相対的に優れた評価指標となることを示す．

プロファイル型ホワイトリスト作成方式の比較対象としては，遺伝的アルゴリズムに依らず発生頻度を基にプロファイル型ホワイトリストを作成する貪欲法（greedy 方式）を用いる．greedy 方式では，検知見逃しリスクが特定の値 M を超えない範囲で，学習期間中の発生頻度が多い活動

から順番にプロファイル型ホワイトリスト W’$_h$ に登録する．提案方式と greedy 方式の比較を通し，遺伝的アルゴリズムを用いることで相対的に優れたプロファイル型ホワイトリストを作成できることを示す．

評価用プログラムは全て Java JDK1.7 で実装した．ステップ数は約 2000 である．また評価に用いた PC のスペックは，OS=Windows 7 64bit Home Premium，CPU=Corei7-2600，Memory=4GB である．

8.4.2. 評価結果

(ア) 学習コスト

図 8.6 に，遺伝的アルゴリズムを適用したときの，ある端末の WL_{index} の学習状況を示す．30 世代程度で値がほぼ収束していることがわかる．

表 8.3 に，W_h の作成及び $Q(W_h,L_{h1},T^1)$の算出にかかる所要時間を示す．ただし，演算開始時点で，活動ログおよびプロファイル型ホワイトリストのデータがメインメモリ上にあることを前提とする．W_h の作成には約 85 秒，$Q(W_h,L_{h1},T^1)$の算出には約 0.2 秒かかっており，Q 値算出の方が約 400 倍速いことがわかる．仮に 1 万台の端末のプロファイル型ホワイトリストを管理する場合，全端末のプロファイル型ホワイトリストの作成には約 236 時間かかる．一方，Q 値算出にかかる時間は 30 分程度である．このため，前述のとおり，再学習時には，一度に全プロファイル型ホワイトリストを作成するのではなく先ず Q 値を求めたうえで必要に応じて作成するのが好ましいといえる．

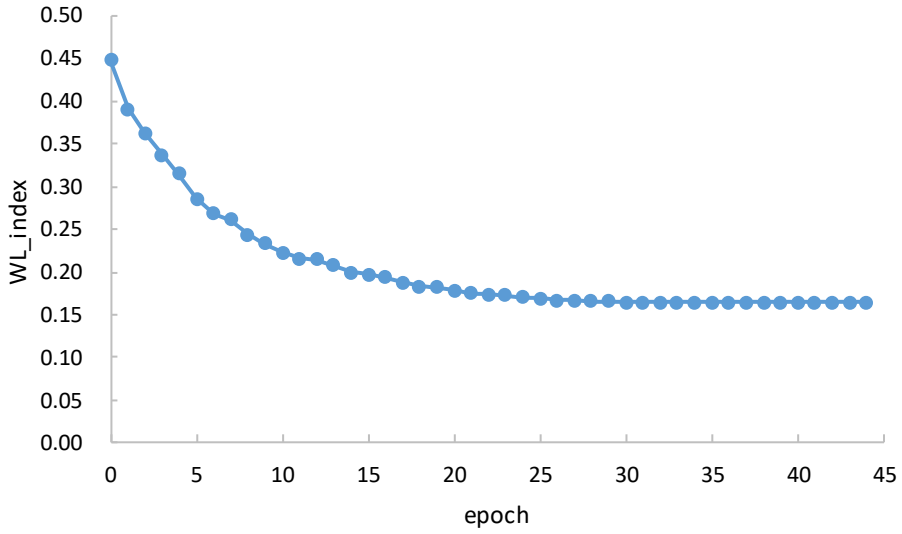


図 8.6　WL_index の更新過程

表 8.3　プロファイル型ホワイトリスト作成及び Q 値算出の所要時間

	プロファイル型ホワイトリスト 作成時間[msec]	Q 値 作成時間[msec]
平均	85,402.58	198.27
標準偏差	22,935.54	42.99

(イ) 誤検知と検知見逃しリスクの関係

図 8.7 に，θ を 1 から 1000 まで変化させた場合の，W_h の誤検知期間と検知見逃しリスクのトレードオフを示す．各値は 26 端末の平均値である．θ を大きくする，すなわち誤検知を減らす方向に WL_{index} をチューニングするほど，誤検知期間は低減し，検知見逃しリスクは増加する．

alt_{index} を基にした W'_h と比較すると，W_h の性能曲線はより原点側に位置する．即ち，検知見逃しリスクが同一である W_h と W'_h がある場合，W_h の方が誤検知期間は短くなる．以上より，運用中の誤検知期間を短縮するという目的に対しては，WL_{index} は alt_{index} に比べて優れた評価指標といえる．

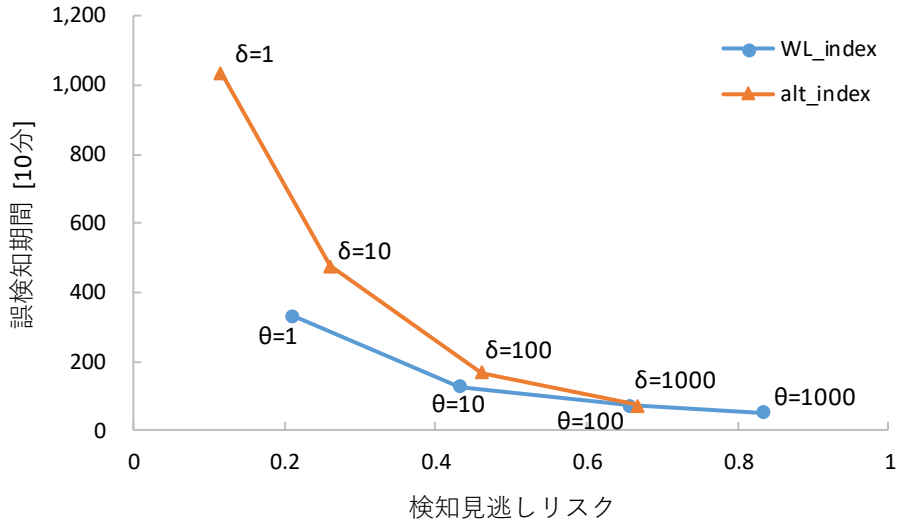


図 8.7　指標評価の比較

図 8.8 に，提案方式で作成した Wh と greedy 方式で作成した W'_h との比較を示す．両方式とも評価指標としては WL_{index} を用いている．提案方式のカーブは greedy 方式と比べて常に原点側に位置する．このことは，遺伝的アルゴリズムの活用で，greedy 方式と比べ高性能なプロファイル型ホワイトリストを作成できることを意味する．具体的には，同一の検知見逃しリスクに対する誤検知期間を θ =1 のとき 20%，θ =10（デフォルト値）のとき 15%削減で

きる．

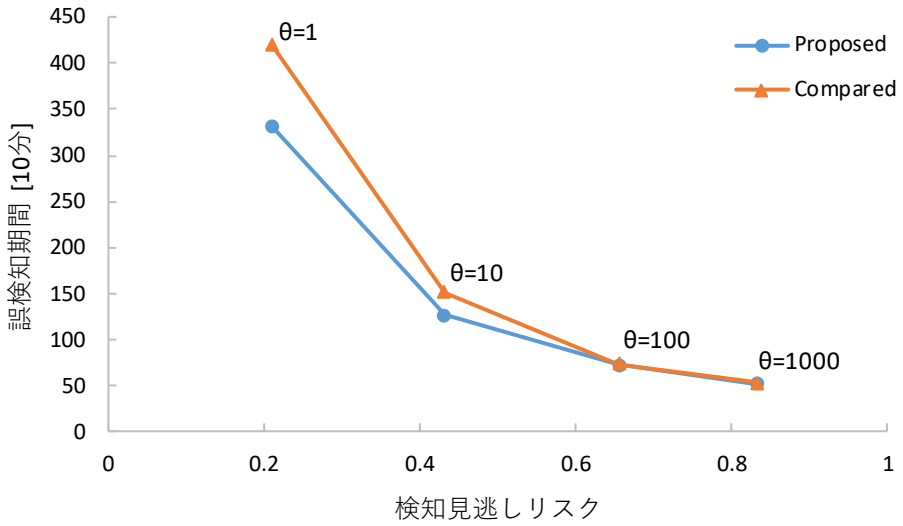


図 8.8　greedy 方式との比較

図 8.9 に，d を 0 から 2 まで変化させた場合のトレードオフを示す．d を大きくするほど，検知見逃しリスクの低減を優先するプロファイル型ホワイトリストが作成される．一方で，d を 0.5 より低くすると，提案手法では，検知見逃しリスク・誤検知期間が両方とも悪化する．このため，d は 0.5 以上の値を設定することが望ましい．

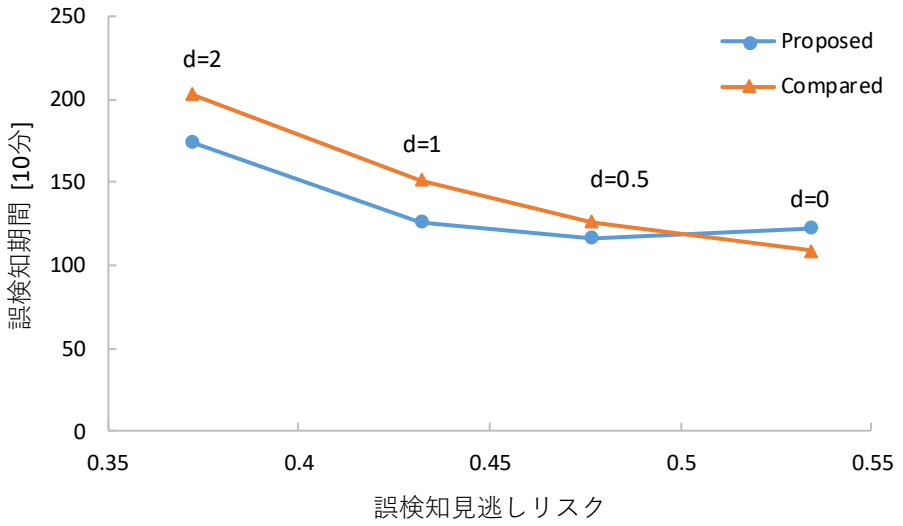


図 8.9　d の影響

図 8.10 に，TH_{max} を 4 から 100 まで変化させた場合のトレードオフを示す．TH_{max} を大きくするほど，検知見逃しリスクの低減を優先する Wh が作成される．一方，TH_{max} を 4，すなわち TH と同値にした場合，検知見逃しリスク・誤検知期間が両方とも悪化する．このため，TH_{max} は TH より大きい値を設定することが望ましい．

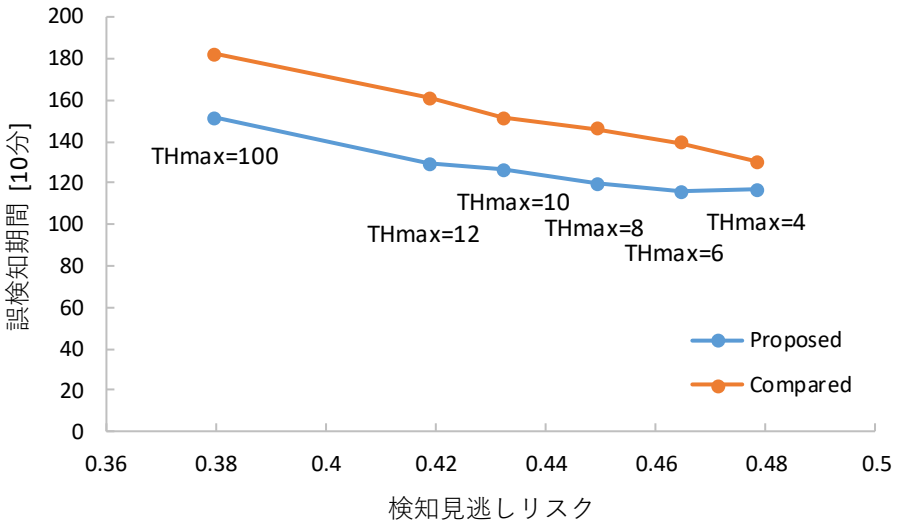


図 8.10　THmax の影響

(ウ) リスク値の性能への影響

図 8.11 に R を 1 から 100 まで変化させた場合の，検知見逃しリスク $g(W_h,L_h,T)$と誤検知期間の関係を示す．R が大きいほど，時刻 T の直前で初めて記録された活動をプロファイル型ホワイトリストに登録することのリスクは高く評価される．

図 8.11 からは，R が大きくなるほど，検知見逃しリスク g，誤検知期間は共に改善するように見える．前述のように，検知見逃しリスク g は，検知見逃しスコア $g_s(W_{full},L_h,T)$に対する $g_s(W_h,L_h,T)$の比率である．時刻 T までの全て活動を含む $g_s(W_{full},L_h,T)$は R に比例して大きくなる．一方，$g_s(W_h,L_h,T)$は R_n $(n\leq 1)$に比例して大きくなる．このため$g \propto R^m (m \leq 0)$となり，$\lim_{R\to\infty} g = 0$が成り立つ．

一方，見逃しスコア gs を横軸とした場合の，誤検知期間との関係は図 8.12 の示すとおりになる．R が大きくなるほど，gs は大きくなり，誤検知期間は増加するというトレードオフの関係が現れる．

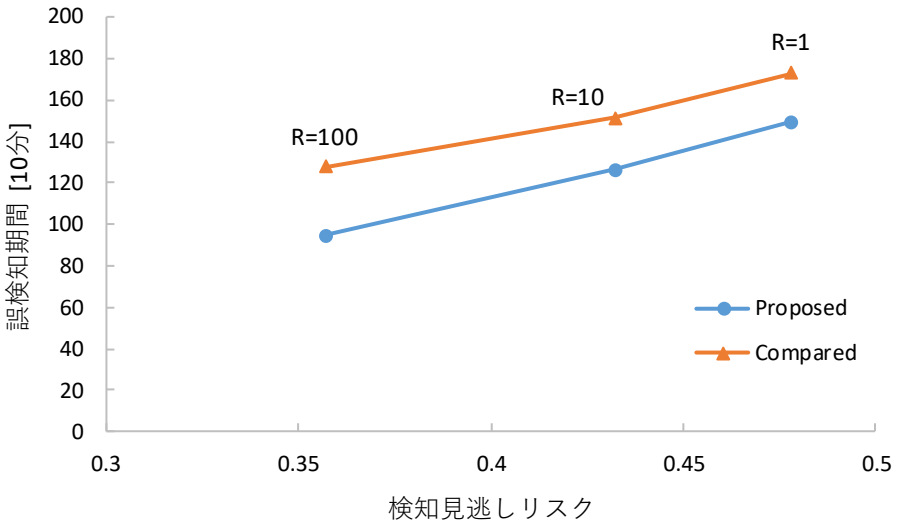


図 8.11　R が変化した場合の検知見逃しリスクと誤検知期間の関係

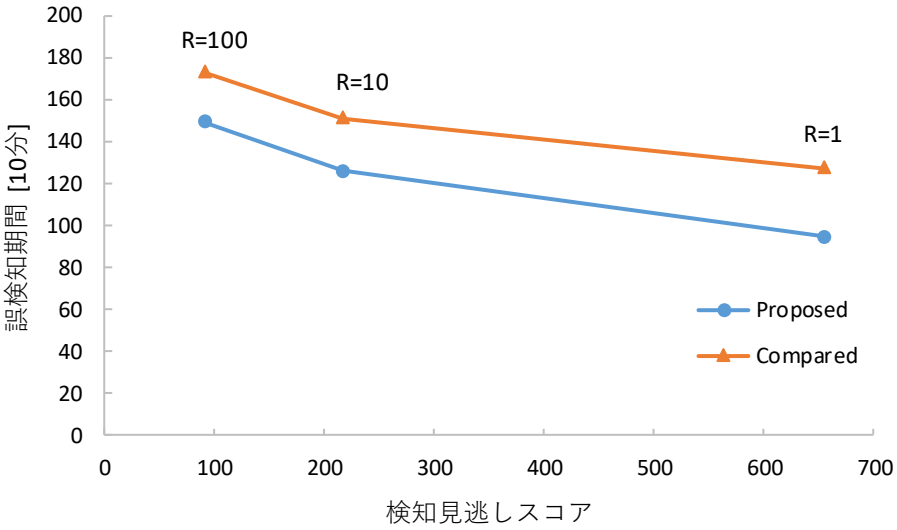


図 8.12　R が変化した場合の検知見逃しスコアと誤検知期間の関係

図 8.13 に，JPCERT/CC が公表している標的型サイバー攻撃に悪用されやすい Windows コマンド 18 種類を Bset に登録した場合の性能評価を示す．提案方式では，図 8.7 の場合と同様に，RB の増加に伴い検知見逃しリスク及び誤検知期間は共に低減する．一方，greedy 方式では，誤検知期間が大幅に増加する．これは，発生頻度が多い Windows コマンドのリスクが増えるに従い，指定された検知見逃しリスク値を超えない範囲で，プロファイル型ホワイトリストに含めることができる活動数が減少するためと考えられる．

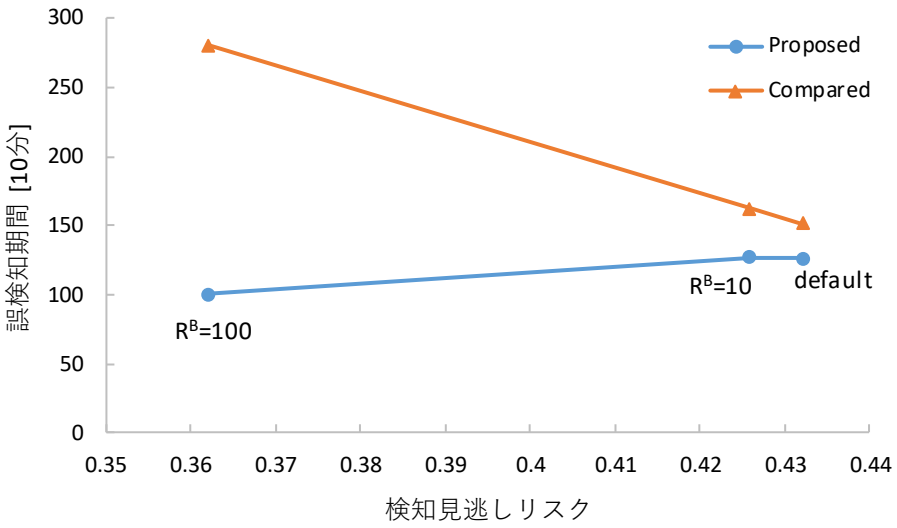


図 8.13　RB の影響

(エ) プロファイル型ホワイトリストの経年変化と再学習

Tk における，Q 値の平均・分散は表 8.4 の示すとおりになる．約一か月が経過しても，Q 値の平均はほぼ 1 のままである．このため，評価実験で得られた範囲では，再学習が必要な端末は少数といえる．ただし，時間経過と共に標準偏差は増加傾向にあり，今後再学習が必要な端末は増加していくと考えられる．

図 8.14 に，Q_{TH} と再学習対象端末の割合を示す．仮に，Q_{TH}=1.2，即ち質が 20%以上悪化したプロファイル型ホワイトリストを再学習対象とすると，T4 までに再学習対象は全体の 15%となる．たとえば，総端末数を 10,000 台とすると，総学習時間は約 36 時間となる．これは，業務が比較的少ない土日の 48 時間以内に再学習を完了できることを意味する．

Q_{TH} の妥当性については今後長期的な実験により検証を進める必要がある．しかし，評価実験で明らかになった範囲では，本報告で提案したプロファイル型ホワイトリスト評価指標・作成方式・再学習手順は，再学習による演算負荷を低減し，多数の端末に対する効率的なプロファイル型ホワイトリスト運用を実現する見込みがあるといえる．

表 8.4　Q 値の平均及び標準偏差

	T1	T2	T3	T4
平均	1	1.020	1.058	1.044
標準偏差	0	0.142	0.213	0.239

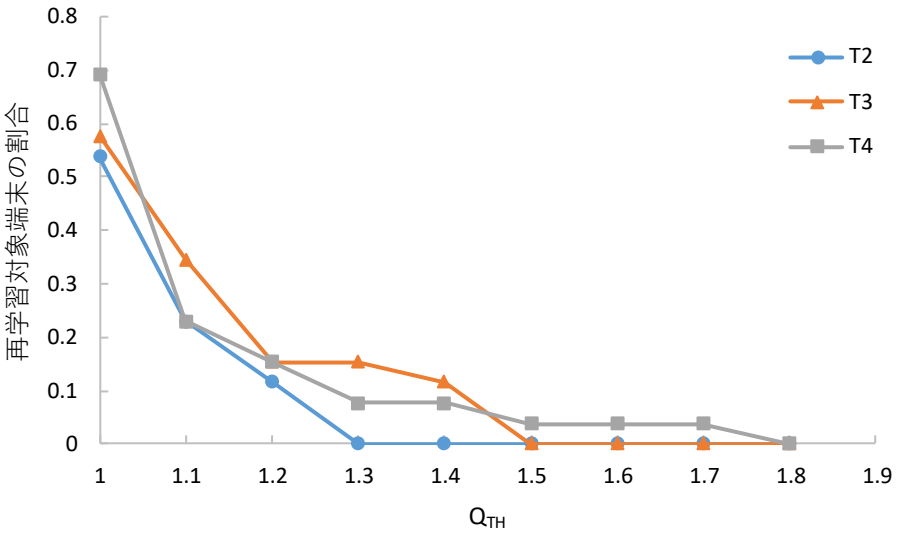


図 8.14　Q_{TH} と再学習対象端末の割合

8.5. 考察

WL_index には 7 つのパラメータ，θ，d，TH_{max}，R，R_B，window，TH が存在する．本節では，各パラメータの設定方法について検討する．

θ は，原則的には，誤検知と検知見逃しのどちらに重みを置くかにより決定される．考え方の 1 つとして，運用面などから誤検知期間に制約を設け，制約内で最適な θ を設定する，というものがある．たとえば，誤検知期間の比率を全端末平均 3%以内に抑えることを制約とした場合，図 8.7 より，本実験では θ =10 が望ましい値となる（実験期間 1 か月・720 時間の 3%は，約 1300 分であるため）．また，各端末の誤検知リスクスコアを一定値以下に抑えたい場合は，端末ごとに異なる θ を採用すればよい．

図 8.9 より，d は θ と比べると，値を変えた場合の誤検知期間の変動が小さい．特に 0<d≤1 の範囲では変動が殆どないため，検知見逃しリスクを最小化するという点で d=1 程度が望ましいと考えられる．

図 8.10 より，TH_{max} も誤検知期間への大きな影響はない．このため，概ね，TH 以上の値を設定すれば良いと考える．

R の設定値は，プロファイル型ホワイトリスト作成 1 日前に観測された活動のリスクを，最小リスク(=1)の何倍程度と捉えるべきかに依存する．リスク評価手法である CVSS[113]ではリスクの最大値と最小値の差は 10 倍である．このため，デフォルトでは，R=10 程度とするのが妥当と考える．R_B についても同様である．

TH 及び window は，拡散活動検知システム[60]の検知パラメータに依存する．デフォルトでは TH=4, window=1hr が望ましいと考える．

8.6. まとめ

本章では，プロファイル型ホワイトリストの性能向上を目的に，ホワイトリスト最適化技術を検討・評価した．具体的には，IDS の評価指標である F-measure を参考に，誤検知及び検知見逃しリスクの観点から，プロファイル型ホワイトリストの質を定量評価する評価指標を策定した．また，遺伝的アルゴリズムを用いて，与えられた活動ログを基に評価指標の点で準最適なプロファイル型ホワイトリストを作成する方式を考案した．さらに，運用期間中にプロファイル型ホワイトリストの再学習が必要なタイミングを判断する手順を検討した．

ある組織の 26 端末の活動ログを用い，本技術は誤検知・検知見逃しの両点で比較対象方式より 15%以上優れたプロファイル型ホワイトリストを作成できることが明らかになった．また，本技術により，低負荷で，再学習タイミングを判断できることを確認すると共に，10,000 台の端末に対するプロファイル型ホワイトリスト運用を実現できる見込みを示した．

評価実験では，約 2 か月という短期間の観測結果を使用して評価を行っている．今後は，半年から一年以上の長期的な観測結果を用いて，提案指標及び提案方式の有用性を検証する必要がある．特に再学習手順に関しては，観測結果を基に適宜改善を行っていく．

9. 結論

本論文ではサイバー攻撃に迅速対応するために，自動対処システムの提案を行った．自動対処を実現するにあたっては，即時性のある脅威情報の入手困難性（問題１）と，不確実な脅威情報に起因する業務悪影響懸念（問題２）が存在することを明らかにした．前者を解決するために，マルウェア解析の高度化と公開情報の構造化により，攻撃者による遠隔操作や情報搾取を防止するために有用となる脅威情報を抽出する目的１と，後者を解決するために業務への悪影響を抑えつつ，マルウェアによる被害発生リスクを軽減する目的２の実現を目指した．

目的１を達成するために，マルウェア解析の高度化（課題１）と，公開されている脅威情報の構造化（課題２）を設定し，目的２を達成するために，脅威情報の高精度化断（課題３）と，業務に影響を与えずにマルウェアの通信を遮断すること（課題４）をそれぞれ設定した．

課題１では，多種解析環境におけるマルウェアの動的解析結果を用いた分類手法を提案した．本手法は，解析対象のマルウェアを類似する既存のマルウェアファミリに分類するものであり，これにより，亜種マルウェアの選別や既存のマルウェアファミリに分類されない新種のマルウェア抽出に応用できることを確認した. また, プロトタイプを実装し, 既知のマルウェアを約 64.80%の正解率で分類できること，各マルウェアファミリへの所属確率を利用することによって，新種マルウェアを抽出でき得ることを示した．

課題２では，未知語を考慮した NER によるインテリジェンス構造化手法について提案した．提案手法は，ベースラインとして，文字の CNN による表現と単語の埋め込み表現を結合したベクトルを入力とし，Bi-LSTM 層，結合層，および CRF 層によって NER を行うモデルを用いることにより，インテリジェンスとして着目すべき固有表現を抽出する．この際，CRF 層で算出した各語の条件付確率に着目することにより，セキュリティ分野では特に頻出な未知語に対する見逃し率を抑制し，認識精度を向上させられることを確認した．また，提案手法のプロトタイプを実装し，ベースラインの手法よりも高精度で固有表現を認識できることと従来手法では見逃していた未知語を認識できていることを確認した．さらに，モデルの再学習時間や１記事に対する NER に要する時間を計測し，実運用にも耐えうることを示した．

課題３では，複数の脅威度推定器を利用し，アクセス先の最終的な脅威度を予測する手法を提案した．良性サンプル 50,000 件と悪性サンプル 50,000 件を用いて，提案手法の脅威度推定精度を検証し，サイトの性質を最大 92.74%の精度で推定できることを確認した．

課題４では，組織における過去の接続ログから脅威情報を対処した際の影響を算出し，算出した影響度に基づき自動対処を行うシステムを提案した．また，プロキシと連携したプロトタイプ

を用いた評価実験により，464 秒でマルウェアの動的解析から対処適用までが完了すること，実マルウェアから 508 件の脅威情報を抽出し，そのうちの 491 件を自動対処できることを確認した．

さらに，まだ世の中に知られていない未知のサイトをマルウェアが利用した場合にも対処できるようにするため，ホワイトリストに定められた接続先以外には追加認証を要求することによって，人間による意図的な通信は許可するとともに，その認証結果を用いてホワイトリストの精度を高めていく，ホワイトリスト型の AED を提案した．評価実験により，遠隔操作型マルウェアの通信を遮断できることを，ユーザ数が少ないうちは，業務に与える影響が高い（追加認証要求率 1.93%）が，ユーザ数が 1,000 人程度になると実用に耐えうることを確認した．

これらの４つの課題を解決するための手法提案，実装，実験評価を通し，マルウェア解析の高度化と公開情報からの脅威情報収集によって目的１を達成し，また，業務影響を考慮した対策適用により目的２を達成した．

以上，本論文では目的１，２を技術によって解決する手法を提案した．

参考文献

[1] IPA: 標的型攻撃/新しいタイプの攻撃の実態と対策, (http://www.ipa.go.jp/files/000024542.pdf, 2017 年 6 月参照)

[2] FireEye: FireEye Threat Intelligence, (https://www.fireeye.jp/content/dam/fireeye-www/regional/ja_JP/products/pdfs/ds-threat-intelligence.pdf, 2017 年 6 月参照)

[3] ラック: 「日本年金機構の情報漏えい事件から得られる教訓」公開のお知らせ, (http://www.lac.co.jp/news/2015/06/09_news_01.html, 2017 年 6 月参照)

[4] JPCERT: 標的型攻撃への対応- JPCERT/CC , (http://www.jpcert.or.jp/present/2015/JNSAWG20150625-apt.pdf, 2017 年 6 月参照)

[5] シマンテック: Backdoor.Emdivi, (https://www.symantec.com/security_response/writeup.jsp?docid=2014-101715-1341-99, 2017 年 6 月参照)

[6] 仲小路博史, 藤井康広, 磯部義明, 重本倫宏, 鬼頭哲郎, 林直樹, 川口信隆, 下間直樹, 菊池浩明: 人間行動を用いた自律進化型防御システムの提案, 暗号と情報セキュリティシンポジウム 2016 (SCIS2016), pp 1-8(2016).

[7] H. Nakakoji, Y. Fujii, Y. Isobe, T. Shigemoto, T. Kito, N. Hayashi, N. Kawaguchi, N. Shimotsuma, H. Kikuchi, Proposal and Evaluation of Cyber Defense System Using Black list Refined Based on Authentication Results, 2016 19th International Conference on Network-Based Information Systems (NBiS), Ostrava, 2016, pp. 135-139.

[8] D. Plohmann, K. Yakdan, M. Klatt, J. Bader, E. Gerhards-Padilla, A Comprehensive Measurement Study of Domain Generating Malware, 25th USENIX Security Symposium (USENIX Security 16), Austin, 2016, pp. 263-278.

[9] The Guardian: Antivirus software is dead, says security expert at Symantec, (https://www.theguardian.com/technology/2014/may/06/antivirus-software-fails-catch-attacks-security-expert-symantec, 2017 年 8 月参照).

[10] National Institute of Standards and Technology: SP 800-61 Rev. 2 Computer Security Incident Handling Guide, (https://csrc.nist.gov/publications/detail/sp/800-61/rev-2/final, 2018 年 7 月参照)

[11] National Institute of Standards and Technology: National Vulnerability Database, (https://nvd.nist.gov/, 2018 年 7 月参照)

[12] Japan Vulnerability Notes, (https://jvndb.jvn.jp/, 2018 年 7 月参照)

[13] R., A., Bridges, C., L., Jones, M., D., Iannacone, K., M., Testa, and J., R., Goodall: Automatic labeling for entity extraction in cyber security, ASE Third International Conference on Cyber Security, Academy of Science and Engineering (ASE), (2014).

[14] N., McNeil, R., A., Bridges, M., D., Iannacone, B., Czejdo, N., Perez, and J., R., Goodall: PACE: Pattern Accurate Computationally Efficient Bootstrapping for Timely Discovery

of Cyber-security Concepts, 12th International Conference on Machine Learning and Applications, pp. 60－65 (2013).

[15] 仲小路博史, 重本倫宏, 鬼頭哲郎, 林直樹, 寺田真敏, 菊池浩明. 多種環境マルウェア動的解析システムの提案および評価. 情報処理学会論文誌, vol. 56, No. 9, pp. 1730-1744, 2015.

[16] 重本倫宏, 徳山喜一, 下間直樹, 鬼頭哲郎, 仲小路博史. マルウェア解析向け通信制御システムの開発. 情報処理学会論文誌, vol. 57, No. 9, pp. 2012-2020, 2016.

[17] L., Obrst, P., Chase, and R., Markeloff: Developing an Ontology of the Cyber Security Domain, CEUR Workshop Proceedings, Vol. 96, pp.49－56 (2012).

[18] 淵上智史, 長谷川皓一, 山口由紀子, 嶋田創, 高倉弘喜. マルウェア感染拡大抑止に向けたネットワーク型動的解析システム. 研究報告インターネットと運用技術(IOT). vol. 2016-IOT-32, no. 36, pp. 1-6, 2016.

[19] Digital Arts Inc.: Web プロキシ機能, (http://www.daj.jp/bs/i-filter/function/proxy/, 2017 年 6 月参照)

[20] Anubis: Malware analysis for unknown binaries, (https://anubis.iseclab.org/, 2016 年 2 月参照)

[21] A., Dinaburg, P., Royal, M., Sharif, and W., Lee: Ether: Malware analysis via hardware virtualization extensions, in Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS '08), pp. 51–62 (2008).

[22] C., Willems, T., Holz, and F., Freiling: Toward automated dynamic malware analysis using CWSandbox, Security Privacy, IEEE, Vol. 5, no. 2, pp. 32–39 (2007).

[23] U., Bayer, C., Kruegel, and E., Kirda: TTAnalyze: A tool for analyzing malware, in Proceedings of the 15th European Institute for Computer Antivirus Research Annual Conference (EICAR), (2006).

[24] H., Yin, D., Song, M., Egele, C., Kruegel, and E., Kirda, Panorama: Capturing system-wide information flow for malware detection and analysis, in Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS '07), pp. 116–127 (2007).

[25] Dhilung, K., Giovanni, V., Christopher, K.: BareCloud: Bare-metal Analysis-based Evasive Malware Detection, 23rd USENIX Security Symposium (USENIX Security 14), 287－301 (2014).

[26] Shafiq, M., Z., Tabish, S., M., Mirza, F., and Farooq, M.: PE-Miner: Mining Structural Information to Detect Malicious Executables in Realtime, Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection (RAID '09), pp.121－141 (2009).

[27] Kolter, J., Z., and Maloof, M., A.: Learning to Detect Malicious Executables in the Wild,

Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04), pp.470－478 (2004).

[28] Ahmed, F., Hameed, H., Shafiq, M., Z., and Farooq, M.: Using Spatio-temporal Information in API Calls with Machine Learning Algorithms for Malware Detection, Proceedings of the 2Nd ACM Workshop on Security and Artificial Intelligence (AISec '09), pp. 55－62 (2009).

[29] Rohitab.com: API Monitor, (http://www.rohitab.com/apimonitor, 2017年8月参照)

[30] R., Tian, R., Islam, L., Batten, and S., Versteeg: Differentiating malware from cleanware using behavioral analysis, 2010 5th International Conference on Malicious and Unwanted Software, pp.23－30 (2010).

[31] Iwamoto, K., and Wasaki, K.: Malware Classification Based on Extracted API Sequences Using Static Analysis, Proceedings of the Asian Internet Engineeering Conference (AINTEC '12), pp.31－38 (2012).

[32] Christodorescu, M., Jha, S., and Kruegel, C.: Mining Specifications of Malicious Behavior, Proceedings of the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering (ESEC-FSE '07), pp.5－14 (2007).

[33] Rafique, M., Z., and Chen, P., Huygens, C., and Joosen, W.: Evolutionary Algorithms for Classification of Malware Families Through Different Network Behaviors, Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (GECCO '14), pp.1167－1174 (2014).

[34] M., Iannacone, S., Bohn, G., Nakamura, J., Gerth, K., Huffer, R., Bridges, E., Ferragut, and J., Goodall: Developing an Ontology for Cyber Security Knowledge Graphs, In Proceedings of the 10th Annual Cyber and Information Security Research Conference (CISR '15), pp. 1－4 (2015).

[35] L., S., Kiat, M., A., Obaja, L., Wei, and O., C., Hui: MalwareTextDB: A Database for Annotated Malware Articles, Proceedings of the 55th Annual Meeting of the Association for Computational, Vol. 1, pp.1557－1567 (2017).

[36] V., Mulwad, W., Li, A., Joshi, T., Finin, and K., Viswanathan: Extracting Information about Security Vulnerabilities from Web Text, In Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 03 (WI-IAT '11), Vol. 3, pp. 257－260 (2011).

[37] A., Joshi, R., Lal, T., Finin, and A., Joshi: Extracting Cybersecurity Related Linked Data from Text, 2013 IEEE Seventh International Conference on Semantic Computing, pp. 252－259 (2013).

[38] C., L., Jones, R., A., Bridges, K., M., T. Huffer, and J., R., Goodall: Towards a Relation Extraction Framework for Cyber-Security Concepts, In Proceedings of the 10th Annual Cyber and Information Security Research Conference, pp.1－4 (2015).

[39] R., R., Ramnani, K., Shivaram, S., Sengupta, and Annervaz K. M.: Semi-Automated Information Extraction from Unstructured Threat Advisories, In Proceedings of the 10th Innovations in Software Engineering Conference (ISEC '17), pp.181－187 (2017).

[40] 山嵜麿与，神谷造：文書からの脅威と脆弱性知識の自動抽出，コンピュータセキュリティシンポジウム 2017（CSS2017）論文集，pp.393－398（2017）.

[41] Tang, S., Zhang, N., Zhang, J., Wu, F. and Zhuang, Y.: NITE: A Neural Inductive Teaching Framework for Domain Specific NER, Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017), pp. 2642－2647 (2017).

[42] Burr, S. and Mark, C.: Active Learning with Real Annotation Costs, NIPS Workshop on Cost-Sensitive Learning (2008).

[43] IBM: IBM Watson to Tackle Cybercrime, (https://www-03.ibm.com/press/us/en/pressrelease/49683.wss, 2018 年 7 月参照)

[44] S., Mittal, P., K., Das, V., Mulwad, A., Joshi, and T., Finin: CyberTwitter Using Twitter to generate alerts for Cybersecurity Threats and Vulnerabilities, International Symposium on Foundations of Open Source Intelligence and Security Informatics", IEEE Computer Society (2016).

[45] A., Ritter, E., Wright, W., Casey, and T., Mitchell: Weakly Supervised Extraction of Computer Security Events from Twitter, In Proceedings of the 24th International Conference on World Wide Web (WWW '15), pp. 896－905 (2015).

[46] A., Javed, P., Burnap, and O., Rana: Prediction of drive-by download attacks on Twitter, Information Processing & Management (2018).

[47] 孫博，秋山満昭，八木毅，森達哉：既知の悪性 URL 群と類似した特徴を持つ URL の検索，コンピュータセキュリティシンポジウム 2014（CSS2014）論文集，pp.1－8 (2014).

[48] Michael, D., Greg, H., Gilad, G., Aravind, A., and Prabaharan, P.: A Lexical Approach for Classifying Malicious URLs, 2015 International Conference on High Performance Computing (HPCS 2015), pp.195－202 (2015).

[49] Sujata, G., Niels, P., Monica, C., and Aviel D. R.: A framework for detection and measurement of phishing attacks, Proceedings of the 2007 ACM workshop on Recurring malcode, pp.1－8 (2007).

[50] Shuang, H., Alex, K., Brad, M., Vern, P., and Nick, F.: PREDATOR: Proactive Recognition and Elimination of Domain Abuse at Time-Of-Registration, Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16), pp.156

8－1579 (2016).

[51] Ma, J., Saul, L. K., Savage, S., and Voelker, G. M.: Beyond blacklists: learning to detec t malicious Web sites from suspicious URLs, Proc. of ACM SIGKDD international confe rence on Knowledge discovery and data mining (KDD 2009), pp.1245－1254 (2009).

[52] 千葉大紀，森達哉，後藤滋樹：悪性 Web サイト探索のための優先巡回順序の選定法，コンピュータセキュリティシンポジウム 2012（CSS2012）論文集，pp.805－812 (2012).

[53] 福島祥郎，堀良彰，櫻井幸一：悪性 Web サイト間の関連性に着目した信頼性評価によるブラックリスト方式の検討，情報処理学会研究報告，Vol-CSEC-52 No.38，pp.1－8 (2011).

[54] Leyla, B., Engin, K., Christopher, K., and Marco, B.: EXPOSURE: Finding Malicious Do mains Using Passive DNS Analysis, 18th Annual Network & Distributed System Securi ty Symposium (NDSS '11) (2011).

[55] Caruana, R., Lawrence, S., and Giles, L.: Overfitting in neural nets: Backpropagation c onjugate gradient, and early stopping, Proceedings of the 13th International Conference on Neural Information Processing Systems (NIPS 2000), pp.381－387 (2000).

[56] 畑田充弘, 稲積孝紀, 有川隼, サンドボックス解析結果に基づく URL ブラックリスト生成方式に関する事例調査, vol.ISEC2014-44, no.115, pp.309-314(2014).

[57] 角田朋, 大鳥朋哉, 藤井康宏, 谷口信彦, 木城武康, グレーリストを用いたホワイトリスト/ブラックリストの自動生成によるマルウェア感染検知方法の検討, 情報処理学会研究報告. SPT, Vol.20 14, No.16, pp.1-7(2014).

[58] 中里ほか: ホスト型 IDS を用いた不審プロセスの特定, SCIS2015 予稿集 (2015).

[59] Keisuke Takemori, et al.: Detection of Bot Infected PC Using Destination-based IP Add ress and Domain Name Whitelists, IPSJ Journal, Vol.52, No.4, pp.1706-1716 (2011).

[60] 川口ほか: 不審活動の端末間伝播に着目した標的型攻撃検知, 情報処理学会論文誌, Vol.57, No. 3, pp.1022-1039 (2016).

[61] A. Bifet, R. Gavalda: Learning from time-changing data with adaptive windowing, Proc. 7th SIAM Int. Conf. Data Mining, pp.443-448 (2007).

[62] Eduardo Viegas, et al.: Stream Learning and Anomaly-based Intrusion Detection in the Adversarial Settings, IEEE ISCC2017 (2017).

[63] Lindorfer, M., Kolbitsch, C., and Milani C., P.: Detecting Environment-sensitive Malwar e, Proceedings of the 14th International Conference on Recent Advances in Intrusion D etection, (RAID '11), pp.338－357 (2011).

[64] S., Honton, and J., Schmidhuber: Long Short-Term Memory, Neural Comput, No. 8, Vol. 9, pp.1735－1780 (1997).

[65] M., Schwartz, and K., K., Paliwal: Bidirectional recurrent neural networks, IEEE Trans actions on Signal Processing, pp. 2673－2681 (1997).

[66] A., Graves: Supervised sequence labeling with recurrent neural networks, PhD thesis, T echnische Universitat Munchen (2008).

[67] Siwei, L., Liheng, X., Kang, L., and Jun, Z.: Recurrent Convolutional Neural Networks for Text Classification, Proceedings of the Twenty-Ninth AAAI Conference on Artificial I ntelligence (AAAI '15), pp.2267－2273 (2015).

[68] V., Nair and G., E., Hinton: Rectified linear units improve restricted Boltzmann machin es, In Proceedings of the 27th International Conference on Machine Learning (ICML '1 0), pp. 807－814 (2010).

[69] 高田雄太, 寺田真敏, 村上純一, 笠間貴弘, 吉岡克成, 畑田 充弘：マルウェア対策のための研究用データセット ～MWS 2016 Datasets～, 情報処理学会研究報告書, Vol.2016-CSEC-74, No.17, pp.1－8 (2016).

[70] Cuckoo Foundation : Automated Malware Analysis, (http://www.cuckoosandbox.org/, 2017 年 8 月参照)

[71] John, D., Elad, H., and Yoram, S.: Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, Journal of Machine Learning Research, Vol. 12, pp.2121－ 2159 (2011).

[72] Kantchelian, A., Tschantz, M., C., Afroz, S., Miller, B., Shankar, V., Bachwani, R., Jose ph, A., D., and Tygar, J., D.: Better Malware Ground Truth: Techniques for Weighting Anti-Virus Vendor Labels, Proceedings of the 8th ACM Workshop on Artificial Intelligen ce and Security (AISec '15), pp.45－56 (2015)

[73] Sheng, V., S., Provost, F., and Ipeirotis, P., G.: Get Another Label? Improving Data Qu ality and Data Mining Using Multiple, Noisy Labelers, Proceedings of the 14th ACM S IGKDD International Conference on Knowledge Discovery and Data Mining (KDD '08), pp.614－622 (2008).

[74] Perdisci, R., and U., ManChon: VAMO: Towards a Fully Automated Malware Clustering Validity Analysis, Proceedings of the 28th Annual Computer Security Applications Con ference (ACSAC '12), pp.329－338 (2012).

[75] Perdisci, R., Lanzi, A., and Lee, W.: McBoost: Boosting Scalability in Malware Collectio n and Analysis Using Statistical Classification of Executables, Proceedings of the 2008 Annual Computer Security Applications Conference (ACSAC '08), pp. 301－310 (2008).

[76] Laskov, P., and Srndic, N.: Static Detection of Malicious JavaScript-bearing PDF Docum ents, Proceedings of the 27th Annual Computer Security Applications Conference (ACSA C '11), pp.373－382 (2011).

[77] Gascon, H. and Yamaguchi, F., Arp, D., and Rieck, K.: Structural Detection of Android Malware Using Embedded Call Graphs, Proceedings of the 2013 ACM Workshop on Art

ificial Intelligence and Security, (AISec '13), pp.45－54 (2013).
[78] James, B., Stephen, M., Caiming, X., and Richard, S.: quasi-recurrent neural networks, 5th International Conference on Learning Representations (ICLR '17) (2017).
[79] 齋藤邦子，鈴木潤，今村賢治：CRF を用いたブログからの固有表現抽出，言語処理学会第 13 回年次大会論文集(2007).
[80] STIX Project Documentation: About STIX, (https://stixproject.github.io/, 2018 年 7 月参照)
[81] US-CERT, (https://www.us-cert.gov/, 2018 年 7 月参照)
[82] ICS-CERT, (https://ics-cert.us-cert.gov/, 2018 年 7 月参照)
[83] Feedspot: Top 50 Cyber Security Blogs and Websites in 2018 For IT Security Pros, (https://blog.feedspot.com/cyber_security_blogs/, 2018 年 7 月参照)
[84] Natural Language Toolkit, (https://www.nltk.org/, 2018 年 7 月参照)
[85] Ma, X. and Hovy, E.: End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF, Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016), pp. 1064－1074 (2016).
[86] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K. and Dyer, C.: Neural Architectures for Named Entity Recognition, Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 260－270 (2016).
[87] ICS-CERT: Alert (ICS-ALERT-17-181-01C) Petya Malware Variant (Update C), available from <https://ics-cert.us-cert.gov/alerts/ICS-ALERT-17-181-01C> (2018-07-25 accessed).
[88] Mark, F., Christian, K., and Vern, P.: On the potential of proactive domain blacklisting, Proceedings of the 3rd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more (LEET '10), pp.1－8 (2010).
[89] Invernizzi, L., Benvenuti, S., Cova, M., Comparetti, P. M., Kruegel, C. and Vigna, G.: EvilSeed: A Guided Approach to Finding Malicious Web Pages, Proc. of IEEE Symposium on Security and Privacy, pp.428－442 (2012).
[90] 福島祥郎，堀良彰，櫻井幸一：ドメイン情報に着目した悪性 Web サイトの活動傾向調査と関連性分析，コンピュータセキュリティシンポジウム 2010（CSS2010）論文集 (2012).
[91] Ricardo, V. S., and Jose, C. B.: Identifying Botnets Using Anomaly Detection Techniques Applied to DNS Traffic, 5th IEEE Consumer Communications and Networking Conference, pp.476－481 (2008).
[92] scikit-learn: machine learning in Python, (http://scikit-learn.org/stable/, 2017 年 3 月参照)
[93] joepie91: GitHub - joepie91/python-whois: A python module for retrieving and parsing WHOIS data, (https://github.com/joepie91/python-whois, 2017 年 2 月参照)
[94] nominum: dnspython home page, (http://www.dnspython.org/, 2017 年 2 月参照)

[95] Amazon Web Services, Inc.: AWS | Alexa Web Information Service - Traffic Metrics for any Website , (https://aws.amazon.com/jp/awis/, 2017 年 2 月参照)
[96] MAXMIND: IP Geolocation and Online Fraud Prevention, (https://www.maxmind.com/en/home, 2017 年 2 月参照)
[97] dmoz: DMOZ - The Directory of the Web, (https://www.dmoz.org/, 2017 年 3 月参照)
[98] hpHosts: hpHosts Online - Simple, Searchable & FREE!, (http://www.hosts-file.net/, 2017 年 3 月参照)
[99] SPAMHAUS: The Spamhaus Project, (http://www.spamhaus.org/, 2017 年 3 月参照)
[100] Malware Domain List: MDL, (http://www.malwaredomainlist.com/, 2017 年 3 月参照)
[101] aguse : aguse.jp: ウェブ調査, (https://www.aguse.jp/, 2017 年 3 月参照)
[102] SMARTBEAR: The Cost of Poor Web Performance, (https://smartbear.com/blog/test-and-monitor/the-cost-of-poor-web-performance-infographic/, 2019 年 3 月参照)
[103] FireEyE: サイバー・セキュリティとマルウェア対策, (https://www.fireeye.jp/, 2017 年 8 月参照)
[104] ThreatTrackSecurity: Dynamic Malware Analysis Tools, Malware Sandbox, (http://www.threattracksecurity.com/enterprise-security/malware-analysis-sandbox-software.aspx, 2017 年 8 月参照)
[105] ASAHI INTERACTIVE, Inc. : 標的型攻撃メール：政府機関の開封率 10%--教育で 3%に減少，思わぬ課題も見つかる, (https://japan.zdnet.com/article/35013306/, 2017 年 6 月参照)
[106] IPA: 「高度標的型攻撃」対策に向けたシステム設計ガイド, (https://www.ipa.go.jp/files/000046236.pdf, 2017 年 6 月参照)
[107] トレンドマイクロ: 起動日時が設定された RAT「PlugX」, (http://blog.trendmicro.co.jp/archives/9357, 2017 年 6 月参照)
[108] van Rijsbergen, C.J.: Information Retrieval. Butterworths, London (1979).
[109] JPCERT/CC: 攻撃者が悪用する Windows コマンド(2015-12-02), (https://www.jpcert.or.jp/magazine/acreport-wincommand.html, 2018 年 4 月参照)
[110] Verizon, 2017 DATA BREACH INVESTIVATION REPORT, (https://www.ictsecuritymagazine.com/wp-content/uploads/2017-Data-Breach-Investigations-Report.pdf, 2018 年 4 月参照)
[111] Verizon, 2015 DATA BREACH INVESTIVATION REPORT, (https://iapp.org/media/pdf/resource_center/Verizon_data-breach-investigation-report-2015.pdf, 2018 年 4 月参照)
[112] J.H. Holland: Adaptation in Natural and Artificial Systems, University of Michigan Press/MIT Press (1975).
[113] FIRST.org, Inc.: Common Vulnerability Scoring System SIG, (https://www.first.org/cvss/, 2018 年 4 月参照)

10. 実績

10.1. 学術論文

No.	発表年月	「タイトル」，単著・共著の別，掲載誌等・巻・号（発行所・出版社等），頁
1	2020年2月（予定）	「機械学習を用いた自律進化型悪性サイトアクセス抑制手法の設計と実装」，共著（藤井翔太，重本倫宏ほか3名），情報処理学会論文誌（情報処理学会）（採録決定）
2	2018年8月	「Development of White List Based Autonomous Evolution of Defense System for RAT Malware」，共著（重本倫宏，菊池浩明ほか5名），Asia Joint Conference on Information Security（IEEE Computer Society），pp. 95-101
3	2018年3月	「ホワイトリストを用いた自律進化型防御システムの開発」，共著（重本倫宏，菊池浩明ほか5名），情報処理学会論文誌第59巻第3号（情報処理学会），pp. 1050-1060

10.2. 学会発表

No.	発表年月	「タイトル」，単独発表・共同発表の別，発表学会，開催地
4	2018年12月	「CyNER: Cybersecurity Named Entity Recognition System for Efficient Intelligence Analysis」，共同（発表代表者：藤井翔太），Annual Computer Security Applications Conference，Condado Plaza Hilton（アメリカ・プエルトリコ）（ポスター発表）
5	2018年10月	「未知語を考慮した固有表現認識によるセキュリティインテリジェンスの構造化手法」，共同（発表代表者：藤井翔太），コンピュータセキュリティシンポジウム，ホテルメトロポリタン長野（長野）
6	2018年7月	「プロファイル型ホワイトリスト準最適化手法の提案」，共同（発表代表者：重本倫宏），マルチメディア，分散，協調とモバイルシンポジウム，芦原温泉清風荘（福井）
7	2018年7月	「分散型SOCアーキテクチャに基づいた複数組織間におけるセキュリティ・オペレーションの連携」，共同（発表代表者：近藤賢朗），マルチメディア，分散，協調とモバイルシンポジウム，芦原温泉清風荘（福井）
8	2018年3月	「大学ネットワーク環境におけるSOC/CSIRT活動に用いる情報共有基盤の提案」，共同（発表代表者：近藤賢朗），インターネットと運用技術研究会，鬼怒川温泉ホテル（栃木）
9	2017年10月	「セキュリティ対処の影響を考慮した自動対処システムの提案」，共同（発表代表者：重本倫宏），コンピュータセキュリティシンポジウム，山形国際ホテル（山形）
10	2017年10月	「多種環境におけるマルウェア解析レポートを用いたマルウェア分類手法に関

No.	発表年月	「タイトル」，単独発表・共同発表の別，発表学会，開催地
11	2017年7月	する一検討」，共同（発表代表者：藤井翔太），コンピュータセキュリティシンポジウム，山形国際ホテル（山形） 「サイト関連情報に基づいたWebサイト脅威度推定機能の提案」，共同（発表代表者：藤井翔太），コンピュータセキュリティ研究会，ユビキタス協創広場CANVAS（東京）

謝辞

本論文は著者が明治大学大学院先端数理科学研究科先端メディアサイエンス専攻博士後期課程において，菊池研究室において行った研究をまとめたものです.

本研究に関して終始ご指導ご鞭撻を頂きました本学菊池浩明教授に心より感謝致します．また本論文を精読いただき有益で建設的なご議論を頂いた本学斉藤裕樹和教授，立命館大学毛利公一教授に深謝致します.

第 1 章から第 8 章記載の各システムの実装には，日立製作所システムイノベーション研究センタセキュリティ研究部の仲小路博史主任研究員，川口信隆研究員，藤井翔太研究員，同社サービス＆プラットフォームビジネスユニットの鬼頭哲郎氏，同社ライフ事業統括本部の藤井康広氏にご協力頂きました．また，評価に用いるマルウェアの検体収集には同社サービス＆プラットフォームビジネスユニットの梅木久志氏にご協力頂きました．第 7 章と第 8 章の評価実験では同研究所セキュリティ研究部の皆様の本来の業務へ悪影響が出る可能性があるにもかかわらず多くの方にご協力頂きました．関係者の方々に心より感謝します.

在学期間中，夜遅くまで熱く議論し，忌憚ない意見を頂いた明治大学菊池研究室の新原功一氏に感謝致します.

最後になりますが，業務と学業を両立する中，最後まで私の研究にご理解ご協力を頂いた妻 泰貞，娘 栞那と，日立製作所の同僚の皆様に心より感謝します．ありがとうございました.

付録 A　評価に用いたマルウェア

7 章の評価に用いたマルウェアと，評価結果を，表 A.1 に示す.

表 A.1　評価に用いたマルウェアと評価結果

No.	種別	ハッシュ値（MD5）	認証プロキシ	提案手法
1	Emdivi	e5653a4bca1239b095509438a3040244	遮断	遮断
2	PlugX	5a22e5aee4da2fe363b77f1351265a00	クリア	遮断
3	ChChes	8a93859e5f7079d6746832a3a22ff65c	遮断	遮断
4	ChChes	c1cb28327d3364768d1c1e4ce0d9bc07	遮断	遮断
5	ChChes	07abd6583295061eac2435ae470eff78	遮断	遮断
6	ChChes	37c89f291dbe880b1f3ac036e6b9c558	遮断	遮断
7	ChChes	472b1710794d5c420b9d921c484ca9e8	遮断	遮断
8	ChChes	23d03ee4bf57de7087055b230dae7c5b	遮断	遮断
9	ChChes	75500bb4143a052795ec7d2e61ac3261	遮断	遮断
10	ChChes	0b6845fbfa54511f21d93ef90f77c8de	遮断	遮断
11	Korplug	c48cdf2ce519307358ead3512e31f264	遮断	遮断
12	PlugX	855f9cf0cac91c50cfc3160bc8e905b9	遮断	遮断
13	PlugX	37316726bf3d3934cf51366db702c24d	遮断	遮断
14	pak!cobra	adc7303a61a7f3341f567ff09c411554	遮断	遮断
15	Chches	19610f0d343657f6842d2045e8818f09	遮断	遮断
16	ChChes	1b891bc2e5038615efafabe48920f200	遮断	遮断
17	ChChes	1d0105cf8e076b33ed499f1dfef9a46b	遮断	遮断
18	ChChes	3afa9243b3aeb534e02426569d85e517	遮断	遮断
19	ChChes	684888079aaf7ed25e725b55a3695062	遮断	遮断
20	ChChes	779dbb88e037a6ecc8ab352961dbb028	遮断	遮断
21	ChChes	7891f00dcab0e4a2f928422062e94213	遮断	遮断
22	ChChes	b0649c1f7fb15796805ca983fd8f95a3	遮断	遮断
23	ChChes	ca9644ef0f7ed355a842f6e2d4511546	遮断	遮断
24	ChChes	d1bab4a30f2889ad392d17573302f097	遮断	遮断
25	ChChes	f03f70d331c6564aec8931f481949188	遮断	遮断
26	ChChes	f5744d72c6919f994ff452b0e758ffee	遮断	遮断
27	ChChes	f586edd88023f49bc4f9d84f9fb6bd7d	遮断	遮断

28	Trojan	0e74be5dfabfb9c1eabb283a42395db0	クリア	遮断
29	Emdivi	2f2bb56fc759213a6377b0b885cabc4e	クリア	遮断
30	Emdivi	f60cdde57bd9ca9412c32a08ef068abc	クリア	遮断
31	Emdivi	4be4ebe1db4ea1be2f293037eb7f8b0f	クリア	遮断
32	Emdivi	dba397405916869fdbfc66fa57f553ae	クリア	遮断
33	PoisonIvy	1aca09c5eefb37539e86ec86dd3be72f	遮断	遮断
34	Emdivi	3b6b7907602f8447b168f1225f37c643	クリア	遮断
35	Emdivi	b3bc4b5f17fd5f87ec3714c6587f6906	クリア	遮断
36	Emdivi	438a3b6783fb290197d3023ce441229c	遮断	遮断
37	Plugx	b18a316b2ce6e099fe7fbf69283cbc5e	遮断	遮断
38	Emdivi	b56aa4a6e4cde2a7126c8d91cb728db4	クリア	遮断
39	Trojan	6be2123f6d5a7890fc33656ca2a17bb3	遮断	遮断
40	Trojan	839b46c9043cecfa61ce10e5c1a6608b	遮断	遮断
41	Zegost	a30262bf36b3023ef717b6e23e21bd30	遮断	遮断
42	Trojan	7a94e32ff340306132920d654057e7c0	遮断	遮断
43	Trojan	1a6113bb0a9757a8c6abfc7a2ebb886b	遮断	遮断
44	Trojan	14e859f0048314a705222a13ead89660	遮断	遮断
45	Trojan	1df1ff8dcbadad643ded759410ffcdb8	遮断	遮断
46	Zusy	f3c514f5653999501793ee8501e21570	遮断	遮断
47	Trojan	074ccdaa6a4641bb336194fd079309ee	遮断	遮断
48	Emdivi	3b2b36edbf2934c7a872e32c5bfcde2a	遮断	遮断
49	Emdivi	8bf944283987de847851d3d2279b8cf8	遮断	遮断
50	Trojan	67b06935b3c919d8dfbe2166fc6f8305	遮断	遮断