修士論文

サイバー空間における擬態とマルウェア忌避技術への応用

北沢 尭宏

主指導教員　篠田 陽一 教授

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

令和２年３月

Abstract

情報処理技術が発展し、文書ファイルや個人情報、パスワード等の重要な情報がコンピュータ上で管理され、インターネット上に存在することが当たり前となった。インターネットを介してそれらの情報を奪取・盗聴を行うサイバー攻撃が存在する。情報処理推進機構が発表した情報セキュリティ10大脅威2019では、2019年に発生した組織に対するサイバー攻撃は、標的型攻撃、ビジネスメール詐欺、ランサムウェアの順に被害が大きかった。

2009年2月3日に公開された第二次情報セキュリティ基本計画において、攻撃による侵入や被害を前提としたセキュリティ対策を行う事故前提社会への対応が必要であると提言している。2019年時点では、IPSやサイバーレジリエンス、欺瞞的防御などが事故前提のセキュリティ対策に代表される。欺瞞的防御は、組織に所属しない攻撃者がネットワーク内に侵入することを前提とし、攻撃者を欺く欺瞞機構を用いる。欺瞞的防御の目的は、一度の攻撃に必要なコストを増加させることであり、コストに対する目標物の価値の低下や攻撃の停止を誘う。欺瞞的防御を実現させる手法には、難解化と擬態の2種がある。また、擬態に関する先行研究では、擬態には更に保護色と警告色に分類が可能であり、保護色擬態はハニーポットのような攻撃者にとって脅威である特徴を隠蔽する環境を、警告色擬態はその対象で特徴を再現することで防御を行う環境であると示していた。

しかし、サイバー空間における欺瞞と擬態に関する先行研究ではそれらの定義が曖昧であり、特に擬態に関して詳細に定義したものは確認できなかった。また、擬態を行うために必要な特徴に関しては、プロセスやファイル等に存在する名前や存在パスのような、表面的な情報しか定義していなかった。

そのため、本研究ではサイバー空間における欺瞞と擬態を明確にすることを目的として詳細に定義した。また、擬態を用いた欺瞞的防御を構築する際の擬態構築フローの提案と、擬態に必要な擬態要素と擬態要素を構成する条件である擬態条件についてまとめた擬態要素分類表を作成した。擬態要素分類表を用いることで、巧妙な擬態の構築と構築した擬態環境の擬態レベルを確認することができる。また、擬態構築フローと擬態要素分類表を使用した一例として、マルウェア忌避技術への応用を行った。本研究で提案したマルウェア忌避を目的とした環境は、既存研究の構築方法とは異なり、各マルウェアプロセスに対して整合性があり、かつ独立した擬態環境を提供が可能になっている。

マルウェア忌避技術への応用では、整合性があり、かつ独立した擬態環境を提供を行う簡単なプログラムを作成した。このプログラムの検証として、対解析機能のみを持った擬似的なマルウェアを作成し、その動作を確認した。検証では、各擬似マルウェアプロセスはそれぞれ異なる擬態環境を提供されていることを確認し、その有用性と実現可能性を示した。また、先行研究と本研究の設計手法と実装手法の定性評価を行なった。

本研究では、サイバー空間における欺瞞と擬態を定義し、その詳細化と構築手順、擬態要素分類表を提案した。また、本研究の応用の一例として、マルウェア忌避技術への応用を提案した。

本研究の展望として、3 つの課題と展望が残る。1 つは、本研究で行った 4 種の擬態カテゴリにおいて、認識型と保護型の擬態について考察する必要がある。認識型と保護型は、周囲に存在するモノへの影響を与えるが、その効果を引き出すために必要な要素やシチュエーション、心理に与える影響の最大値などを明らかにする必要がある。2 つ目に、本研究では Windows を対象としており、セキュリティツールのみ実装した。そのため、他の OS やセキュリティツール以外に擬態する際の実装に関しては、擬態要素分類表を変更する必要がある。OS やコンテンツの目的が異なる場合に変化する擬態要素や注目すべき点が変更する可能性が生じる。そのため、これの考察を行う必要がある。3 つ目に、Mimetic DLL またはサーバに高い擬態レベルを持つ環境を実装することである。本研究では簡単な実装例として 2 種の対解析機能に対応したが、それ以外の対解析機能にも対応させる必要がある。擬態を用いた欺瞞的防御は攻撃者の攻撃コストを増加させ、攻撃の停止を誘う目的ではあるが、より巧妙な擬態の提供ができれば、攻撃者とセキュリティ従事者のいたちごっこの関係を優位に進めることが可能になると、私は考える。

目 次

図 目 次

表 目 次

第1章　はじめに

1.1　背景

コンピュータ技術の発展により、文書ファイルや個人情報、パスワードのような重要な情報をコンピュータ上で扱うようになった。同様に発展したインターネット技術とセキュリティ技術は、機密性と完全性、可用性を担保して通信を可能にしている。これらの技術の発展がユーザに利益を与える一方で、技術を悪用しシステムの脆弱性を利用することで、重要な情報の奪取や破壊、サービス運用の停止などを狙うサイバー攻撃技術も同様に発達している。情報通信研究機構（NICT）が公開している NICTER プロジェクトレポート 2018 によれば、ダークネットで観測されるパケットは過去 10 年で約 60 倍に増加している [1]。全てが攻撃パケットではないが、攻撃対象の調査や攻撃パケットが含まれているため、サイバー攻撃は身近なものである。

サイバー攻撃から重要情報やネットワーク機器を防御するために、内閣サイバーセキュリティセンター（NISC）はセキュリティソフトウェアやセキュリティ機器で対策を行うことを推奨している。代表的なサイバーセキュリティ技術としては、Firewall や IDS、マルウェア対策ソフトなどがある。

しかし、上記のようなサイバーセキュリティ技術だけでは対応が困難な攻撃も存在する。未知の脆弱性をついた攻撃やソフトウェアの設定不良などのソーシャルエンジニアリングが、対応が困難な攻撃に該当する。これが起因となるサイバー攻撃が発生することは珍しくなく、サイバー攻撃の発生から管理者の対応までの間は、端末やネットワークは無防備な状態になってしまう。高度で持続的な標的型攻撃（APT 攻撃：Advanced Persistent Threat 攻撃）も、対策が困難な攻撃の一つである。APT 攻撃は特定の職業を持つユーザや組織を対象としたサイバー攻撃であり、その巧妙化された攻撃手法と未知の脆弱性をつくマルウェアは対処が難しく、システム管理者が気づかないうちに攻撃が完了していたということも多々

ある。

このように、事前対策を目的とした既存のサイバーセキュリティでは対策が困難な攻撃も多々存在するため、適切なセキュリティ対策を行なった場合であっても、安全は確立されない。2009 年 2 月 3 日に公開された第 2 次情報セキュリティ基本計画において、内閣サイバーセキュリティセンター（NISC）は、侵入を事前に阻止するセキュリティ対策を行うだけでなく、侵入を許した後の対策を考える「事故前提社会」を提唱している [2]。

そのため、既存の事前対策型セキュリティに加え、侵入を前提として重要情報の保護等を行う侵入前提型セキュリティが重要になる。

1.2 目的

1.1 節で述べた通り、事故前提のサイバーセキュリティとして、侵入前提型セキュリティに関する研究が重要となる。そのような現状に対し、本研究では既知のサイバー攻撃への対応が可能であり、かつ未知のサイバー攻撃にも対応可能な環境を構築することで、侵入前提型のサイバーセキュリティを実現すること目的とする。

本研究では、侵入前提型のサイバーセキュリティとして欺瞞的防御に注目した。欺瞞的防御とは、ネットワークやコンピュータ構造に偽の情報やシステム構造の把握が困難な機構を保持することで、容易に攻撃できない状態を構築する防御手法である。これにより、攻撃者が攻撃行動をとる際に必要となる時間的、コンピューティング的資源を増大させ、攻撃行動の中止を誘う。そのため、コンピュータへの侵入と同一ネットワーク内の他コンピュータへの攻撃を繰り返し、組織内権限の昇格を狙う APT 攻撃のような攻撃に効果的であるとされている。

欺瞞的防御を実現する手法には、難解と擬態の 2 種類が存在する。本論文では、擬態を用いた欺瞞的防御を実装することで、侵入前提型のサイバーセキュリティを実現することを最終的な目的とする。

1.3 本論文の構成

本論文は、本章を含めて 7 章から構成される。2 章の「サイバー空間における欺瞞的防御と擬態を用いた防御」では、本研究の位置付けと欺瞞的防御とサイバー

空間における擬態の整理について述べる。3 章「先行研究と関連技術」では、欺瞞的防御と擬態を用いた防御について言及した研究の紹介や、擬態の実装に関する研究をまとめ、先行研究との差異や本研究における取り組みを述べる。4 章「サイバー空間における擬態の分類」では、サイバー空間における擬態の分類を行い、擬態構築フローと擬態要素分類表を述べた。5 章「警告型擬態を用いたマルウェア忌避環境の設計と実装」では、前章で提案した構築フローと擬態要素分類表を使用し、広く Windows を対象としたマルウェア忌避環境の構築について述べる。6 章「まとめと考察」では、作成した警告型擬態を用いたマルウェア忌避環境について述べ、7 章「おわりに」で本研究で行ったことと今後の課題について述べる。

第2章　サイバー空間における欺瞞的防御と擬態を用いた防御

本章では、本研究に関連する技術、概念と用語についてまとめる。

2.1　サイバーセキュリティ各種技術の区分

世界初のマルウェアと呼ばれる The Creeper[1] とその対抗プログラムが開発された 1970 年代から今日まで、約 50 年が経過している。その間、攻撃者とサイバーセキュリティ技術者の間ではいたちごっこが行われてきた。

アメリカ国防総省の定めた軍事施策である Joint Publication3-13 Information Operations（JP3-13） [3] において、コンピュータネットワーク操作（CNO）について述べている。CNO とは、コンピュータネットワークエクスプロイト（CNE）とコンピュータネットワーク防御（CND)、コンピュータネットワーク攻撃（CNA）の 3 大要素から構成される、ネットワークを介したコンピュータ操作の分類である。本施策では、CND 保護システムの実現について、7 種のカテゴリを述べている。CND 保護システムには、Deny（拒否)、Disrupt（中断)、Degrade（劣化)、Deceive（欺罔)、Discover（探索)、Destroy（破壊)、Detect（検出）が存在する。表 2.1 では、JP3-13 を参考に、それぞれの定義と効果をまとめた。

また、Almeshekah ら [4] はサイバー防御のメカニズムを 4 つに分類している。

- Denial & Isolation
- Degradation & Obfuscation

[1] https://www.exabeam.com/information-security/creeper-computer-virus/

表 2.1: 保護システム

保護技術	目的	効果
Deny （拒否）	悪意のある第三者による攻撃や操作を防ぐ。	攻撃行動の未然阻止
Disrupt （中断）	悪意のある第三者による攻撃や操作が行われている際に、それを失敗させる。	攻撃行動の動的阻止
Degrade （妨害）	悪意のある第三者による攻撃や操作に対し、行動の邪魔をし、遅延させる。	攻撃行動の妨害 攻撃対象としての価値の低下
Deceive （欺罔）	悪意のある第三者による攻撃や操作に対し、成功したと錯覚させる。	攻撃者の分析が可能
Destroy （破壊）	悪意のある第三者への攻撃。 法執行機関が対応する。	攻撃の停止
Discover （探索）	悪意のある第三者による攻撃や操作のログを記録する。	攻撃手法やツールが分析可能
Detect （検出）	監視によって得たログを参照し、悪意のある第三者による攻撃や操作を確認した際に検出通知を送信する。	攻撃行動の静的阻止

- Negative Information & Deception
- Attribution & Counter-Operations

Almeshekah らの分類は JP3-13 に類似し、8 つのカテゴリを定義し、さらに性質の似たカテゴリをまとめた 4 カテゴリに分類している。JP3-13 での分類と似ているが、Obfuscation（難読化）と Counter-Operations（反撃）カテゴリが異なる。

本研究では、CND 保護システムの「Deceive」や「Deception」、Almeshekah らの提案したサイバー防御のメカニズム「Negative Information & Deception」に位置する防御手法を提案する。

2.2 既存のセキュリティ技術

2.1 節のように、サイバー防御技術は主に 4 種に分類される。各サイバー防御技術によって得られる効果や目的、対策可能な規模も様々でり、悪意のある通信を遮断するものもあれば、悪意のある動作を停止するものもある。

本節では、サイバー防御技術の中で広く普及しているマルウェア対策に関連する技術を紹介する。

2.2.1 既存セキュリティで使用される検出技術

一般的に、既存セキュリティの検出技術は以下の 3 種類に分類される。

- パターンマッチング方式
- ヒューリスティック方式
- ふるまい検出方式

サイバーセキュリティでは、しばしばシグネチャという言葉が用いられる。シグネチャとは、悪意のあるスクリプトファイルや実行可能ファイルに共通するバイトコードを指す言葉であり、セキュリティエンジニアはリバースエンジニアリングを行う事で、その特徴をまとめた定義ファイルを作成する。

パターンマッチング方式を用いる対策では、定義ファイルを元にファイルの検査を行い、シグネチャに一致したファイルの削除を行う。この方式のメリットとして、一度定義したシグネチャは非常に高い精度で検出が可能なことが挙げられる。

ヒューリスティック方式を用いる対策では、悪意のあるプロセスに共通した処理を事前に設定ファイルに記述しておき、のちに実行されるプロセスが設定と比較して危険であるか否かを検知する。ヒューリスティック方式では、パターンマッチング方式では検出不可な未知のマルウェアや亜種に対して、多少柔軟に対応可能である特徴を持つ。

ふるまい検出方式では、実行ファイルを実際に動作させ、そのプロセスの動作を確認してからマルウェアか否かを判断する。ヒューリスティック方式と検出方法は近いが、柔軟性という点でヒューリスティック方式に優っている。一方、検出方法に柔軟性を持つヒューリスティック方式とふるまい方式では、悪意のないプロセスであっても悪意のあるプロセスに共通した処理を行うプロセスが存在した場合、誤検知してしまうというトレードオフの関係にある。

また、機械学習の発達により、上記の方式を組み合わせた検出技術も存在する。

アンチウィルス対策などでは上記の検出技術を使用することで、端末やネットワークの保護を行なっている。

2.2.2　攻撃行動の抑止を重視した侵入前提型セキュリティ対策

2.1 節で述べたように、サイバーセキュリティには様々な分野が存在する。一般には、上記の方式を利用した事前対策型のサイバーセキュリティが知られている。本節では、サイバーセキュリティ分野の中でも攻撃者の侵入を前提とした分野である、侵入前提型のサイバーセキュリティについて述べる。

Intrusion Detection System（IDS）/ Intrusion Prevention System（IPS）

IDS は、外部からの不正な通信やファイルの改ざんなどを検出するセキュリティツールである。

主にネットワーク型とホスト型に分けられる。前者はネットワークセグメント上に設置され、ネットワーク上を流れるトラフィックの監視を行うことで不正な通信を検出している。後者は、ホスト上にソフトウェアとして設置され、ホストの受信したトラフィックや OS やアプリケーションが生成するイベントログを監視し、異常を検出する。ホスト型 IDS の検出方法は主に 2 種類ある。一つは、シグネチャ型の検出方法であり、事前に設定した不正な通信・動作の特徴をシグネチャとして設定しておく必要があるブラックリスト型の検知である。もう一つは、アノマリ型と呼ばれ、通常時の動作を事前に設定ファイルに記述するホワイトリスト型の検知である。

これらの IDS を設置することで攻撃者の侵入を前提としたセキュリティ対策はできるが、以下のような問題がある。

- 検知の不確定性（ネットワーク型）
- ログの常時監視による負荷増大（ホスト型）
- 導入に伴うホストへの影響（ホスト型）
- ホストの信憑性による検出困難化（ホスト型）

ネットワーク型 IDS が不正な通信を検知した場合、本当に不正な通信であるとは限らない。ネットワーク型 IDS は、トラフィックが受信ホストに与える影響やその重要度を判別することができない。そのため、適切な閾値の設定が必要になる。

ホスト型 IDS の設置は、OS やアプリケーションが出力するログを常に監視するため、負荷が高くなる。また、システムコールなどのカーネルに近い動作を監視する場合、カーネルに変更を加える必要がある。そのため、ホスト型 IDS の導入には十分な知識が必要になる。侵入者の設置するマルウェアには、ログへの出力を阻害するモノがある。そのため、ホスト型 IDS を設置した場合でも、ログが存在しないため検出ができない場合もある。

IPS は、IDS 同様のトラフィック監視とログ監視が可能であり、これに加え不正な通信を検出した際にはそれを遮断することを目的としたセキュリティツールである。IDS 同様、設置方法としてネットワーク型とホスト型があり、検出方法としてシグネチャ型とアノマリ型が存在する。

欺瞞的防御

欺瞞的防御は一般的なセキュリティ方式とは根本的に異なり、マルウェアや不正な通信の検知・フィルタリングを目的としない。欺瞞的防御では、攻撃者の侵入を前提として事前に設置するサイバー攻撃防御手法であり、攻撃者の活動に不安要素を与える情報を保持する。

この防御手法の目的は、侵入を行なった攻撃者の行動を抑制する欺瞞機構を用いて、攻撃に費やすコストを増大化させることにある。ここで対象となるコストとは、攻撃者が目的達成するまでに必要な時間などが対象となる。欺瞞的防御は、攻撃者の消費コストに対して得られた情報の価値を低下することで撤退を誘う。

2.2.3 サービス提供の継続を重視した侵入前提型セキュリティ技術

前節では、攻撃者がネットワークへ侵入した後に行うセキュリティ技術について述べた。本節では、セキュリティ技術の中でもクライアントへのサービスの提供を重視したセキュリティ技術について述べる。

ディジタルフォレンジック

ディジタルフォレンジックとは、法科学に属する分野に位置する。

ディジタルフォレンジックについては、Digital Forensics Research Workshop（DFRWS） [5] が以下のように定義している。

> the use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation, and presentation of digital evidence derived from digital sources for the purpose of facilitation or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations.（論文より引用）

まとめると、ディジタルフォレンジックは、サイバー攻撃によって被害が発生した際、その原因の究明を行うことを目的としている。デバイスやネットワーク機器上に蓄積された情報を収集、保存し、その特徴を分析、特定、分析、解釈し、その手口の証明を行っている。

インシデントレスポンス

インシデントレスポンスは、その名称の通りインシデントに対応することを指す。インシデントが発生した際、被害拡大の防止や原因調査、対応策の検討や実施、復旧までを行う。

JPCIRT/CC は、インシデント観測から対応までに行うべき対処について、マニュアルを公開している [6]。

IDS と IPS のようなセキュリティ機器は広く普及しているが、未知のマルウェアを検出することは難しい。また、フォレンジックやインシデントレスポンスは、サービス稼働率を維持することを第一の目的に原因究明を行うため、侵入を前提とした対策ではあるが、攻撃者の活動を許していることが問題である。本研究では、システムの保護を目的としている欺瞞的防御に注目した。

2.3 欺瞞的防御

前述した通り、欺瞞的防御では欺瞞機構を用いて攻撃者に不安要素を与える。これによって、攻撃コストの増大化を狙い、情報やシステムを保持する。

本研究において、複数の欺瞞的防御に関する先行研究を調査する中で、欺瞞的防御の定義が不確かであることに気づいた。そのため、本節では、先行研究をもとに欺瞞的防御の定義を固める。

2.1 節で紹介した JP3-13 では、2 種の欺瞞に関連する言葉が定義されている。一つは Deceive であり、偽の情報を攻撃者の目標物と錯覚させることを目的としている。もう一つは Degrade であり、攻撃者が調査行動によって得る情報を改悪することで、攻撃成功率の低減を目的としている。これらは、攻撃コストの押し上げを行う技術要素であるため、欺瞞の定義であると言える。

Pingree らの研究 [7] では、コンピュータセキュリティにおける欺瞞を、攻撃者にミスリードさせ、それによって攻撃者にコンピュータセキュリティ防御を援護する特定の行動を取らせるために計画された行動、と定義している。

2.1 節で述べた Almeshekah らの研究では、欺瞞によって得られる効果を以下のようにまとめている。

- 攻撃者を混乱、妨害、遅延させる
- おとりを用いて侵入検知する
- 攻撃者が得る情報の価値を落とす

角丸らの研究 [8] では、欺瞞的防御を 3 種に分類している。

- 操作的欺瞞
- 模倣的欺瞞
- 模擬的欺瞞

模擬的欺瞞

模擬的欺瞞（Simulative Deception）とは、実体はないが虚偽の情報を流布し、実体のない標的を捏造する欺瞞手法である。実体はないため、攻撃者が攻撃を行なった場合でも被害は発生しない特徴を持つ。主に飽和を目的とした欺瞞に効果的な手法であるとしている。

模倣的欺瞞

模倣的欺瞞（Imitative Deception）とは、実体の存在する対象に偽の情報を付与することで標的と認識させない欺瞞手法である。模擬的欺瞞と異なり実体はあるため、攻撃を受ける場合もある。主に誘惑に対して有効であり、探知に対しても応用できるとしている。

操作的欺瞞

操作的欺瞞（Manipulative Deception）とは、攻撃者の探索行動または攻撃行動に対し偽の情報を与えることで、攻撃者の撹乱を行なう防御手法である。主に探知に対して有効であり、その他 2 つに対しても応用が可能としている。

以上から、本研究では先行研究を整理して欺瞞的防御の定義を行なった。

サイバー空間における欺瞞とは、第三者に対し電子的特徴を与えることで心理に影響を及ぼし、その後の行動を制限又は誘導することと定義した。

2.2.2 節で示したように、欺瞞的防御の目的は、欺瞞機構を用いることで攻撃コストを押し上げ、撤退を誘うことである。

また、欺瞞的防御の実現手法は 2 種が存在する。

- 難解化
- 擬態

難解では、システム構成やファイル名など、防御対象に似た特徴を持ったファイル構造を保持することで、価値のある情報の認知を困難にすることである。擬態では、防御対象のシステムや情報を、偽の情報を用いることで第三者には価値のないものに見せている。

2.3.1 難解を用いた欺瞞的防御

難解を用いた欺瞞的防御では、防御対象の情報に手を加えることなく、対象の認知を困難にしている。認知の困難化の例として、対象に似た情報の保持がある。

同じ階層にファイル名が一文字違いの偽ファイルを多数保持するシステムが存在している場合、本物のファイルの認知が困難になる。その他にも、ディレクトリ構成が類似した階層構成を持たせることで、カレントディレクトリや操作の把握を困難にする場合もある。

このようにすることで、認知の困難化を行う欺瞞的防御を、本研究では難解と定義した。

難解による欺瞞的防御の目的は、防御対象の認知の困難化にある。認知を困難にすることで、システム構成を知らない第三者である攻撃者は、システムに存在する偽のファイルを含めたファイル群を確認する必要が発生し、攻撃コストが増大化する。

また、認知の困難化を実現する手法は以下のようなものが挙げられる。

- 希薄
- 迷宮

希薄とは、任意の階層において攻撃者が選択可能な行動の数を増加させることである。この選択肢には、防御対象の特徴を模倣したディレクトリやファイル、デバイスなどが該当する。

迷宮とは、攻撃者が目的とする対象までの過程を複雑にすることである。これは、防御対象までのパスを複雑化し、カレントディレクトリの認知を困難にすることなどで成立する。

上記の欺瞞手法はよく似ているため、これらを組み合わせることも可能であり、より幅のある欺瞞が可能になる。

また、希薄は同一階層の認知の困難化を、迷宮は複数階層の認知の困難化を行なっているため、これを軸とした二次元の難読欺瞞のグラフが生成可能である。これを用いて、難読欺瞞の欺瞞度合いを確認することができる。

2.4 擬態を用いた欺瞞的防御

擬態を用いた欺瞞的防御では、防御対象の情報に手を加えることで、対象の認知を困難にしている。

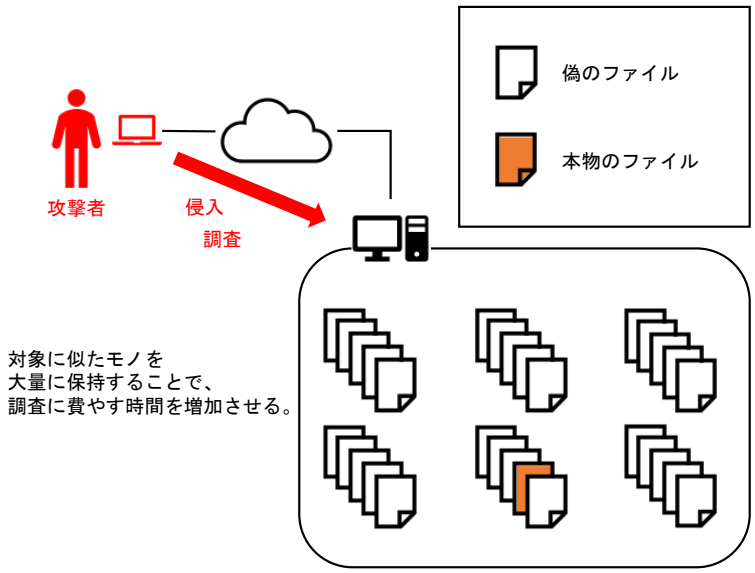

図 2.1: 難読化における希薄の概念図

サイバー空間における欺瞞と同様に、サイバー空間における擬態についての定義も曖昧である。そのため、本節でも先行研究を整理してサイバー空間における擬態の定義を行う。

Frank J [9] は、サイバー空間において欺瞞を行うためのフレームワークである Cyber–Denial&Deception（D&D）Framework（表 2.2）を提案している。

真実の開示、虚偽の開示、真実の隠蔽、虚偽の隠蔽の 4 つからなるこの行列は、サイバー空間における擬態を定義している。真実の開示は、サイバー空間に存在する情報の中で、実際に存在する情報を外部へ公開することを示している。虚偽の開示は、サイバー空間において、偽の情報を外部へ公開することを示している。同様に、真実の隠蔽は実際に存在する情報を隠蔽し、外部への公開を行わないことを、虚偽の隠蔽は、偽の情報を外部公開しないことを示している。

Leyi Shi らによる研究 [10] では、サイバー空間における擬態を 2 種に分類している。

- 保護色（protective coloration）
- 警告色（warning coloration）

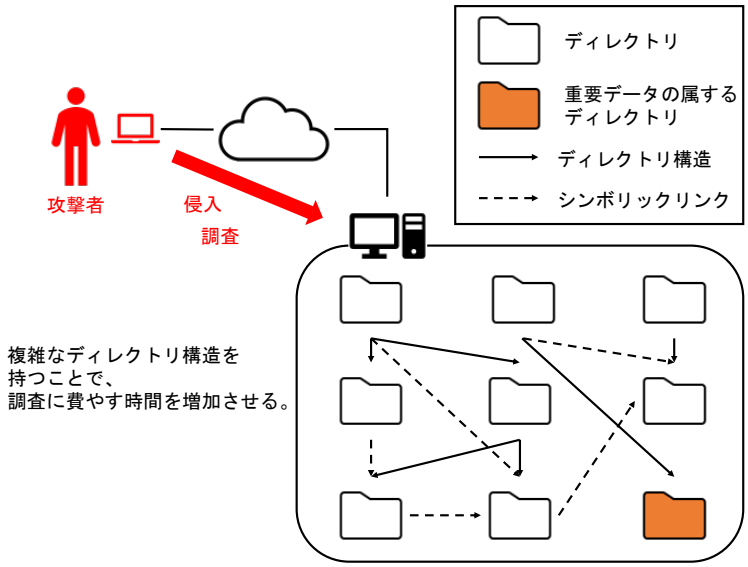

図 2.2: 難読化における迷宮化の概念図

サイバー空間における保護色擬態は、ハニーポットのように、環境が持つ特徴を隠蔽するものと定義している。より詳細には、同一組織に存在するサーバなどのネットワークホストやソフトウェア、サービス情報などを模倣することを指し、基本的に設置者以外には模倣対象と同等の存在として認識される。

対して、サイバー空間における警告色は、ハニーポットが持つ特徴や特性に類似した特徴を再現したものと定義している。保護色同様に、基本的に設置者以外には模倣対象と同等の存在として認識される。しかし、先行研究では擬態を詳細に定義していない。

以上から、本研究では先行研究を整理してサイバー空間における擬態の定義を行なった。

先行研究で述べられている擬態行動から、サイバー空間における擬態とは、電子的特徴を再現または隠蔽することで、第三者からの認識を阻害することと定義する。

擬態を用いた欺瞞的防御の目的は、以下の 3 つである。

- 認識の阻害
- 攻撃活動の妨害

表 2.2: Cyber–D&D Framework
（著者の論文を参考に作成）

欺瞞 オブジェクト	欺瞞：情報の公開	拒否：情報の隠蔽
真実	• 内部情報の公開 • 補助的情報の公開 • アクセス権の付与	• アクセスの拒否 • 経路情報の変更 • トラフィックの傍受
虚偽	• ハニーポットの設置（公開） • 偽情報の公開 • シミュレーション	• フィンガープリントの隠蔽 • セキュリティ情報の隠蔽

- 攻撃ツールの妨害

攻撃者が活動を避ける環境には、以下のようなものがある。

- ハニーポット
- サンドボックス
- 解析ソフトウェア
- 仮想環境

ハニーポットは、攻撃者の使用する攻撃ツールや攻撃手法などを調査することを目的としている。任意の脆弱性を保持したサーバーまたはホストを用いることで攻撃者に侵入を許し、攻撃行動のログや攻撃ツールを記録するセキュリティツールである。外部から観測可能な脆弱性の保持や意図してマルウェアを実行させることで攻撃者に脆弱性情報や攻撃機会を提供し、攻撃観測を可能にしている。ハニーポットには攻撃者の行動等を記録するロギング機能が存在し、これを用いることで攻撃の解析が可能になっている。

また、サンドボックスは脅威情報が不明なファイルやコンテンツの脅威を動的解析するセキュリティツールである。これもハニーポットと同様に、ロギング機

能やセキュリティ機能を利用することで、調査対象がマルウェアか否か、リモートアクセスツール（RAT : Remote Access Tool）を仕掛けているのかなどの調査・解析が可能である。

解析ソフトウェアもサンドボックスと同様に、脅威情報が不明なファイルやコンテンツを解析するセキュリティツールである。サンドボックスとの違いは、静的に解析を行うという点である。解析ソフトウェアはマルウェアの解析にも使用されるが、通常のソフトウェアデバッグにも使用される。

上記のような環境は、攻撃者にとって攻撃手口が明らかにされる可能性があるため、攻撃者は避ける傾向にある。その理由として、攻撃者にとって利益が得られる期間が挙げられる。攻撃者の利益は、攻撃活動の開始やマルウェアの拡散からセキュリティ従事者による観測・解析・対策がなされるまでの間しか得られない。そのため、攻撃による利益を得られる期間を確保するために、解析環境を検出し、意図して避けている。

擬態を用いた欺瞞的防御では、この攻撃者心理と検出技術を逆用している。これまでは、セキュリティ従事者はマルウェアに実装されている攻撃観測環境と解析環境の検出技術を騙して解析し、攻撃者は攻撃観測環境と解析環境の検出技術によるのマルウェアの堅牢化を行なっていた。

擬態による欺瞞的防御は、セキュリティ従事者がこのようないたちごっこの関係で優位に立つためのセキュリティ手法である。

次節では、擬態による欺瞞的防御の根幹となる、攻撃観測環境や解析環境を検出する耐解析技術について述べる。

2.5　攻撃ツールやマルウェアの持つ耐解析技術

2.4 節では、擬態を用いた欺瞞的防御について述べた。

本節では、擬態による欺瞞防御の根幹となる技術である、耐解析技術について述べる。

耐解析技術とは、ソフトウェア製作者が自身の作成したソフトウェアをリバースエンジニアリングから保護するために用いられる技術である。これは、著作物を保護するために実装されるため、それ自体に違法性はない。

本節で述べる技術は、マルウェアに実装されている耐解析技術である。

2.5.1 Anti-VirtualMachine 技術

2.4 節で述べたように、仮想化技術はサンドボックスやデバッガを利用した解析環境やハニーポットを構築するために使用されることが多い。そのため、攻撃者は仮想環境を攻撃観測環境または解析環境の特徴として扱うことが多い。

Anti-VirtualMachine（以降、Anti-VM）技術は、耐解析技術の一つであり、仮想環境を検出する技術である。仮想化を支援するソフトウェアは多く存在し、ソフトウェアごとに特徴的なフィンガープリントが存在する。その中でも VMware 社の VMware Workstation Player や Oracle 社の VirtualBox は多くのセキュリティ従事者に利用されているため、Anti-VM 技術うの検出対象として実装されることが多い。Anti-VM 技術に利用される技術等の動向 [11] では、FFRI Dataset 2016 [12] に含まれる 8243 のマルウェア検体を分析し、Anti-VM 技術が対象とする仮想化ソフトウェアとフィンガープリントを調査し、まとめている。

例として、Anti-VM 技術の一般的な検出方法と検出対象を述べる。

一つ目は、エミュレートされた H/W の特徴が挙げられる。仮想化ソフトウェアでは、H/W をエミュレートする基盤を提供している。

仮想化された OS であるゲスト OS とホスト OS の間に仮想的なネットワークにが作成される。ゲスト OS の Network Interface Card（NIC）には、物理的な NIC と同様に MAC アドレスが付与される。MAC アドレスは 32bit で構成され、前半 16bit は提供企業の商品であることを示すベンダコードが与えられる。仮想化ソフトウェアによってエミュレートされた NIC では、ベンダーコードには仮想化ソフトウェア提供会社のベンダーコードが付与される。これがフィンガープリントとなる。

以下に、エミュレートされた NIC に付与される MAC アドレスのベンダーコードを記す。

- 00:05:69（VMware）
- 00:0C:29（VMware）
- 00:1C:14（VMware）
- 00:50:56（VMware）
- 08:00:27（VirtualBox）

- 00:03:FF（Hyper-V）
- 00:15:5D（Hyper-V）

そのほかにも、OS 内部から確認可能な外部記憶装置や演算装置、入出力デバイスなどの情報からフィンガープリントを確認することができる。

また、CPU には CPU に関する情報を得るために cpuid という命令が存在する。これを用いることでもフィンガープリントを確認することができる。

この命令を実行する際にはアセンブリを使用する。実際には、EAX レジスタに任意の命令番号を与え実行することで、EAX と EBX、EDX と ECX の 4 つのレジスタに結果が出力される。

cpuid による検出は 3 種類ある。一つは、cpu ベンダーの調査。二つ目は、OS 環境がゲスト OS か否かを示すフラグ。三つめは、仮想化ソフトウェアベンダーの調査である。

一つめは、EAX に「0x0」を入力することで調べることが可能である。すると、EBX に先頭 4 文字、EDX に中 4 文字、ECX に終端 4 文字のビット列が出力され、これを ASCII 文字として並べることでベンダーの判別が可能である。VMware 製品でこれを実行した場合、「VMwareVM」という文字列が表示される。二つめは、EAX に「0x1」を入力して実行する。これによって、cpu の持つ機能を示すフラグが得られる。その中で、ECX に出力されるビット列の 31bit 目は OS がゲスト OS であることを示すフラグとなっているため、これが「1」であれば調査環境はゲスト OS であると判別できる。最後に、EAX に「0x40000000」を入力して実行することで、仮想化ソフトウェアのベンダー文字列を得ることができる。VMware 社の製品であれば「VMwareVMware」、Hyper-V であれば「Microsoft HV」という文字列が出力される。

そのほかにも、VMware ではゲスト OS とホスト OS との間で行う通信のために、Backdoor I/O Port と呼ばれるインターフェースを構築する。

二つ目は、仮想化ソフトウェアの S/W 的特徴がある。仮想化ソフトウェアが正常に動作するためのソフトウェアを、ゲスト OS 内部に保持している。例えば、VirtualBox 上に構築された Windows10 のゲスト OS には、以下のディレクトリに様々なソフトウェアファイルが構築される。

List 2.1: VirtualBox のファイルディレクトリ

	特徴的なプロセス例
VMware Workstation Player	vmtools.exe, vmwaretrat.exe, vmwareuser.exe vmachlp.exe, vmhgfs.exe, vmmouse.exe, vmscsi.exe
VirtualBox	vboxservice.exe, vboxtray.exe

表 2.3: 仮想環境特有のプロセス例

C:\textbackslash Program Files\textbackslash Oracle\textbackslash VirtualBox

また、動作しているプロセスを確認することで検出が可能である。仮想化ソフトウェアは、仮想化を正常に動作するために特定のプロセスを動作している。その例を、表 2.3 に記す。

また、Anti-VM 技術にはゲスト OS を検出する手法の他に、ホスト OS を検出する手法も存在する。これは、ゲスト OS のファイル検出と同様に、特定のディレクトリやファイルの存在を確認することで検出が可能である。

通常、ゲスト OS 上に構築された攻撃観測環境や解析環境では、Anti-VM 技術に検出されないようにプロセス名の変更などの、フィンガープリント隠蔽処置が施されていることが多い。しかし、ホスト OS は侵入や調査されることを前提としていないため、それらフィンガープリントの隠蔽がなされていないという違いも存在する。

Anti-VM は、上記のような情報を元に仮想環境の検出を行うこと技術であり、攻撃者はこれを用いてマルウェアに自己防衛処理を付与する。

2.5.2 Anti-Honeypot 技術

2.4 節で述べたように、ハニーポットは攻撃者を誘引・観測し、分析に役立つログを記録するセキュリティツールである。

仮想化ソフトウェア同様、様々な種類のソフトウェアが存在し、それぞれにフィンガープリントが存在する。

ハニーポットは、攻撃者の行動記録や使用ツールを保持するために、ロギング機能が実装されている。また、ハニーポットを踏み台とした二次被害を発生させ

ないために、通常利用では起こり得ない設定がされている場合もある。

また、ハニーポットには調査対象を限定し、特定の機能・対話のみをシミュレートする低対話ハニーポットが存在する。設置者の実装にもよるが、低対話ハニーポットの性質上、対応する動作が十分ではない。

その他にも、特徴的なロギング手法を検出することが可能である。Windows には、Windows API や Win32 API と呼ばれる API 群が存在する。様々なソフトウェアでは、この API 群をまとめた動的リンクライブラリ（DLL）や静的リンクライブラリを利用することで簡単な操作を可能にしている。ハニーポットでは、これを仲介するライブラリを作成し、利用することで API の利用履歴の保持を行なっている。また、Sebek Honeypot では、システムコールレベルで操作記録を書き出し、ネットワークを介して外部へログ情報を送信・記録している。この様に、呼び出されたライブラリの数やメモリの変化を調査することで検出が可能であり、Sebek Honeypot ではネットワーク状況の変化やメモリへの負荷によって検出が可能である。

上記のような特徴的な動作や設定がフィンガープリントとなる。攻撃者はこれを検出することでハニーポットを回避している。

2.5.3 Anti-SandBox 技術

サンドボックスは、脅威情報が不明なファイルやコンテンツの調査・解析を行うセキュリティツールであるため、ハニーポットと同様のロギング機能を持っている。サンドボックスは、ハニーポットのように攻撃行動の監視を目的としておらず、マルウェアの動作と脅威の分析を目的としている。そのため、調査対象の挙動を記録するロギング機能と事前にシグネチャを設定し検出する機能が備わっている。他のセキュリティツールと同様に、サンドボックスは Cuckoo Sandbox [13] や Windows Sandbox [14] など多種存在する。

サンドボックスを検出する Anti-Sandbox 技術は、脅威を調査するためのシグネチャや設定ファイルがフィンガープリントとなる。そのほか、Anti-Sandbox 技術は Anti-Honeypot 技術に似たロギング技術が使用されるため、これもフィンガープリントとなる。

2.5.4 Anti-Debug 技術

一般に、デバッグとはコンピュータプログラムに存在する欠陥を探して取り除くという意味がある。しかし、本研究で言及するデバッグはバグの修正ではなく、リバースエンジニアリングのことを指す。

マルウェアを静的解析する場合は解析ソフトウェアを使用する。

解析ソフトウェアを検出する技術である Anti-Debug 技術は、解析用ソフトウェアやプロセス情報などがフィンガープリントとなる。

そのほかにも、以下のような解析ソフトウェアを検出する手法も存在する。

- IsDebuggerPresent()
- PEB の BeginDebugged フラグ
- PEB の NtNtGlobalFlags フラグ
- CloseHandle() による例外処理
- OutputDebugStringA() の戻り値
- 解析ソフトウェアのウィンドウクラス
- H/W ブレークポイントの検出
- 実行時間計測による検出

例として、Windows NT の OS では Process Environment Block（以降、PEB）に存在する、BeingDebugged フラグを確認することで解析ソフトウェアの検出が可能である。PEB は、OS 上で動作している全てのプロセスに与えられるデータ構造である。基本 OS 以外での使用は考えられておらず、カーネルがプロセス動作時に PEB を付与している。BeingDebugged フラグを確認する方法には複数手法が存在する。一つは、BeingDebugged フラグを直接確認する手法である。もう一つは、Win32 API を利用する手法である。Win32 API を使用する手法では、IsDebuggerPresent() という API を利用することで容易に BeingDebugged フラグを確認可能である。

解析ソフトウェアには、特定区間のプログラムが正常に動作しているのかを確認するためにブレークポイントという機能が存在する。ブレークポイントは、任

意のポイントでプログラムを停止させることや任意のポイントを処理しないようにすることが可能である。解析者は、この機能を多用して対象の解析を行う。これを使用する場合、ブレークポイント毎に対象の解析を行うため、ブレークポイント前後で時間が経過してしまう。Anti-Debug 技術の実行時間計測による検出では、不連続的に実行時の時間を記録することで、実行時間計測による解析ソフトウェアの検出が可能になる。

Anti-Debug 技術では、上記のようなフィンガープリントを検出することで、解析の回避を行なっている。

2.5.5 Anti-Disassembly 技術

Anti-Disassembly 技術は、リバースエンジニアリングを難しくする技術であり、それ自体に解析ソフトウェアの検出機能はない。これは、リバースエンジニアリングによる解析はあると言う前提で、コードの解読を妨害し、困難にする技術である。

既存の解析ソフトウェアが逆アセンブルする際には、各々に導入された逆アセンブリアルゴリズムを用いている。そのほかにも、デバッガで使用されるアルゴリズムには線形逆アセンブル方法やフロー志向型アセンブルなどがある。マルウェア製作者は、アルゴリズムが持つ脆弱性を利用して解読を困難にしている。

例えば、線形逆アセンブルでは、実行ファイルの内容が命令とデータがワンセットになって記述されていることを前提としている。また、条件分岐などがあった場合でも 1bit 目から順に逆アセンブルするため、複雑な条件分岐やエラー除外、ポインタを使用するコードは非常に可読性が低くなる。

また、パッカーと呼ばれる実行ファイルを圧縮する技術を施されたものや、暗号化を施すものも存在し、マルウェア製作者はこれらの技術を使用して解読を困難にしている。

第3章　先行研究と関連技術

本章では、難解を用いた欺瞞的防御と擬態を用いたサイバー防御に関する先行研究と先行研究をまとめる。

3.1　難解を用いた欺瞞的防御

3.1.1　APT 攻撃の対策のための欺瞞的防御

角丸らの研究では、APT 攻撃の対策を行うための欺瞞機構設計方針を行っていた。また、2 章で述べたように、欺瞞的防御の分類も行なっている。

標的型攻撃対策に向けた欺瞞機構を用いた防御アーキテクチャ

APT 攻撃は特定の組織や企業に対して行う高度で継続的な標的型攻撃であり、2009 年ごろから観測され始めた。このサイバー攻撃は、攻撃開始から終了までの間で攻撃を検出することが非常に難しく、被害が出てから攻撃に気づく場合や、そもそも攻撃に気づかない場合もある。このような現状に対し、角丸らは、APT 攻撃を 9 つのフェーズに分割し、各フェーズごとに対策を行う縦深防御を提唱していた。

角丸らが行った欺瞞的防御の分類では、サイバー空間における欺瞞の定義と欺瞞を用いた防御の分類を行なっている。角丸らは、サイバー空間における欺瞞を、相手の目的や意図をコントロールするための情報を与える行為であると定義している。

欺瞞的防御の目的では、誘惑、飽和、探知の 3 つを挙げている。誘惑では、攻撃者の真の目標物から、偽の目標物に対象を変更させる目的がある。飽和は、攻撃者の攻撃を偽の標的に飽和させることで他の対象への攻撃の可能性や余地を低

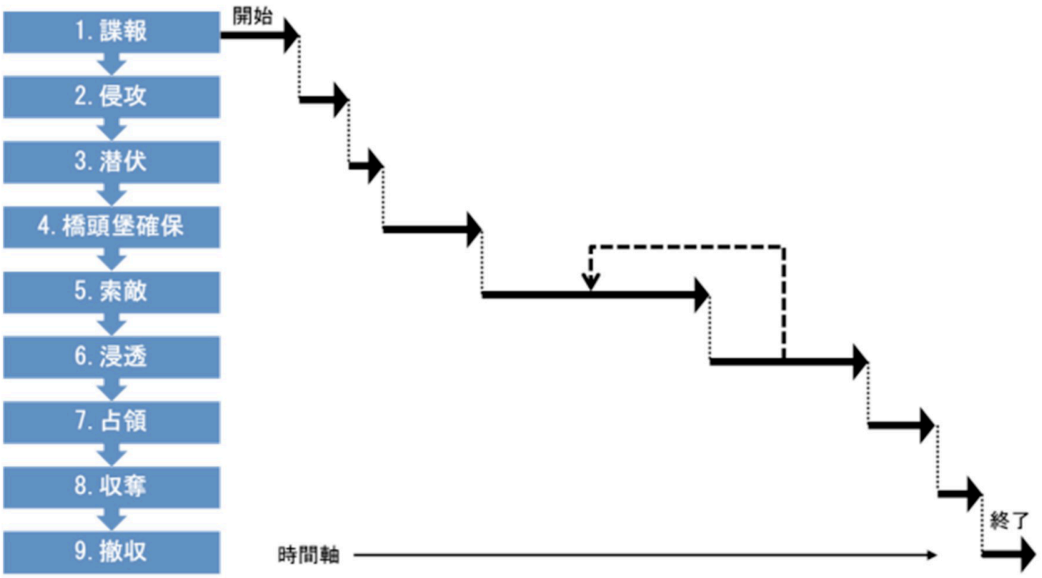

図 3.1: 縦深防御
（著者の論文より抜粋）

減させることを目的としている。探知は、攻撃者を偽の標的に誘導することで攻撃者の存在を明確にすることを目的としている。

欺瞞的防御に相当する既存のサイバー防御の割り出しと分類では、既存のサイバー防御手法を欺瞞的防御の観点から 3 種の欺瞞的防御に分類している。著者の分類の対象となる技術は、攻撃者の注意を引きつけるハニーポットの様な囮システムが対象となっている。

組織ネットワークにおける内部攻撃に対する模擬的欺瞞方式 [15]

角丸らは上記の研究に加え、模擬的欺瞞の提案を行なっている。

攻撃者が組織内ネットワークに侵入し、次の標的を探索する場合を想定している。APT 攻撃における索敵フェーズでは、ping や arp、ネットワークスニッフィングやネットワーク機器独自のプロトコルを使用する。

この様な現状に対し角丸らは、模擬的な手法で欺瞞を行なっている。ping 等の返答が存在するソフトウェアに対し、多数の宛先のない IP アドレスを返答することで実在する端末を希薄化することを提案している。索敵フェーズで入手した情報を元に、浸透フェーズにおいて攻撃者は次の攻撃対象を決定し、マルウェアの送信などを行う。角丸らの提案では、探索フェーズで難解の希薄を用いた欺瞞的

防御を行うことで攻撃を困難にする手法である。これにより、攻撃コストの引き上げが可能になっている。

3.1.2 欺瞞ネットワークの効率的な配置の評価

杉生らによる研究 [16] では、既存の研究以上に攻撃者を誘引可能かつ検知可能な欺瞞的防御を行うネットワーク配置の提案を行なっている。杉生らの提案は、難解の迷宮を用いた欺瞞的防御を提案している。

杉生らが提案した欺瞞機構では、以下の 3 つの機能を持つ。

- ネットワークへの閉じ込めと迷路化
- 認証情報の欺瞞
- 検知

ネットワークへの閉じ込めは、攻撃者は ssh 等を利用して欺瞞ネットワークのホストへ侵入した際に、ホストを踏み台にして他のネットワークへ移動できないように Firewall の設定を行うことで可能にしている。これにより、索敵・侵攻フェーズを妨害することを可能にしている。また、杉生らが行なった迷路化では、侵入者が自身の所在地を把握困難な構成をネットワーク規模で行っている。

認証情報の欺瞞は、HoneyToken を応用して実装されている。杉生らが設置した偽の認証情報は、欺瞞ネットワークに属するハニーポットの IP アドレスを多数設置していた。これにより、認証情報の記録に希薄の難解を用いた欺瞞的防御を実装している。

また、ハニーポットへのログインが実行された場合にはアラートを IDS サーバに送信することで、攻撃者検出を可能にしている。

3.2 擬態を用いた欺瞞的防御

3.2.1 Fake Honeypots: A Defensive Tactic for Cyberspace

Neil Rowe らによる研究 [17] では、サイバーセキュリティで擬態を行うことを提唱している。Fake Honeypots とは、通常のホストに対し、ハニーポットが持つ

フィンガープリントを埋め込むことで実現するサイバー防御システムである。これにより、通常のホストがハニーポットか否かの判別が困難になる。Neil Rowe らは、サイバー空間における簡単な擬態の基準、フィンガープリントを事前実験によって定め、実装を行なっている。まとめでは、攻撃者が攻撃を躊躇する環境は、本物の情報と偽の情報の割合が半々の時であると結論付けている。また、組織内ネットワークにおいて重要なシステムを担うホストにのみ実装することで、最大の効果を発揮すると提唱している。

このアプローチは本研究の元となっているが、サイバー空間における擬態の定義が不十分である。特に、サイバー空間において擬態を行う場合、整合性がなければ非常に稚拙なものとなってしまう。また、擬態の基準にはファイル名だけでは少なく、その他にも多くの要素が存在する。

3.2.2 Mimicry honeypot

2.4 節で述べた通り、Leyi Shi らによる研究であり、Leyi Shi らは警告色擬態と保護色擬態の提案と Neil らの Fake Honeypots を利用したハニーポットシステムを提案している。

Leyi Shi らが行なったサイバー空間における擬態分類の提案では、その名称と具体例の例示であり、定義はなされていなかった。

また、Leyi Shi らが提案する他の研究 [18] では遺伝的アルゴリズムを適用したハニーポットと複数のハニーポットを用いたシステムを提案している。

3.2.3 Stopping Malware With a Fake Virtual Machine

これは Thomas Rowe による記事 [19] である。

2.5.1 節で述べた様に、マルウェアには解析を阻害する機能が実装されている。著者は Anti-VM 技術に注目し、windows 環境下で仮想環境のフィンガープリントをエミュレートしている。著者が定義した仮想環境のフィンガープリントは、レジストリとプロセス、ファイルと MAC アドレスであった。

また、著者によって実装された環境に対し、OSS の仮想環境検出ツールである pafish を利用して実装環境の評価を行なっている。pafish は一部のマルウェアに耐解析機能として実装されているソフトウェアである。これによって全てのマルウェ

アを撃退できるとは言えないが、実際に多くのフィンガープリントが検出されているため、これらを検出する耐解析機能を備えたマルウェアは撃退できると考えられる。

攻撃者はマルウェアを解析されることを嫌うため、マルウェアには網羅的にフィンガープリントを検出する耐解析機能が施されていることが多い。あえて検出させて防御するだけであれば、これに合わせた様々なソフトウェアのフィンガープリントをシミュレートすることで防御が可能になる。しかし、この実装方法では、通常ではありえない環境のエミュレートになってしまう。つまり、著者の提案では、実装する耐解析機能に一貫性がないと言う問題がある。

3.2.4 Honeyfiles: deceptive files for intruction detection

Jim Yuill ら [20] は、端末に存在するファイルを防御するために、サイバー防御に欺瞞を取り入れた。端末に存在するファイル名に酷似した偽のファイルを多数保持することで、欺瞞的防御の希薄化を実現している。また、偽のファイルに侵入検知機能を付与することで、欺瞞的防御を IDS への拡張を行なっている。実際には、Honeyfiles と名付けられたこのファイルは、様々なディレクトリに多数設置される。ファイルが開封された場合にのみ、エンドユーザへの通知を送信する仕様となっている。

しかし、Honeyfiles にはいくつか弱点が存在する。例えば、攻撃者がキーロガーを使用している場合は、Honeyfiles の判別ができてしまう。また、著者は言及していないが、ファイルの変更の有無などを調査することでも Honeyfiles の判別が可能である。

3.2.5 セキュリティ無効化攻撃を利用したマルウェアの検知と活動抑止手法の提案

これは松本らと寺田らによる研究 [21] である。

マルウェアには、耐解析機能の他にアンチウィルスソフトの機能を無効化する機能である Anti-アンチウィルスソフトウェアが実装されている場合がある。松本らはこの機能をマルウェアのセキュリティ無効化攻撃と定義し、これに注目した

IDS を提案している。Jim Yuill らの Honeyfiles に似ているが、検出対象が異なる。Honeyfiles では、端末に侵入した攻撃者を対象としているのに対し、松本らの研究では、マルウェアを対象としている。

3.2.6 マルウェアの耐解析機能を逆用した活動抑止手法の提案

セキュリティ無効化攻撃を利用したマルウェアの検知と活動抑止手法の提案と同様、松本らと寺田らによる研究 [22] である。

この論文では、マルウェアに実装される耐解析機能の一つである Anti-Debug 技術に注目している。松本らは Windows の Anti-Debug 技術の手法の一つである、IsDebuggerPresent API をフックすることで、仮想的にデバッグソフトウェアの存在をエミュレートしている。これにより、デバッグ状態でなくてもデバッグ状態であるようにマルウェアに認識させている。

このアプローチは、マルウェア製作者の自己防衛を逆手に取った手法である。

3.2.7 欺瞞を用いた能動的サイバー攻撃防御手法の提案と実装

これは山田らによる研究 [23] である。

脆弱なシステムを検出し攻撃対象を決定する攻撃者は、Nmap を使用する場合が多い。Nmap は、一般的なポートスキャナー機能に加え、OS の種類だけでなく OS バージョンやサービスの種類とサービスバージョンの特定が可能なアプリケーションである。ポートスキャンを検知した場合の一般的な対処はパケットを破棄することであるが、著者はパケットを破棄した後に偽のメッセージを返答している。これを実装したホストから返答メッセージを受信した攻撃者はこの情報を元に攻撃または退避を行うため、攻撃が失敗する確率が高くなる。

山田らは、この提案を欺瞞と表記しているが、実際の端末情報を偽の端末情報にすり替えているため、欺瞞ではなく擬態であると考えられる。

第4章　サイバー空間における擬態の分類

自然界において、擬態とは対象となるモノや環境に似た特徴を持つことで、個を個としての判別を困難にさせる行為を指す。また、環境に擬態することで個としての認識を困難にする隠蔽的擬態と、目立つことで攻撃者を欺く標識的擬態などが存在する。これと同様に、サイバー空間で擬態を用いた欺瞞的防御を行う場合、擬態対象のフィンガープリントを把握し、再現または隠蔽する必要がある。

本章では、擬態を用いた欺瞞的防御を構築するための前段階として、欺瞞と擬態について整理し、各環境において擬態に必要な要素の分類表を作成した。

4.1　擬態を用いた欺瞞的防御の整理

先行研究では、擬態による欺瞞的防御を保護色と警告色の2種に分類する研究と、4種の擬態に分類するフレームワークについて述べた研究があった。

保護色の擬態について、先行研究では自身の持つ脅威情報を隠蔽し、ハニーポットの様に振る舞う擬態と記載されていた。これは、周囲の環境または一般的な環境に擬態を行なっているため、自然界における保護色の定義と類似しているが、その目的が明確に異なる。自然界における保護色は自身の保身を目的としているのに対し、既存研究が定義したサイバー空間における保護色では、自身への誘引を目的としている。先行研究で説明している警告色擬態は、第三者が脅威とみなす特徴を再現することで欺いているため、自然界のそれと近しい意味を持つ。

しかし、2種の先行研究の述べた分類手法では分類できない擬態が存在する。自身がもつ重要情報の特徴を変更し、価値のない情報であるように認識させる擬態が存在する。また、通常のファイルであるが、第三者に対して他のファイル以上に重要である様に認識させる擬態も存在する。つまり、擬態によって周囲の環境に対して何らかの効果を与えている。先行研究で定義されている保護色と警告色

表 4.1: 擬態カテゴリ

	詳細	効果	目的
保護型	第三者の目標物にのみ存在する特徴を隠蔽すること	自身の価値の低下	防衛
認識型	第三者の目標物にのみ存在する特徴を再現すること	自身の価値の上昇 周囲の価値の低下	おとり
隠伏型	第三者が脅威とみなす特徴を隠蔽すること	脅威レベルの低下	おとり
警告型	第三者が脅威とみなす特徴を再現すること	脅威レベルの上昇	防衛

擬態は、絶対的な観点から自身の持つ情報の脅威レベルを変更している。これに対し、のちに述べた 2 種の擬態は、相対的な観点から自身の重要性を変化させることで擬態を行なっている。つまり、擬態によって自身の属する環境にネガティブまたはポジティブな印象を与えている。

以上から本研究では、先行研究の分類手法では不十分であると考え、目的と効果を軸に再定義を行なった。

本研究では表 4.1 に記したように、4 つの擬態のカテゴリを定義し、その目的と自身または周囲への効果についてまとめた。

本研究では、サイバー空間における保護型擬態を自然界のそれに近しい定義付けを行なった。そのため、サイバー空間における保護型擬態は、自身の価値を低下させることで、周囲に存在する情報を自身と同様またはそれ以上の価値であるように見せる擬態手法と定義した。よって、本研究で述べる保護型擬態は、先行研究で述べられた保護色とは異なることに注意すべきである。

これに対して認識型擬態は、自身の情報の価値が他の情報の価値以上に見せることで、周囲の情報の価値を低下させる手法と定義した。保護型擬態は、自身にネガティブな変更を加えることで周囲の情報にポジティブな印象を与えている。逆に、認識型擬態は、自身にポジティブな変更を加えることで周囲の情報にネガティブな印象を与えている。

警告型擬態は、先行研究で言及されている定義と同様に、第三者が脅威とみなす特徴を再現することで実現される擬態手法である。

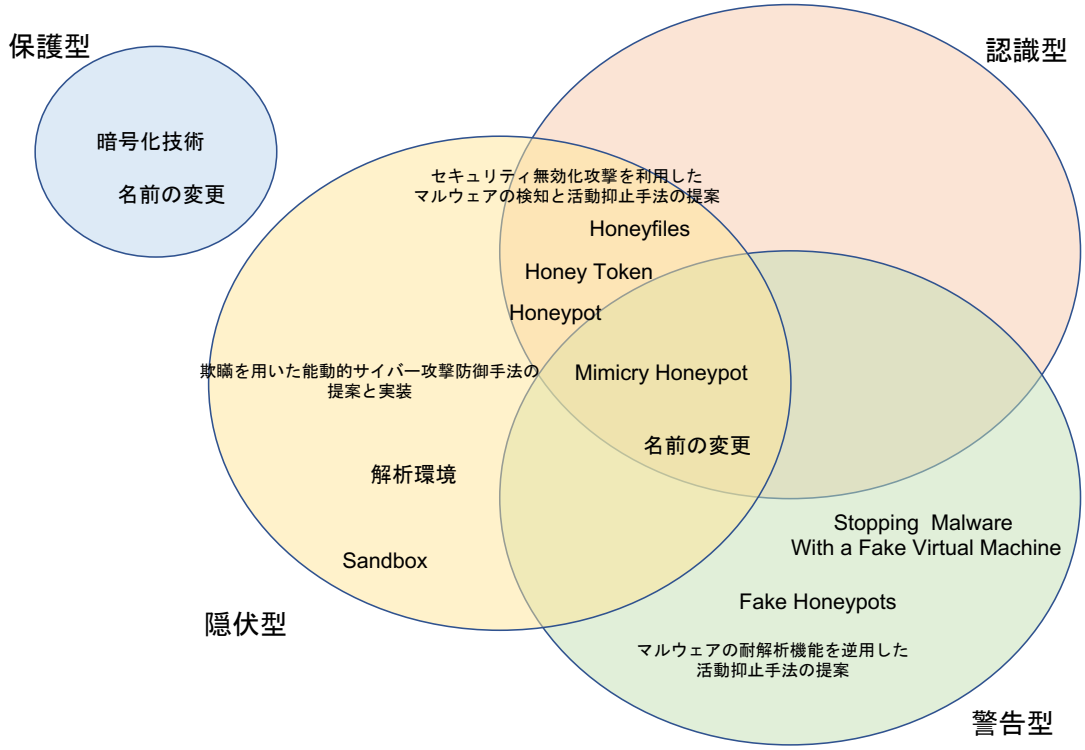

図 4.1: 擬態を用いた欺瞞的防御に関する先行研究の分類

本研究では、上記のようなサイバー空間における擬態を 4 つのカテゴリに分類した。

また、上記のカテゴリを元に擬態を用いた欺瞞的防御に関する先行研究と関連技術を分類した（図 4.1）。

情報の暗号化技術は、通常状態ではデータの解読が不可能な対象に変更を加えるため、保護型擬態に分類した。

ハニーポットや Honey Token、Honeyfiles などの技術は、第三者に対して自身の存在を示す認識型擬態と、IDS のような検知機能を隠蔽する隠伏型擬態の特徴を持っているため、認識型擬態と隠伏型擬態の共通部分に属する。

また、各データにつけられる名前は識別子の役割を担っている。つまり、データの名前を価値のない名前に変更することでも擬態が可能である。これは全ての技術に対して応用が可能であるため、全てのカテゴリに属する。

また、本分類手法は攻撃者防御者問わず、擬態技術の分類が可能である。攻撃者の使用する技術の場合、「第三者」を「防御者」と読み替える必要があることに注意したい。攻撃者の用いる技術には、第三者に対する脅威を隠蔽する技術が存

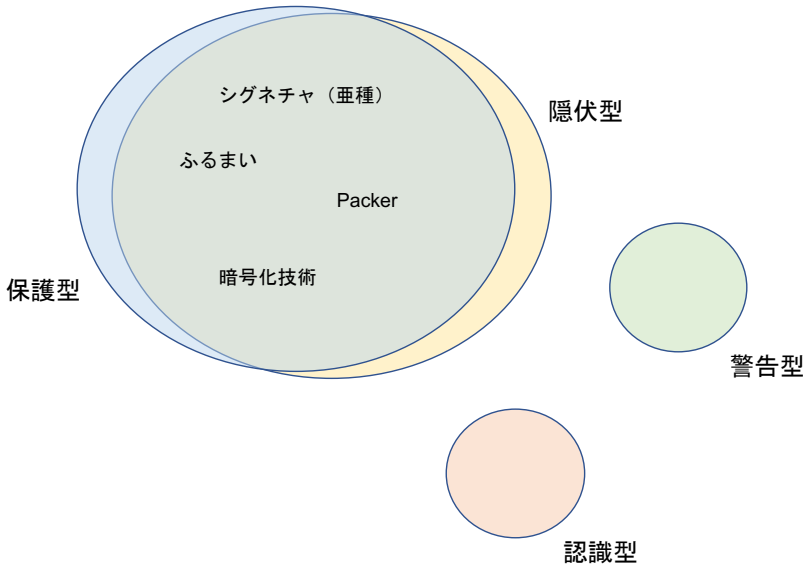

図 4.2: 攻撃者の用いる擬態技術の分類

在する。例えば、ソフトウェアの圧縮と暗号化、難読化を目的としたツールである Packer はこれに該当する。図 4.2 では、攻撃者の用いる技術を分類した。

Anti-VM 技術や Anti-Debug 技術などの耐解析技術は、任意のフィンガープリントを調査し、特定の環境を検出する技術である。この技術は、情報の隠蔽ではなく検出を目的としているため、この分類には含まれない。

4.2 擬態構築フロー

擬態の構築を行う際、図 4.3 のような 5 つのステップ（目的の整理、コンテンツの決定、擬態型の選択、フィンガープリントの整理、擬態の実装）を踏んで実装が行われる。これを擬態構築フローとする。

擬態を実装するにあたり、その目的を整理する。今回は一例として、ハニーポットの設置を想定する。ハニーポットの存在は攻撃者にとっては脅威となるため、攻撃者は Anti-Honeypot 技術を用いてこの検出を行う。フィンガープリントを隠蔽しない場合、攻撃者はフィンガープリントを検出し、ハニーポットを回避してしまう。そのため、今回の目的は、攻撃者に検出されないハニーポットの構築である。

使用するコンテンツの決定では、使用するハニーポットツールの具体化を図る。公開されているハニーポットは多種存在し、各ハニーポットツールによって異な

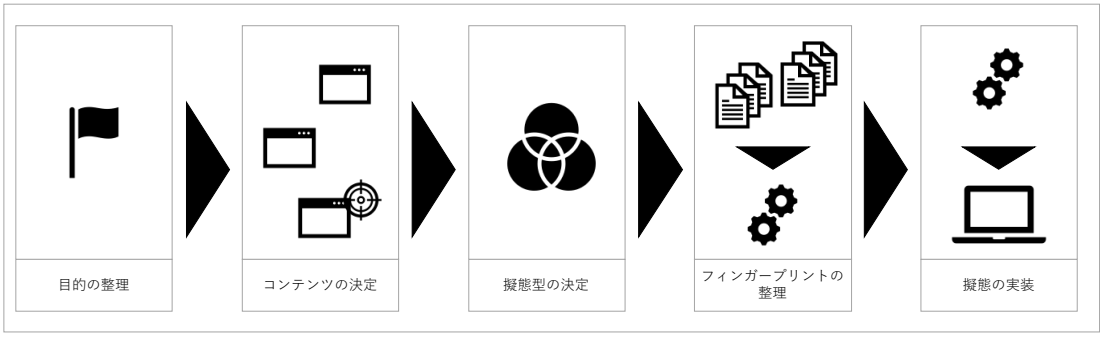

図 4.3: 擬態構築フロー

るフィンガープリントが存在する。そのため、使用するツール、つまり擬態対象となるツールの決定を行う。

擬態カテゴリの決定では、対象となるコンテンツに適用する擬態型を決定する。このステップは目的の整理ステップで決定される。今回の例の目的は、攻撃者に検出され辛いハニーポットの構築であるため、隠伏型擬態が適切である。

フィンガープリントの整理では、使用するコンテンツに内在するフィンガープリントの洗い出しを行い、擬態の実装ではこれを元に各情報の隠伏型擬態を図る。

本研究の提案は擬態構築フローのフィンガープリントの整理に位置し、フィンガープリントの分類（擬態要素）と詳細化を図り、擬態要素分類表を提案する。

4.3 本研究の位置付け

自然界で隠蔽的な擬態を行う場合、擬態対象の特徴を複数個隠蔽する必要がある。対して、外部への警告を行う標的型擬態の場合は、一つ以上の特徴的な要素を再現することで警告可能な場合も存在する。サイバー空間において擬態を行う場合であっても、擬態対象の特徴を再現する必要があり、それは環境ごとに様々である。

しかし、APT 攻撃等の標的型攻撃では、長期にわたり検証を行うため、網羅的な擬態の検出は容易に可能である。つまり、網羅的または単純な擬態では、擬態による十分な効果は得られない。

対象環境への巧妙な擬態を行う場合には、擬態対象の様々な特徴に注目し、それを再現または隠蔽する必要がある。擬態対象の特徴となる情報は、擬態対象となるコンテンツの設置または動作時に現れる。例えば、コンテンツ A をインストー

ルしたとする。その際、コンテンツ A を動作させるための実行ファイルが生成される。また、これをの動作を補助するためのドライバや設定ファイルの生成や、レジストリへのキー登録が発生する。動作時には、そのプロセスやサービスが動作し、OS に対して何らかの変化を生じさせることもある。これらがコンテンツ A の特徴となり、つまりはフィンガープリントである。

このようなフィンガープリントは擬態対象とするコンテンツや環境ごとに様々である。また、このフィンガープリントはコンテンツに対して一意であるため、擬態を行う際の要素になる。

本研究では、サイバー空間において擬態を行うにあたり必要な要素と条件についてまとめ、これを用いた擬態要素分類表を提案した。

4.4 擬態レベルの指標

巧妙な擬態を行うにあたり、再現または隠蔽すべき擬態要素は対象ごとに様々である。前節で述べたように、サイバー空間における擬態には 4 種のカテゴリが存在する。擬態は、一つ以上の擬態要素を満たすことで実現可能である。

4.4.1 擬態要素

サイバー空間において擬態を行う場合、様々な要素が存在する。本研究では、これを擬態要素と呼ぶ。例として、本研究では Windows を対象とし、すべてのコンテンツに共通する擬態要素をまとめる。

以下が、Windows 上のすべてのコンテンツに共通する擬態要素である。

- ファイル・ディレクトリ
- プロセス・サービス
- レジストリ

ファイルやディレクトリでは、擬態対象の実行ファイル（.exe、.bin）やコンフィグファイル、デバイスドライバ（.sys）や動的リンクライブラリ（.dll）などがある。これは、ソフトウェアを正常に動作させるために必要なデータを保持するファイルや、そのデータの存在するディレクトリのことを示す。

プロセスでは、擬態対象のコンテンツのプロセスや、その動作を補助するプロセスなどがある。複数のコンテンツが依存する関係を持つ場合には、擬態対象だけでなく、依存関係にあるコンテンツのフィンガープリントも擬態する必要がある。サービスとレジストリは windows 特有の機構である。サービスは、カーネルが実行するプロセスのことであり、構築する環境によっては作成されない場合もある。

レジストリは、Windows 特有のデータベースであり、システムやアプリケーション、デバイスドライバ等の各種設定を記録するために存在する。この中に、サービスとして動作するプロセスの登録情報なども存在し、これがフィンガープリントとなる。

4.4.2 擬態条件

擬態要素には、それぞれ擬態のレベルを決定するための指標が存在する。プロセス（Real A）の擬態プロセス（Fake A）動作させるとする。プロセスが対象への擬態を行う場合、プロセス名が一致する必要がある。また、そのプロセスの元となる実行ファイルの存在パスも、擬態を行うためには必要な条件（擬態条件）となる。上記に示した 2 項目（名前、存在パス）は、擬態における絶対条件である。このように、擬態要素の擬態レベルを上昇させるための項目を、本研究では擬態条件と呼ぶ。

擬態条件の一例として、4.4.1 節で述べた擬態要素に対する要件を以下に述べる。

- 名前
- 存在パス

Fake A が Real A に擬態するために満たすべき擬態条件はこれだけではない。名前と存在パス以外にも擬態レベルを上昇させる擬態条件が存在する。

プロセスの場合、プロセスのメモリ使用サイズや動作（操作コマンド）などがこれに該当する。以下にその必要条件を記す。

- メタ情報
- 動作

- 内容

内容に関しては、プロセスには存在しない。しかし、擬態要素がファイルであった場合、ファイルやレジストリに記述されている内容も擬態条件となる。そして、すべての擬態要件と擬態条件を満たす状態は、そのプロセスが擬態対象そのもの、つまり本物のことを指す。

極端な例ではあるが、Real A のメモリ使用サイズは 50.0MB だったとする。対して、擬態を行う Fake A のメモリ使用サイズが 5.0MB であった場合、明らかに不自然である。

Real A が実行コマンドを持っていた場合、Fake A も実行コマンドを持たなくてはいけない。これが擬態条件の動作に該当する。

警告型擬態を行う先行研究では、絶対条件として示した名前と存在パスのみを再現することで擬態を行なっていた。そのため、擬態レベルは低い。また、より詳細に擬態を行う場合、メタ情報なども重要となる。

ファイルやディレクトリには、名前やサイズ等の情報を含めたメタ情報が保持される。メタ情報には、名前やサイズの他にも、作成日時や更新日時、作成者やアクセス権などがある。これらも擬態条件となる。特に更新日時は、サンドボックスやハニーポットのフィンガープリントとなる場合が多く、攻撃者によって検出される可能性がある。

このように、擬態を行う際に必要な擬態要素には、それぞれ擬態条件が存在し、それを満たすことで擬態レベルが向上する。

4.5 擬態要素分類表

巧妙な擬態を行うには、擬態要素と各擬態要素に内在する擬態条件を満たす必要がある。本研究では、擬態環境の擬態要素と擬態条件をまとめた表を、擬態要素分類表とした。

擬態要素分類表には、すべての擬態環境に共通する擬態要素と擬態条件をまとめた基本的な擬態要素分類表と、特定の擬態環境にしか存在しない擬態要素と擬態条件をまとめたユニークな擬態要素分類表が存在する。

そのため、擬態構築フローのフィンガープリントの整理では、基本的な擬態要素分類表の他にユニークな擬態要素分類表を作成する必要がある。

表 4.2: 基本的な擬態要素分類表

擬態要素		擬態条件						
		存在 (名前)	存在 (パス)	内容	動作	メタ情報	依存関係	その他条件
ファイル	テキスト ファイル	○	○	○	-	○	-	-
	システム ドライバ (sys)	○	○	○	○	○	実行ファイル	-
	モジュール (lib, dll)	○	○	○	○	○	実行ファイル	-
	実行 ファイル (exe, bin)	○	○	○	○	○	モジュール 独自ファイル	-
	独自 ファイル	○	○	○	○	○	実行ファイル システム ドライバ	-
プロセス	フォア グラウンド	○	○	○	○	○	-	ウィンドウ
	バック グラウンド	○	○	○	○	○	-	実行コマンド
レジストリ		○	○	○	-	○	-	-

4.5.1 基本的な擬態要素分類表

先に述べたように、基本的な擬態要素分類表はすべての擬態環境に共通する擬態要素と擬態条件をまとめた表を指す。基本的な擬態要素分類表は、4.4.1 節で述べた要素と条件を元に作成する。

表 4.2 では、基本的な擬態要素分類表をまとめた。

4.6 マルウェア忌避環境における擬態要素分類表

前節では、サイバー空間で擬態を行うにあたり必要な、基本的な擬態要素分類表を提案した。本研究では、マルウェアが対解析機能により回避対象の環境を検出し、忌避行動をとる環境を、マルウェア忌避環境とする。本研究ではマルウェア忌避環境を構築する際に関連する仮想環境と攻撃観測環境、解析環境を例として取り上げ、ユニークな擬態要素分類表を提案する。

4.6.1 仮想環境の擬態

仮想環境の擬態を行う場合、ホスト OS とゲスト OS の双方に対する擬態を考慮する必要がある。

表 4.3: 仮想環境（ホスト OS）の擬態要素分類表

擬態要素		擬態条件						
		存在（名前）	存在（パス）	内容	動作	メタ情報	依存関係	その他条件
ファイル	スナップショット	○	○	○	-	○	-	-
	仮想ドライブファイル	○	○	○	○	○	実行ファイル	-
	VM SSHキー	○	○	○	○	○	実行ファイル	-

2.5.1 節で述べた通り、ホスト OS もゲスト OS も多くのフィンガープリントが存在する。つまり、仮想環境へ擬態を行う場合は、このフィンガープリントを再現または隠蔽する必要がある。

H/W 仮想化ソフトウェアは、VMware 社の製品や VirtualBox、KVM や Hyper-V など、複数のベンダから提供されている。本研究では、セキュリティ環境の構築に多く利用されている VMware Workstation Player と VirtualBox、Hyper-V について考える。

仮想環境（ホスト OS）の擬態要素分類表

2.5.1 節で述べたように、通常仮想環境（ホスト OS）ではフィンガープリントを隠蔽しない。

仮想環境（ホスト OS）の特徴的なフィンガープリントとしてスナップショットがある。そのほかにも、ゲスト OS の操作に必要な SSH キーや、仮想 OS のイメージファイルである仮想ドライブファイルなどがある。

本研究では、仮想環境（ホスト OS）の擬態の要素を、表 4.3 にまとめた。

仮想環境（ゲスト OS）の擬態要素分類表

セキュリティツールとして構築される仮想環境（ゲスト OS）では、フィンガープリントが隠蔽される場合が多い。これは、Anti-Anti-VM 技術とも呼ばれ、Anti-VM 技術に対抗する技術である。しかし、攻撃者は Anti-Anti-VM 技術によって隠蔽されたフィンガープリントをさらに検出しようとする。

仮想環境（ゲスト OS）では、基本擬態要素に加え、以下の様なフィンガープリントがある。

表 4.4: 仮想環境（ゲスト OS）の擬態要素分類表

擬態要素		擬態条件				
		存在 (名前)	性能	動作	メタ情報	その他条件
システム	演算装置	○	○	○	○	CPUID命令 仮想化支援フラグ
	制御装置	○	○	○	○	-
	主記憶装置	○	○	○	○	-
	補助記憶装置	○	○	○	○	-
	入力装置	○	○	○	○	-
	出力装置	○	○	○	○	-
	NIC	○	○	○	○	MACアドレス
	その他 デバイス	○	○	○	○	-
その他	VMM命令	○	-	○	-	-
	バックドアI/O	-	-	○	-	-

- MAC アドレス
- H/W 情報
- CPUID
- BIOS 情報
- 仮想化ソフトウェアごとのフィンガープリント

システムは、H/W 情報から得れる擬態要素である。記憶デバイスや入出力デバイス、NIC 等がこれに該当する。また、CPU の提供企業や型番、OS バージョンなどもこれに該当する。

上記のような仮想環境（ゲスト OS）の擬態要素となる情報を表 4.4 にまとめた。

4.6.2 攻撃観測環境の擬態要素分類表

2.5.2 節で述べた様に、ハニーポットなどの攻撃観測環境は、攻撃者やマルウェアの行動を記録するためにロギング機能を持つ。ロギング手法には、内部的に記録するものやネットワークを介して外部デバイスに蓄積するものもある。

表 4.5: ハニーポットの擬態要素分類表（a）

擬態要素		擬態条件						
		存在（名前）	存在（パス）	内容	動作	メタ情報	依存関係	その他条件
ファイル	-	基本的な擬態要素分類表と同様						他のソフトウェアのインストール状況 使用履歴

表 4.6: ハニーポットの擬態要素分類表（b）

擬態要素	擬態条件	
ユーザ操作	入力操作	○
	出力装置の有無	○
	外部補助記憶装置	○
	インターネット接続性	○
ユーザ情報	ユーザ名	○
	ホスト名	○
	権限	○
その他	通信の有無	○

攻撃観測環境は仮想環境上に設置される場合が多い。そのため、仮想環境（ゲスト OS）上の攻撃観測環境に擬態する場合、攻撃観測環境と仮想環境（ゲスト OS）の間に包含関係が発生する。しかし、これらは独立して存在する環境であるため、攻撃観測環境と仮想環境の併用によって発生する特殊な擬態要素は存在しない。そのため、攻撃観測環境の構築に使用する要素のみ擬態を行う必要がある。

ハニーポットの擬態要素分類表

本節では、攻撃観測環境の例としてハニーポットの擬態要素となる情報を述べる。実際には、2.5.2 節で述べたフィンガープリントが、ハニーポットの擬態要素である。

表 4.8 と表 4.9 では、ハニーポットの擬態要素をまとめた。

4.6.3 解析環境の擬態

2.5 節で述べた様に、動的解析環境と静的観測環境の 2 種類存在する。本研究では、動的観測環境をサンドボックス、静的観測環境を解析ソフトウェアを用いた環境を例として挙げる。

動的解析環境の擬態要素分類表

サンドボックスは仮想環境（ゲスト OS）上に構築されるため、これらの間に包含関係が生じる。また、サンドボックス環境は攻撃観測環境の擬態と同様に、独立した環境であるためサンドボックスと仮想環境の併用によって発生する特殊な擬態要素は存在しない。

Anti-SandBox 技術には含まれないが、サンドボックスのシステム的特徴を利用した回避方法も存在する。その一つに、ユーザ操作履歴の有無がある。サンドボックスの特徴として脅威不明なコンテンツの挙動を調査するために、操作履歴の存在しないクリーンな状態で観測を開始する。そのため、サンドボックスにはユーザによる操作履歴がほぼ存在しない。また、サンドボックスは脅威の特定が目的であるため、脅威の判定が終了するまでユーザの入力やソフトウェア操作は行われない。その他、企業を標的としたマルウェアでは、特定のソフトウェアの存在や操作履歴を検出する場合もある。例えば、企業に存在する PC であれば存在するはずの Office ソフトウェアや、業界特有のソフトウェアなどが対象となる。

サンドボックスでは、一連の動作の観測が終了した際、または任意の時間が経過した後にログと検出結果のみを保持して初期化を行う。つまり、一定時間経過するまでの間にリセットを行う環境はサンドボックスである可能性が高い。これは、リセットまでの感覚を調節することでフィンガープリントの隠蔽が可能である。しかし、警告型擬態を行う場合はほぼ不可能となっている。その理由として、特定時間が経った後リセットを行う環境は外部記録デバイスへのデータの保持が必須となり、可用性に欠けているからである。

以上から動的解析環境では、ハニーポットの擬態要素分類表と似た擬態要素表になる。そのため、ハニーポットと異なる要素のみ抜粋し、表 4.7 にまとめた。

表 4.7: 動的解析環境の擬態要素分類表

擬態要素		擬態条件						
		存在（名前）	存在（パス）	内容	動作	メタ情報	依存関係	その他条件
ファイル	-	基本的な擬態要素分類表と同様						業務ソフトのインストール状況 使用履歴
システム	デスクトップ解像度	-	-	-	-	○	-	-

表 4.8: 静的解析環境の擬態要素分類表（a）

擬態要素		擬態条件						
		存在（名前）	存在（パス）	内容	動作	メタ情報	依存関係	その他条件
ファイル	-	基本的な擬態要素分類表と同様						他のソフトウェアのインストール状況 使用履歴

静的解析環境の擬態要素分類表

攻撃観測環境や動的解析環境と同様に、静的解析環境も物理マシン上に構築される場合と仮想環境（ゲスト OS）上に構築される場合の 2 種存在する。悪意や脅威がないことが判明しているコンテンツの場合、静的解析環境は物理マシン上に構築する。しかし、悪意や脅威のある可能性のあるコンテンツであれば、仮想環境（ゲスト OS）上に構築し、解析を行う。その理由として、仮想環境（ゲスト OS）は仮想環境（ホスト OS）からある程度隔離されているため、セキュリティ性の確保が可能であるからである。その他、スナップショットによる容易なバックアップが可能であるためである。

また、静的解析環境において、基本となる 4 つの擬態要素には静的解析ソフトウェアのプロセス名だけでなく、マルウェアプロセスの親プロセスもフィンガープリントとなる。

そのほかにも、2.5.4 節で述べたようなフィンガープリントが擬態要素となる。

上記のような静的解析環境における擬態要素となる情報を表 4.8 と表 4.8 にまとめた。

表 4.9: 静的解析環境の擬態要素分類表（b）

擬態要素	擬態条件	
ユーザ操作	入力操作	○
	出力装置の有無	○
	外部補助記憶装置	○
	インターネット接続性	○
ユーザ情報	ユーザ名	○
	ホスト名	○
	権限	○
その他	通信の有無	○

第5章　警告型擬態を用いたマルウェア忌避環境の設計と実装

本章までに、耐解析技術とサイバー空間における擬態の定義、擬態要素と擬態条件について述べた。これらを元に、本章では、ホストに侵入・実行された各マルウェアプロセスに対し、仮想的な擬態環境を提供し、忌避を促す手法の提案を行う。

5.1　対象とする状況

サイバー攻撃には、サービスの妨害を目的とするDoS・DDoS攻撃やサーバに不正な文字列を送信するインジェクション、RATやトロイの木馬を使用した攻撃など多種存在する。本研究では、メール添付機能やマクロウィルスを利用した攻撃の様に、マルウェア感染を発端としたサイバー攻撃を対象とする。

より詳細には、本研究が対象とする状況は、マルウェア感染を発端としたサイバー攻撃の中で、図3.1の侵攻と浸透の攻撃フェーズを対象とする。

5.2　構成モデルと機能

2.5節で述べた通り、マルウェアには耐解析機能が実装されている。本研究では、耐解析機能を逆用することで、マルウェアが忌避行動をとる環境の構築を行う。

図4.3を参考に、擬態環境の構築を図る。先に述べた通り、本章での目的は、マルウェアが忌避する環境への擬態である。

次にコンテンツの整理に入る。マルウェアが忌避行動をとる環境は、2.5節で述べている。例として、本研究では仮想環境のVMware Workstation PlayerとVirtualBoxを擬態対象のコンテンツとする。

擬態型の決定は、マルウェアが忌避行動をとるための環境の構築を目的するため、警告型の擬態を行う。

フィンガープリントの整理に関しては、4.3 節で述べた通りである。

これを元に、本章では警告型擬態の設計と構築を行う。

5.2.1 構成モデル

擬態環境の構築にあたり、2 種の手法が存在する。これらの構築手法は、環境を構築するレイヤと擬態の構築方法によって区別される。

情報生成モデル

擬態情報を実環境に設置する構築手法であり、ほとんどの先行研究がこのモデルに該当する。この手法の利点として、OS 機能の変更点が少ないことが挙げられる。また、事前に設置しておくことができるため、CPU やメモリに負荷を与えることなく環境の構築が可能である。

対して、情報生成モデルを用いて忌避環境を作成する場合は、定期的にファイルの更新を行う必要がある。一度生成した忌避環境構築用ファイルは、利用者によって利用されることを想定していない。そのため、ファイルの更新状況等を確認するマルウェアにとっては、フィンガープリントとして扱われる可能性が生じる。あえて利用者が定期的にアクセスすることで対応は可能ではあるが、その作業効率はいいと断言できない。また、擬態環境を変更する際には、忌避環境構築用ファイルやプロセスの除去と再配置を行う必要があるため手間がかかる。

仮想モデル

仮想モデルでは、擬態対象のフィンガープリントを実環境上に保持しない。これはプロセスとして動作し、任意のプロセスに対して任意の擬態環境を錯覚させる。これにより、実環境への影響は少なくなり、擬態環境の変更や更新は非常に容易であるという特徴を持つ。

5.2.2　警告型擬態を用いた仮想的なマルウェア忌避環境の機能

本節では、警告型擬態を用いた仮想的なマルウェア忌避環境に必要な 5 つの機能を述べる。

整合性の確保

本提案における整合性とは、実際のシステムとして成立する環境のことを指す。前節で述べたように、3.2 節で紹介した先行研究や技術では、複数の仮想化ソフトウェア上に存在するゲスト OS を構築していた。ゲスト OS A の特徴とゲスト OS B の特徴を持つ環境を実際に構築することは不可能である。この環境を、本研究では整合性の取れていない環境と呼ぶ。

耐解析機能を網羅的に実装されたマルウェアであれば、整合性の取れていない環境であっても十分な効果を得ることができる。しかし、忌避環境の整合性を調査するマルウェアが存在した場合、整合性の取れていない既存の警告型擬態の効果は期待できない。この点、耐解析機能を網羅的に実装されたマルウェアは、整合性の取れた忌避環境に対しても忌避行動をとるという利点がある。

整合性の取れた環境の場合、整合性の取れていない環境よりも擬態要素が少なくなるため、4.3 節で述べた擬態要素と要件を多く満たす必要がある。

耐解析機能の検出

マルウェアの侵入をリアルタイムで検出することは難しく、マルウェアとその他のソフトウェアを分別することは難しい。しかし、警告型擬態を開始するためには、そのトリガが必要になる。

また、マルウェアに実装される耐解析機能と耐解析機能の調査対象となる環境は、マルウェアごとに様々である。しかし、耐解析機能を逆用する場合、2 つの情報が重要となる。一つは、耐解析機能に使用された API は何か。もう一つは、その対象となるフィンガープリントは何かである。

上記の情報は、耐解析機能の利用の確実性を示すために使用されているほか、これをトリガーとして擬態環境の構築を行うことも可能である。また、これらの情報は未知の耐解析機能やフィンガープリントの特定などに応用することも可能である。

独立した環境の提供

耐解析機能の検出では、マルウェアプロセスの耐解析技術に応じて様々なアクションが可能であることを保証している。

マルウェアに実装されている耐解析機能は様々であり、整合性を調査するマルウェアでは耐解析機能の環境ごとに対応しなくてはいけない。複数のマルウェアがホスト内に侵入し、ほぼ同時に実行した場合も考えられる。この場合、ほぼ同時に動作する複数のマルウェアプロセスに対応することが求められる。

擬態履歴の保持

欺瞞的防御の目的は、攻撃コストを押し上げることで、攻撃行動の中止を誘うことである。しかし、欺瞞的防御だけでは、防御側からの攻撃の中止や切断を行うことはできない。そこで、擬態の履歴を記録することで、マルウェア侵入の検出やサイバーレジリエンスへの活用や、即座の対応が可能になる。

潜伏

全てのツールには必ずフィンガープリントが存在する。本ツールであっても同様に、プロセスとしてフィンガープリントとして生成される場合もあれば、ファイルとして生成される場合もある。

そのため、本ツールの存在を隠蔽する手法についても考察する必要がある。

5.3 設計

警告型擬態を行う先行研究では、情報生成モデルを用いており、事前にフィンガープリントを設置することで忌避環境を構築している。これらの先行研究や先行技術には、以下の様な問題がある。

- 擬態要素を網羅的に実装しているため、構築された忌避環境は整合性が取れない
- 擬態要素を実ホスト上に構築するため、擬態環境の変更が容易でない

- 特定の環境を検出するマルウェアが複数動作した際、各マルウェアへの対応が難しい
- 擬態条件は存在（名前・パス）しか対応していない

先行研究では、マルウェア忌避環境に存在するフィンガープリントを網羅的に実装している。網羅的な実装は、様々なマルウェアへの対応が可能であるのに対し、擬態によって構築された環境の整合性が取れなくなってしまう。そのため、擬態環境の検出が容易である。

先行研究を、網羅的な実装ではなく、整合性の取れた擬態環境として場合、情報生成モデルでは一つのマルウェアプロセスにしか対応できない。これでは、設置者が指定した整合性の取れた環境とマルウェアが検出する環境が一致しない場合が存在するため、十分な効果があるとは断言できない。耐解析機能を検出してから擬態環境の構築を行う場合であっても、2 つの問題が発生する。一つが、擬態環境の構築に時間がかかってしまうこと。もう一つが、複数のマルウェアがほぼ同時に動作した場合には 1 マルウェアプロセスにのみしか対応できないことである。図 5.1 に先に述べた問題のある状態を紹介する。

複数のマルウェアが侵入し、ほぼ同時に耐解析機能を実行した場合、先に実行されたマルウェアによって擬態環境が構築される。環境設置型の実装の場合、後に動作したマルウェアが耐解析機能を実行した場合、新規の擬態環境が生成され、整合性が取れない状態になる。整合性を重視し、1 組以上の擬態環境を生成しないように設定したとしても、マルウェアごとに擬態環境を構築することができない。

本研究では、既存研究の構築モデルである情報生成モデルではなく、仮想モデルに注目した。

各プロセスには、基本的にそのプロセスのみしか利用できないメモリ領域が Kernel によって割り当てられる。つまり、各マルウェアプロセスは独立した環境を持っている。本研究ではこれを利用し、各マルウェアプロセスに対し独立した忌避環境を提供することで解決を図った。

本研究では、仮想的な擬態環境を提供するレイヤを、本研究ではマスキングレイヤ（Masking Layer）として扱い、擬態のための基盤を考案した。

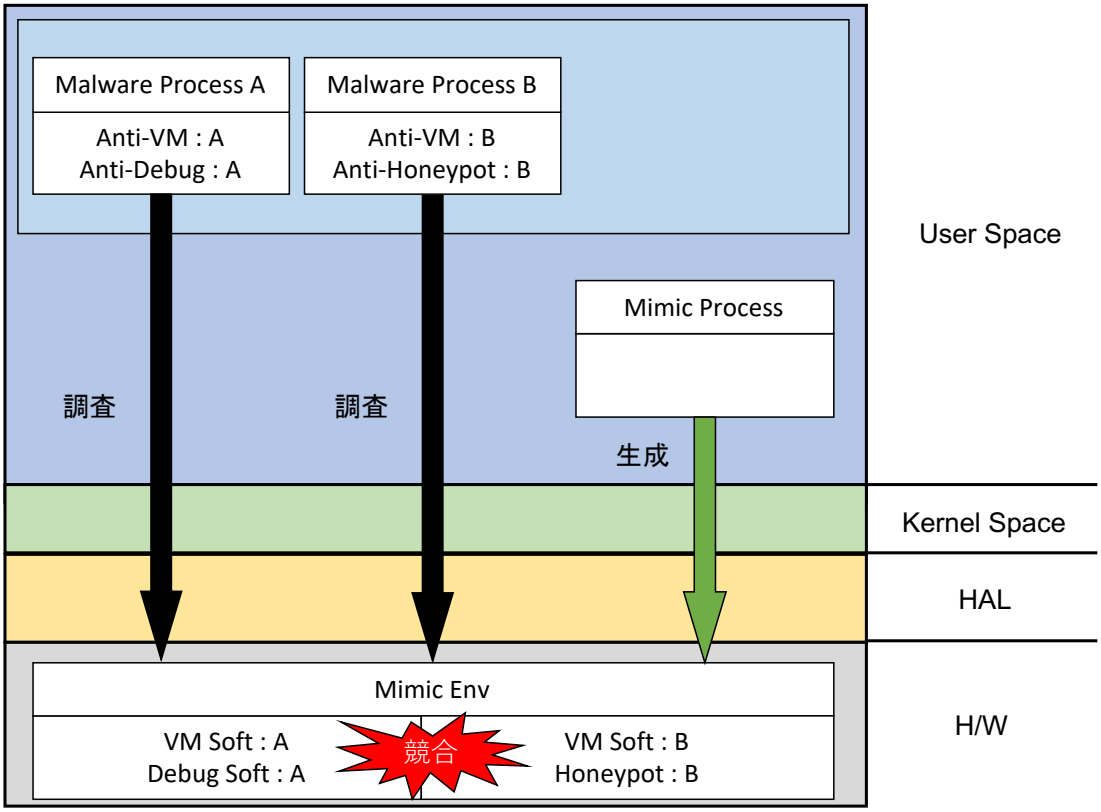

図 5.1: 情報生成モデルによる擬態環境の構築

マスキングレイヤは、マルウェアプロセスで実行される耐解析機能の情報を受け付け、これに合わせた擬態環境の構築と提供を行う。マスキングレイヤでは、マルウェアプロセスが動作している間、この擬態環境を提供し続ける。

マスキングレイヤの役割は以下のものが挙げられる。

- 擬態環境の提供
- 擬態環境の整合性の保証
- 実環境情報の隠蔽
- 耐解析機能以外の処理の通過

整合性の保証では、耐解析機能を利用するマルウェアに対し、特定の環境情報のみを提供し続ける。耐解析機能によって仮想環境 A を検出するマルウェアには、仮想環境 A のフィンガープリントを提供する。その後、同プロセスが仮想環境 B について調査した際には、仮想環境 B のフィンガープリントは提供しない。このようにして、擬態環境の整合性を保つ。

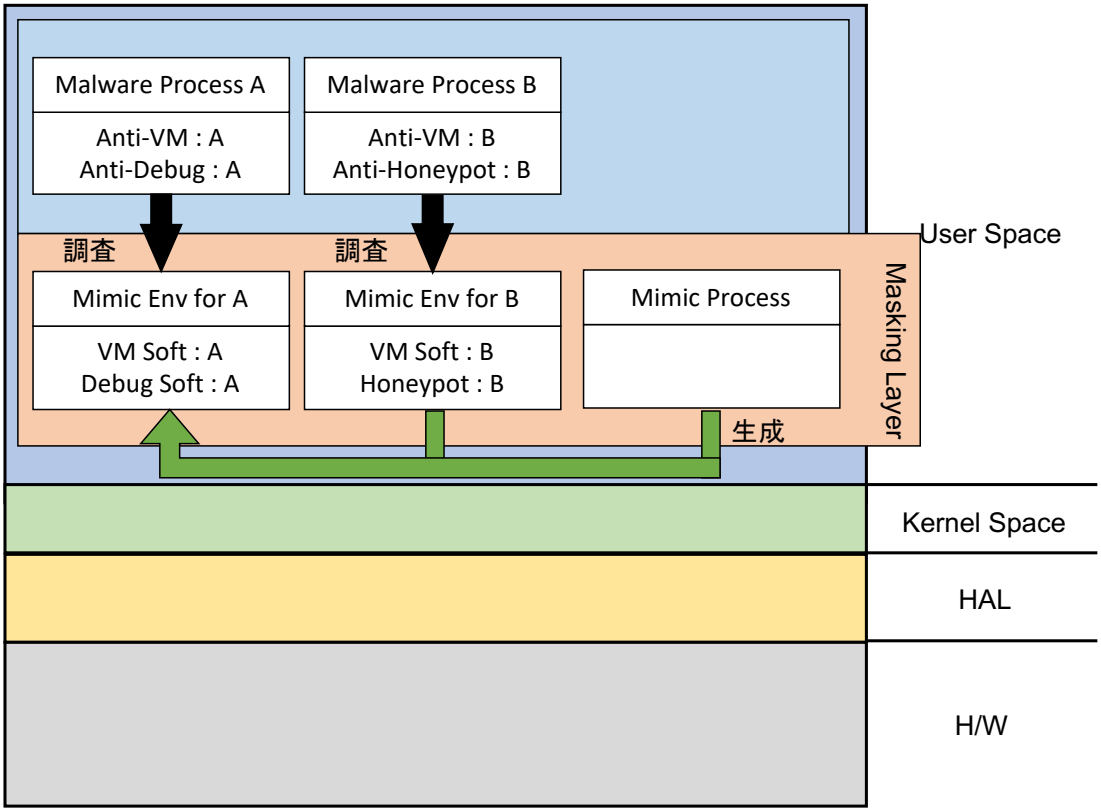

図 5.2: 仮想モデルによる擬態環境の構築

この状態に加え、マルウェアプロセスはハニーポット A を調査したとする。このとき、実環境にハニーポット B がインストールされている場合、両方のフィンガープリントを保持した環境になってしまう。実環境情報の隠蔽では、耐解析機能によって初めに調査された情報を優先し提供し続け、擬態の妨げとなるその他の情報を隠蔽する。これにより、整合性を保つ。

システムの動作への影響の観点から、耐解析機能以外の処理をフィルタリングしてはいけない。そのため、マルウェア以外の処理は、通常通り処理しなくてはいけない。

実装を行うフィンガープリントは、4.3 節で述べた分類表を使用する。

上記の機能を満たし、マルウェアプロセスへの擬態環境を提供するために、図 5.3 のような擬態構成を考えた。

Mimetic サーバはマスキングレイヤに位置し、各プロセスからの API を受け付ける。特定のプロセスが使用する API が耐解析機能である場合、または API の引数が耐解析技術の対象となる場合、そのプロセスに対して擬態環境の提供を開始

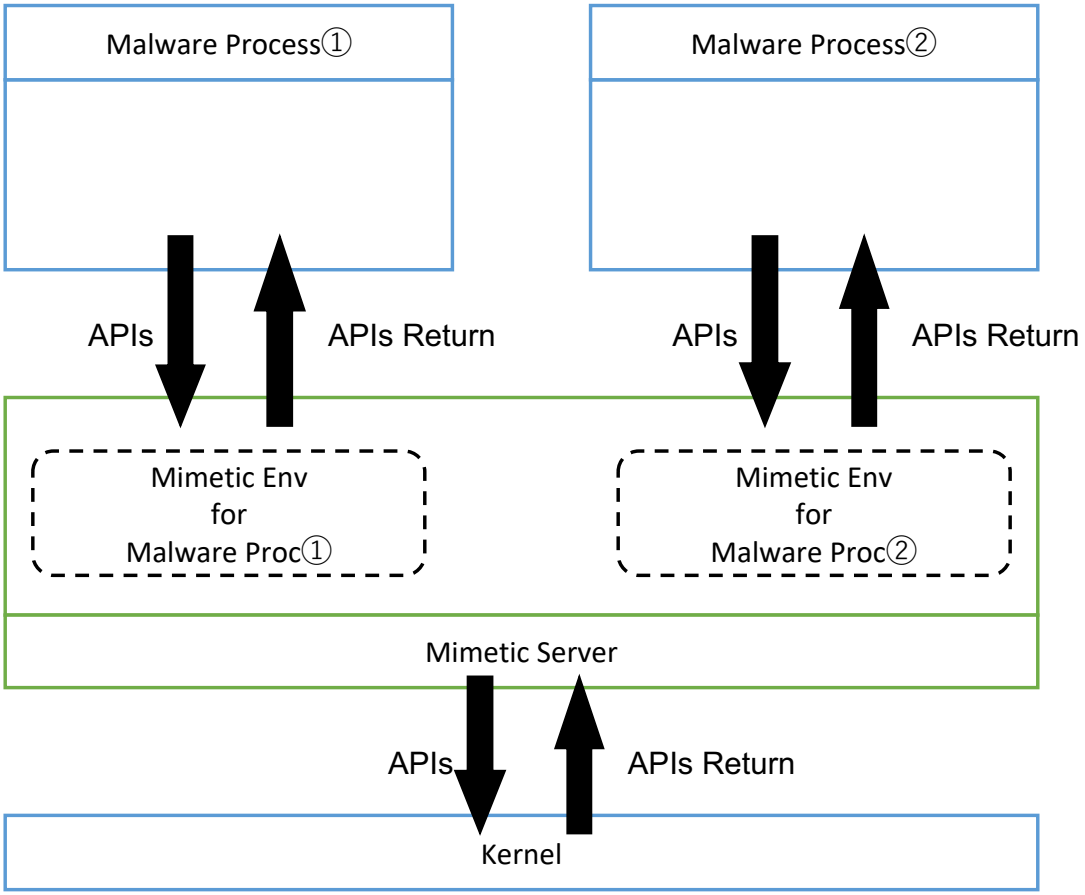

図 5.3: 警告型擬態によるマルウェア忌避環境の構成図

する。それ以外の場合、代理として Mimetic サーバがプロセスが使用する API の仲介を行う。

このようにして各マルウェアプロセスに対する擬態環境の提供を実現する。

5.4 仮想的な警告型擬態を用いたマルウェア忌避環境の実装

5.2 節で述べたように、マルウェアプロセスごとに警告型擬態を行う場合には様々な機能が必要となる。本節では、上記の機能を実装し、警告型擬態を用いたマルウェア忌避環境を提案する。

本研究で提案するマルウェア忌避環境は、図 5.3 の近似解として図 5.4 のような構成で実現する。

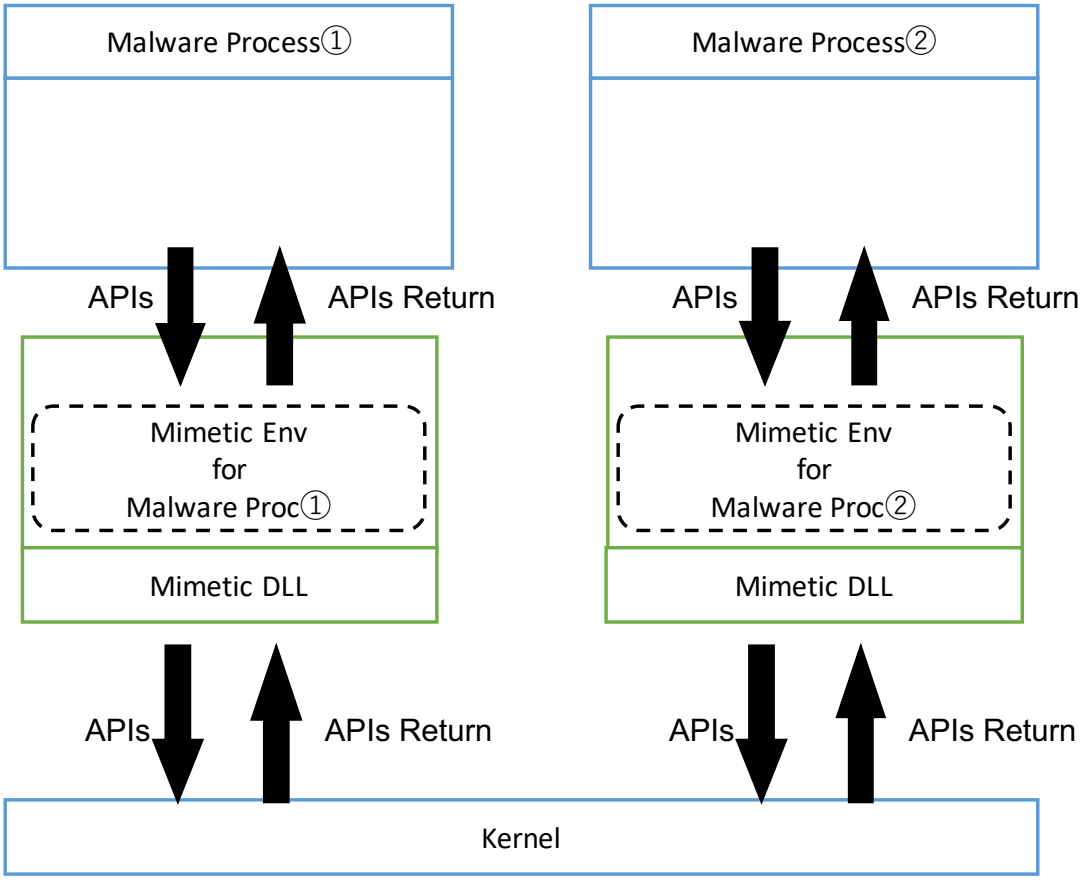

図 5.4: 警告型擬態によるマルウェア忌避環境の実装

本実装では、Mimetic サーバプロセスを動作させるのではなく、プロセスの耐解析機能を検出し、それに応じた擬態環境を提供する DLL を各マルウェアプロセスに呼び出させている。

本研究では、耐解析機能の使用を検出するためにラッパー DLL を使用する。このラッパー DLL では、実行された耐解析機能の情報をサーバプロセスに送信し、その情報を元に Mimetic サーバが警告型擬態情報を提供する。これにをトリガとして、Mimetic サーバが仮想的な警告型擬態環境の提供を開始する。トリガーの検出には、Detours [24] を用いたラッパー DLL を作成した。これを用いて API フックを行う。ラッパー DLL では、API 名とその引数を元に耐解析機能か否かを判断している。

本実装では、耐解析環境を検出した際にそれに合わせた擬態環境を構築している。Mimetic DLL を呼び出したマルウェアプロセスは、擬態環境構築状況をパラメータとして保持する。一度構築された擬態環境は、マルウェアプロセスが終了

するまでパラメータの変更を行わない。このようにして擬態環境の整合性を保ち、かつ擬態状況の記録を行う。

また、Mimetic DLL はマルウェアプロセス内に展開されるため、フィンガープリントとなるプロセスとして生成されない。よって潜伏機能は実現される。

本研究では、上記のような実装を行うことでマスキングレイヤを実装し、各マルウェアプロセスに対して警告型擬態の環境を提供した。

5.5 動作

本ツールでは、ラッパー DLL を用いて、マルウェアプロセス内で動作する擬態機構を作成した。本ツールの動作を確認するための一例として、Windows API である PathFileExists と PathIsDirectory を使用し、解析環境を検出する耐解析機能を保持した擬似マルウェアプログラムを作成した。

動作環境は以下の通りである。

- デバイス：Surface Pro 4
  CPU: Intel(R) Core(TM)i5-7300U 2.6GHz
  RAM: 8.00G
- OS：Windows10 （1909）

以下に、動作例を表示する。

プログラム実行時、擬似マルウェアプログラムは、本研究で作成した Mimetic DLL を読み込む。その上で、上記に記した API を用いて解析環境を検出する。

節で述べたように、本実装ではマルウェアプロセス内に擬態情報を保持する。そのため、フィガープリントとなるプロセスは生成されない。

API が呼び出された際、ラッパー DLL である Mimetic DLL が引数の調査を行う。その中に耐解析機能の対象となる環境に関する文字列が存在した場合、Mimetic DLL 内の擬態環境パラメータに擬態情報がセットされる。擬態環境パラメータを元に、耐解析機能に対する返答を変化させる。

例えば、擬似マルウェアプロセスが仮想環境 A を先に調査し、その後仮想環境 B を調査した場合は、仮想環境 A 以外は存在しないという返答を行う。たとえ、仮

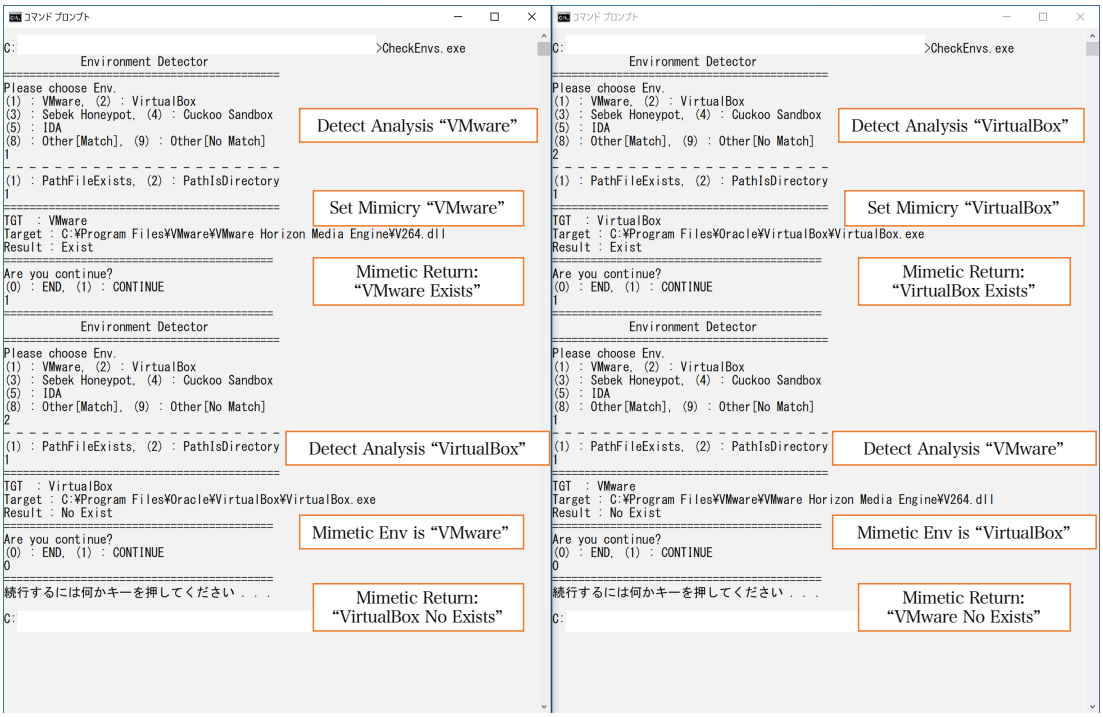

図 5.5: 複数の擬似マルウェアプロセスが動作した際の例

想環境 B がホスト上に存在する場合であっても同様である。これは、擬態の整合性を保つ目的から、先に調査された環境情報のみ返答を行うよう実装した。

次に、複数の擬似マルウェアプロセスが動作した場合を説明する。先に述べた通り、本提案ではマルウェアプロセス内に擬態情報を保持するため、各マルウェアプロセスに適した擬態を行う。

図 5.5 に示した通り、マルウェアプロセスはそれぞれ異なる擬態環境を見せられている。

第6章　まとめと考察

5章では、4章で示した擬態構築フローを用いて、マルウェア忌避技術への応用を行なった。

本章では、先行研究の実装手法と本研究で提案した設計手法、実装手法の定性評価を行う。

先行研究の特徴として、擬態環境のフィンガープリントを網羅的に保持している点と、フィンガープリントを実体のあるファイルや情報として保持している点が挙げられる。前者の利点は、様々な耐解析機能を持ったマルウェアに対応が可能なことである。耐解析機能を事前に検出することは難しいため、あらゆるフィンガープリントを持つことで対応の幅が広がる。その一方、環境の整合性が取れなくなる問題が生じ、新たなフィンガープリントとなる問題がある。

後者の利点は、マルウェアプロセスやOSの機能に対して変更を加えない点である。つまり、フィンガープリントとなる情報を実際に保持するため、API フックなどを行う必要がない。そのため、マルウェアの持つ耐解析機能の中で、プロセス実行時間やDLL インポートによる使用メモリサイズの超過に検出されることなく擬態が可能である。その一方、擬態対象を変更する場合にはそれらのファイルを破棄する必要性が生じる。また、フィンガープリントとなるファイルへのアクセスが一定期間無い場合、これがフィンガープリントとなる場合もあるため、定期的に更新を行わなければいけない。

本研究における提案の特徴は、各マルウェアに応じて仮想的な擬態環境を提供している点と、整合性を重視している点である。前者の利点は、各マルウェアプロセスに対し独立した擬態環境を提供することができる。これに加え整合性を重視することで、各マルウェアプロセスに最適な擬態環境の提供が可能である点である。これにより、先行研究と同等の耐解析機能への対応力を持つ。さらに、仮想的に擬態環境を提供するため、大量のフィンガープリントを保持しない。つまり、ファイルへのアクセス時間等のメンテナンスが不必要になる。その代償とし

表 6.1: 先行研究と本研究の提案に関する定性評価

	既存研究	提案手法 （設計）	提案手法 （プロトタイプ）
擬態環境の整合性	×	○	○
独立した擬態環境	×	○	○
擬態ログの収集	×	○	○
耐解析機能への 応答速度	○	△	△
フィンガープリント	• 網羅的に実装された 擬態要素	• Mimeticサーバプロセス • レジストリ • ファイル	• プロセスメモリ領域 • レジストリ • ファイル

て、API フックを行う DLL またはプロセスや実行時間、使用メモリサイズがフィンガープリントとなる可能性が残るため、これに対応しなくてはならない。

表 6.1 では、先行研究と本研究における提案（設計、実装）の定性評価を行った。

本研究の設計におけるフィンガープリントは 3 つ存在する。Mimetic サーバは、擬態環境の提供を行うプロセスであるため常駐する必要がある。また、プロセスとして動作するため、その時実行ファイルも存在する。そのため、必然的にプロセスと実行ファイルがフィンガープリントとなる。最後にレジストリには、プロセスの常駐をはかるために実行プログラムの登録をしなくてはいけない。以上の 3 点が、本研究の設計におけるフィンガープリントである。

本研究の実装では、本ツール固有のプロセスは存在しない。実装のフィンガープリントはマルウェアプロセスの中に生じる。本実装では、マルウェアプロセス内に DLL をロードさせるため、確保されるメモリ領域が変化する。また、ラッパー DLL により耐解析機能を検出、判断し、擬態環境を提供するため、処理時間にも影響が出る。

今回のようマルウェア忌避を目的とする擬態を行う場合には、各マルウェアプロセスに対応する必要があるため、単一マルウェアプロセスにしか対応ができない情報生成モデルの先行研究ではなく、仮想モデルによる独立した擬態環境を構築する本提案手法が適切であると考える。

第7章　おわりに

2章では、サイバー空間における欺瞞的防御について整理し、サイバー空間における擬態の定義を行なった。4章では、サイバー空間における擬態の分類と擬態構築フロー、擬態レベルを決定する分類表を提案した。5章では、4章で示した擬態構築フローを用いて、マルウェア忌避技術への応用を行なった。

これまで曖昧であったサイバー空間における擬態と擬態について調査し、定義した。また、サイバー空間における擬態の定義では、擬態カテゴリを4種定義し、擬態を行うために必要な要素と要件を定義し、擬態のための分類表を作成した。これを利用し、警告型擬態を用いた欺瞞的防御の一例として、マルウェア忌避技術への応用を行なった。

本章では本研究における展望についてまとめ、本論文の締めとする。

7.1　今後の課題と展望

本節では、擬態の分類と警告型擬態を用いたマルウェア忌避技術における課題、今後の展望について述べる。

7.1.1　擬態の分類における課題

本研究では、サイバー空間における擬態の分類と詳細化を行なった。

(1) 使用するOSに適した擬態分類表の作成

本研究では、PCの中で最もユーザ数の多いOSであるWindowsを対象として擬態分類表を行った。しかし、サーバとしてCentOSやLinuxが使用される場合や、MacOSが使用される場合もある。OSごとにファイルシステムの違いが存在する場合もある。よって、Windows以外のOSで擬態をおこなう場合、新たに分類表を作成しなくてはいけない。

(2) 擬態の限界について

自然界における擬態に限界があるように、サイバー空間における擬態にも限界が存在する。例えば、警告型擬態を行った環境において、プロセスの実行コマンドやその応答、コマンドによる変更までを確認した場合、それら全てを再現していない限りは擬態であることが判明してしまう。しかし、そこまで対応した擬態となれば、擬態対象の環境をインストールする方が明らかに容易である。このように、擬態の実装には非常に大きなコストがかかる可能性が生じるため、擬態構築者の目的に合わせた最適な擬態項目の提示が必要になってしまう。

7.1.2 警告型擬態を用いたマルウェア忌避技術における課題

(1) 本ツールにおけるフィンガープリントの隠蔽

警告型擬態の実装方法として、先行研究で 1 つ、本研究での提案で 2 つ提案した。6 章で言及した通り、本研究で提案した手法にも様々なフィンガープリントが存在する。本システムをより巧妙に擬態させるためには、Mimetic システムに対して隠蔽型の擬態を施さなくてはいけない。

(2) 様々な対解析機能への対応

本提案の一例では、Windows に存在する 2 種（PathFileExists と PathIsDirectory）の API にしか対応させることができなかった。つまり、本提案では擬態要素「ファイル」に対し、擬態条件「名前」と「パス」にしか対応していない。よって、上記以外の擬態要素と擬態条件に対応させる必要がある。

7.1.3 今後の展望

本節では、本研究における今後の展望を述べる。

(1) サイバー空間における擬態の可能性

本研究では、サイバー空間における擬態の分類と、擬態要素の分類表を提案した。擬態のカテゴリにおいて、保護型擬態の再定義と認識型擬態、隠伏型擬態の定義を行なった。本研究で提案した擬態要素の分類表を元に擬態レベルを概算することで、任意の擬態レベルを持った環境の作成が可能である。

(2) 警告型擬態によるサイバー攻撃者への牽制

本研究で確認した警告型擬態に関する先行研究は少ない。また、警告型擬態を積極的に取り入れることで、攻撃者に対してトレードオフの関係を与えることができる。サイバー攻撃者とセキュリティ従事者の間にあるいたちごっこの関係を有利に進めるために、サイバー空間における擬態と警告型擬態に関する分野をさらに開拓する必要がある。

(3) 仮想モデルによるセキュリティツールへの応用

本研究では、警告型擬態の実装方法として仮想モデルを用いた。既存のハニーポットやセキュリティツールは、情報生成モデルで構築されている。しかし、本研究で定義した仮想モデルは、マルウェアに実装された耐解析機能を仮想的に欺く手法であるため、攻撃観測ツールや動的解析ツールなどに応用することで、任意の環境を容易に再現することができるのではないかと考える。そのため、仮想モデルを用いて任意の環境を詳細に構築することで、認識型擬態と隠伏型擬態を行うセキュリティツールが作成できるのではないかと考える。

謝辞

本研究を進めるにあたり、多大なご助言と議論をさせていただいた関係者の方々にはこの場をかりて謝意を述べたいと思います。

主指導教員の篠田陽一教授を始め、副指導教員として知念賢一特任准教授、インターンシップ指導教員として丹康雄教授、本研究室の宇多仁助教には示唆に富んだアドバイスを頂きました。

WIDE Project の方々にも研究にかかるアドバイスをいただき、非常に有意義な議論を展開していただきました。

また、国立研究開発法人情報通信研究機構北陸 StarBED 技術センターの技術員の太田悟氏、本研究室の博士後期課程の三浦良介氏に感謝いたします。また先輩の淺葉祥吾氏、阿波史和氏、砂川真範氏、橋本光世氏、広瀬太志氏、三島航氏、宮崎駿氏、山口礼央氏に加え、同輩の菅野洋信氏、廣中颯氏、渡邊司揮氏また後輩の馬越紘氏、門脇真之佑氏、古寺雄馬氏、本間可楠氏、油布翔平氏、吉原昴司氏にはゼミ活動や輪講等において非常に有意義な議論をしていただいたことに感謝いたします。

最後にこれまでの学生生活や私生活を支えてくださった秀司氏、智恵子氏をはじめ、昂平氏、ちひろ氏、一枝氏に感謝いたします。

本研究に関する対外発表

- 北沢尭宏, *"Fake Honeypot* を用いた欺瞞的防御*," WIDE Project* ポスターセッション, Sep. 2019.
- 北沢尭宏, 篠田陽一, *"Fake Honeypot* を用いた欺瞞的防御*,"* マルチメディア、分散、協調とモバイル *(DICOMO2019)* シンポジウム, Jul. 2019. [25]

参考文献

[1] 国立研究開発法人情報通信研究機構 サイバーセキュリティ研究所サイバーセキュリティ研究室. Nicter 観測レポート 2018. http://www.nict.go.jp/cyber/report/NICTER_report_2018.pdf.

[2] NISC. 第2次情報セキュリティ基本計画「ＩＴ時代の力強い「個」と「社会」の確立に向けて」. https://www.nisc.go.jp/active/kihon/pdf/bpc02_ts.pdf.

[3] Joint Chiefs of Staff. Joint publication 3-13 information operations. https://www.jcs.mil/Portals/36/Documents/Doctrine/pubs/jp3_13.pdf.

[4] Mikhail J. Atallah Mohammed Almeshekah, Eugene H. Spafford. Improving security using deception. 11 2013.

[5] DFRWS. The digital forensic research conference dfrws 2001 usa. https://www.dfrws.org/sites/default/files/session-files/a_road_map_for_digital_forensic_research.pdf.

[6] 一般社団法人 JPCERT コーディネーションセンター. インシデントハンドリングマニュアル. https://www.jpcert.or.jp/csirt_material/files/manual_ver1.0_20151126.pdf.

[7] Lawrence Pingree. Emerging technology analysis : Deception techniques and technologies create security technology business opportunities. 2016.

[8] 角丸貴洋, 島成佳, 吉岡克成. 組織ネットワークにおける内部攻撃に対する模擬的欺瞞方式. コンピュータセキュリティシンポジウム 2014 論文集, Vol. 2014, No. 2, pp. 735–742, oct 2014.

[9] Frank Stech, Kristin Heckman, and Blake Strom. *Integrating Cyber–D&D into Adversary Modeling for Active Cyber Defense*, pp. 1–24. 07 2016.

[10] L. Shi, L. Jiang, D. Liu, and X. Han. Mimicry honeypot: A brief introduction. In *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1–4, Sep. 2012.

[11] 大山恵弘. マルウェアによる対仮想化処理の傾向についての分析. コンピュータセキュリティシンポジウム 2016 論文集, Vol. 2016, No. 2, pp. 534–541, oct 2016.

[12] 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 datasets 2016〜. No. 17, jul 2016.

[13] Cuckoo sandbox. `https://cuckoosandbox.org/`.

[14] Windows sandbox. `https://techcommunity.microsoft.com/t5/windows-kernel-internals/windows-sandbox/ba-p/301849`.

[15] 角丸貴洋, 島成佳, 渡部正文, 吉岡克成. 標的型攻撃対策に向けた欺瞞機構を用いた防御アーキテクチャ(技術と社会・倫理). 電子情報通信学会技術研究報告 = IEICE technical report : 信学技報, Vol. 114, No. 116, pp. 69–74, jul 2014.

[16] 杉生雅樹, 辻秀典, 橋本正樹. 欺瞞ネットワークの効率的な配置の評価. Technical Report 2, 情報セキュリティ大学院大学, 情報セキュリティ大学院大学, 情報セキュリティ大学院大学, dec 2018.

[17] N. C. Rowe, B. T. Duong, and E. John Custy. Fake honeypots: A defensive tactic for cyberspace. *2006 IEEE Information Assurance Workshop*, pp. 223–230, 2006.

[18] Deli Liu, Honglong Chen, Leyi Shi, Yuwen Cui, and Xu Han. Mimicry honeypot: an evolutionary decoy system. *International Journal of High Performance Computing and Networking*, Vol. 14, p. 157, 01 2019.

[19] Stopping malware with a fake virtual machine. https://www.mcafee.com/blogs/other-blogs/mcafee-labs/stopping-malware-fake-virtual-machine/.

[20] Jim Yuill, Michael Zappe, Don Denning, and Fred Feer. Honeyfiles: deceptive files for intrusion detection. *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, 2004.*, pp. 116–122, 2004.

[21] 松木隆宏, 新井悠, 寺田真敏, 土居範久. セキュリティ無効化攻撃を利用したマルウェアの検知と活動抑止手法の提案. 情報処理学会論文誌, Vol. 50, No. 9, pp. 2127–2136, sep 2009.

[22] 松木隆宏, 新井悠, 寺田真敏, 土居範久. マルウェアの耐解析機能を逆用した活動抑止手法の提案. 情報処理学会論文誌, Vol. 50, No. 9, pp. 2118–2126, sep 2009.

[23] 山田大. 欺瞞を用いた能動的サイバー攻撃防御手法の提案と実装. http://iss.iisec.ac.jp/sympo16/posters/M1_107poster16.pdf.

[24] Microsoft Inc. Microsoft - detours. https://github.com/microsoft/Detours/wiki.

[25] マルチメディア、分散、協調とモバイル dicomo2019 シンポジウム. =http://www.dicomo.org/2019/2019/.