# Pattern Matching Accelerator を用いた車載 IoT Edge 向け

# Malware Cyber-Security に関する研究

令和　2 年　1 月

日本大学大学院理工学研究科博士後期課程

情報科学専攻

柏　　山　　正　　守

# −目次−

# 第1章　総論

## 1.1　概要

クルマなどに搭載される IoT(Internet of Things)システムにおいて，潜在的脅威である Malware を自律的に Edge 部分で Real-Time Filtering する手段は，計算コストや消費電力の問題，対応する効果的なアルゴリズムが提案されていないなど多くの課題があり，現在まで，有効性の大きさに比較し，実用化へ向けた研究が推進されることは無かった。しかし，Connected Car という通信でつながるクルマの登場は，車両データの利活用や応用社会実装が進展し，Probe データによる交通管制や渋滞緩和，地図生成，ドライビングの利便性向上，クルマのプログラムアップデートによる不具合改善や高性能化など，様々な用途への展開が実現され，その重要性の高まりとともに Cyber-Security の脅威と戦わざるをえないという PC やモバイルデバイスと同じ宿命も生じている。このような状況の中で，人命にかかわる事故や都市機能障害など大きなインシデント(Incident)を未然に防ぐ手段を車載 IoT の Edge 部分においても組み入れていく必要性が生じている。

これらの状況への対応策として，Edge 部分にハードウェアセキュリティモジュール(HSM)を組み入れる研究開発が進んでいる。HSM は，ハードウェアとソフトウェアが協調した環境で，TEE(Trusted Execution Environment)や同等の耐セキュリティ機能を内蔵する。しかし，これら機能は，暗号鍵・認証による安全なプログラム実行領域の確保やデータ保護を目的としており，サイバー攻撃(Cyber-Attack)や異常の検出時に，セーフティな縮退運転を行うトリガーにはならない。そのため，Malware 侵入検知機能を追加装備することが必要である。尚，参考として，付録 A.1 に Connected Car が創造する価値や Society 5.0 との連携による System of Systems について解説する。さらに，TEE に関しては，付録 A.2 に構成概要を説明する。

現在，これら Malware 侵入の検知手段は，Cloud 側の AI(Artificial Intelligence)を用いた統計的解析処理による検出手法が一般的である。しかし，Edge 部分において，複雑な AI 処理等を用いた手法は，計算コストの問題からリアルタイム自律識別処理に限界があった。そこで本提案では，Malware を Texture Image として抽象化し，Pattern Matching Accelerator(以降 PM Accelerator と呼ぶ)を用いた多重テクスチャ(Texture)解析を行うことで，IoT の Edge 部分への適用が可能で低い計算コストを実現するシンプルな検出手法を創出した。

従来，パターンマッチングは非常に高負荷な処理と考えられてきたが半導体技術の進化により，汎用プロセッサに PM Accelerator を組み合わせたヘテロジニアスコンピューティング(Heterogeneous Computing)環境は，非常に低計算コストで Malware Texture Image の全スキャン実行が可能である。提案手法では，高次局所自己相関(HLAC: Higher-order Local Auto Correlation)から得られるマスクパターンを用いて Malware Texture Image の構造レベル解析を PM Accelerator で行い，パターンマッチングから得られる Malware Texture Image 固有のパワースペクトル特徴量(Power Spectrum Features)を主成分解析するアルゴリズムを用いて Malware 識別を実現した。提案手法の Malware 識別システムは，80%程度の識別性能を持つ。これらの性能評価は，6 種類の Malware ファミリー群，641 サンプルを用いてエミュレータシステムで確認した。さらに，既往開発結果から導出した計算コスト概算比較では，一つの Malware ファイルの識別処理は，数 100μsec オーダー程度にまで縮退させることが可能である。

提案手法の実証結果より，クルマなどに搭載される IoT システムが求める重要な要件が実現できる。具体的には，PM Accelerator の適用と新しい提案アルゴリズムの採用により，Malware 検出を目的とした低消費電力かつ高速処理の車載 Edge 組み込みシステムの実現が可能になる。

## 1.2 本論文の位置

近年，IoT を標的とした Cyber-Security のリスクが増大している。Cyber-Security は，Malware 感染，不正アクセス，DDoS 攻撃などからシステムを防御することであり，これらの脅威は年々増加傾向にある。具体的な数字で見ると，IoT 機器は，2017 年に全世界で 275 億個だったものが，2020 年には 1.5 倍の 403 億個へ

急激に伸びると予想されている。このような拡大の中で，2017 年，Cyber-Attack が急増し，IoT 機器を狙った Cyber-Attack は，700 億パケットに上り，2015 年比で 5.7 倍に達した[1][2]。IoT を標的とした Malware の急激な増加は，大きな脅威であり，IoT Edge システムにおいて，どのような手段で Malware を防ぐか，新しい技術開発が求められている。さらに，近い将来，クルマが Edge 端末の一部へ進化することから，クルマを狙った Malware の増加も懸念される状況にある[3]。特に NHTSA(National Highway Traffic Safety Administration：米国運輸省道路交通安全局)は，ICT 高度化するクルマの安全性リスク軽減に関して，Malware 侵入検知システム，ファームウェア更新の Cyber-Security，および V2X 通信インターフェースの Cyber-Security など，重要分野の Cyber-Security 研究に焦点を当てている[4]。

これらの状況を鑑みると，車載 IoT Edge における Malware Cyber-Security の防御技術の整備と進化は，喫緊の課題であるといえる。

図 1.1 は，ICT システムにおいて，Cyber-Attack からの防御手段として装備すべき基本的要素を列挙して整理したものである。例えば，Apple の最新 Edge 端末(2018/2H~)は，独自のハードウェアセキュリティモジュール(HSM)として T2-Security chip を装備し，ユーザのプライバシーに絡むクリティカルデータの暗号化や暗号鍵とその認証手段により，悪意のアクセスから保護する Cyber-Security 機構を具備している(Apple Secure-Enclave)[5]。さらに，不正アクセスなどの異常検知を記録するトレーサビリティ機能を備え，問題発生時はセキュアな限定的機能のみの縮退動作を行うことで Edge 端末と Cloud で構成するシステム(系)の堅牢性を担保している。ここで，重要な要素として，HSM があり，Cyber-Security 弾性を持ったデバイスとして最新トレンドになっている。

これら HSM を中心とした一連の暗号鍵生成・管理・認証機能の手段は，Cyber-Attack からクリティカルデータを保護するという思想に基づいており，極めて有効な防御手段である。しかし，既に組み込まれたファームウェアや前述の防御手段が有効化されるまでのシーケンス，通常稼働中のセキュリティチェックなど Malware 検知を行う手段を HSM へ入れ込むことが，より Cyber-Security の安全性を高める手段として必要であると考えている。一方，全ての ICT 機器に共通な課題として，製造工程や長期間の使用(ライフサイクル)における Cyber-Security 破綻がある。これらの課題に対しても Malware 検知手段を持つことは，

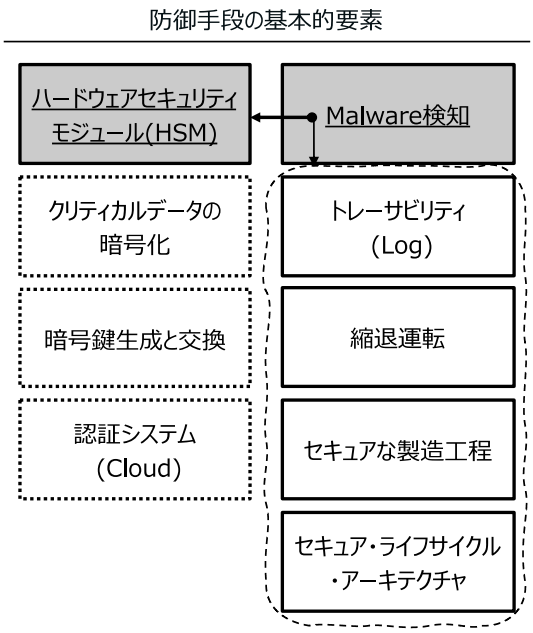



図 1.1 Cyber-Security における防御手段の基本的要素

全体の Cyber-Security 防御手段を有機的に機能させることが可能である。

本研究は，防御手段の中心で機能するハードウェアセキュリティモジュールに組み合わせる Malware 検出手段の研究を推進したものであり，特に Mobility の IoT Edge 側(クルマ)の Cyber-Security に焦点を当てた研究である。以下，最新のクルマにおける本研究が占める重要度と，そのシステム要件を説明する。

クルマは，自動運転制御や先進運転支援システム(ADAS)などの高度な ICT 機器を多数内蔵することから，組み込まれるプログラム量も膨大なステップになっており，図 1.2 に示すクルマと Cloud の接続構成(Connected Car)においては，OTA(Over-The-Air)による機能アップデートやディーラー作業による不具合修正なども頻繁に実行されるようになってくる。このような技術進化の中，ソフトウェア開発や工場生産現場におけるプログラム注入作業の過程において，Malware の混入や悪意のある故意の組み込み，それらが数年後に活動を開始するような潜伏型 Malware が忍んでいることなど，全体の製造工程，修理・メンテナンス，中古再利用などのライフサイクルの中で生じる Malware 混入防止のセキュリティ脆弱性を完全に担保することは難しい。

図 1.3 は，クルマの ICT システムの概要を示した図である。Navigation やインジケータなどの IVI-System 領域は，USB や SD Card などの接続が可能であり，Cyber-Security 上の攻撃可能面(Attack Surface)となる。さらに，TCU や OBD ポー

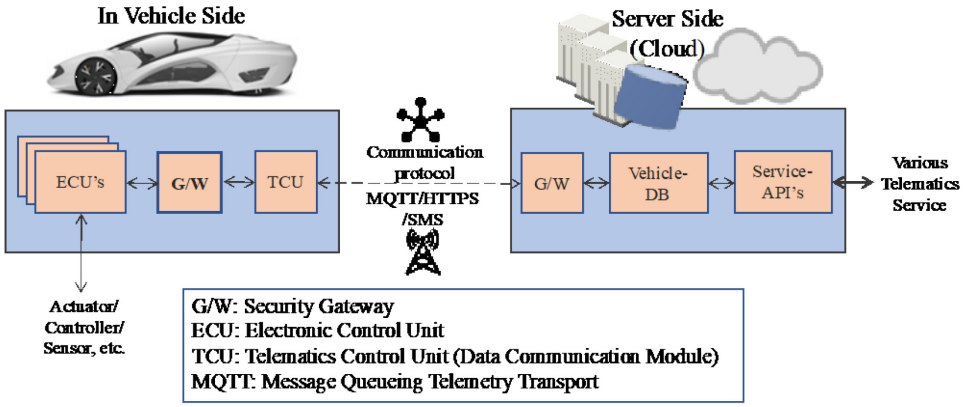



図 1.2 クルマと Cloud の接続構成

トを介した接続も Attack Surface になる可能性がある。これらの Attack Surface から侵入するハッキングなどを防御する目的で G/W(Security Gateway)を設け，クルマの走行安全に関わる ECU 群をシステム階層の奥へ配置し，厳重に守るような接続構成を採用している。OS の観点からは，IVI-System などは Linux 系が採用されているが，ECU は高速動作に優れたリアルタイム OS(RTOS)が用いられており，Malware 感染のリスクから考えると ECU 部位のリスクは低く，IVI-System 領域の可能性は高いと言える。しかし，今後は ADAS Camera などのデータ量増加に伴い車内ネットワークをシリアル通信プロトコルの CAN(Controller Area Network)から Ether Net へ，LAN 導入の動きやコスト面とデータ処理効率の観点から複数の ECU を統合した SOC(System on a Chip)化を行い，そこに TEE などのハードウェアセキュリティモジュール(HSM)を組み込む構想も進められており，システム進化の中で Linux 搭載が拡大して行く可能性も大きいと考えている。

ところで，近年の IVI-System は Navigation とクルマの状態を表示する計器類が LCD 一体となった構成も製品化されており，Malware 感染による異常な振る舞いは，危険な情報をドライバーに知らせる可能性もある。さらに，クルマとクルマ間，クルマと道路インフラ間などの V2X と呼ぶ情報交換機能も検討されておりクルマのオープン化は益々拡大していくことから図 1.3 に示した Security Gateway を設け Cyber-Security 性向上を担保することは重要な要件である。

Security Gateway の車載要件は，システムの立ち上がり段階からセキュア状態に保つセキュアブートやシステム稼働中の異常・攻撃検知などの機構が必要で

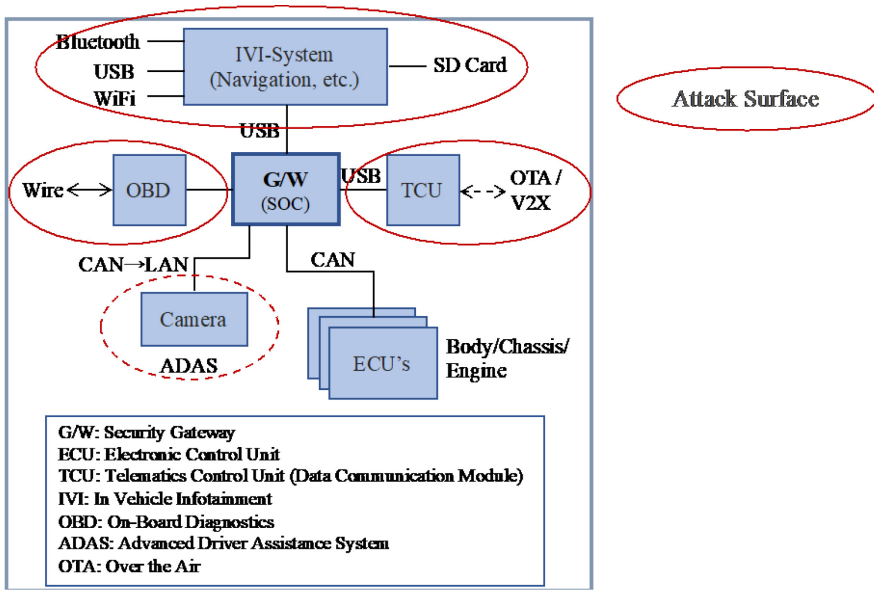



図 1.3 車載 Security Gateway

あり，セキュアブート時間は 100msec~500msec の高速動作が必要である。さらに，少しでも疑わしい攻撃や異常を検知したケースにおいては，クルマはセーフティを優先することから，検知性能に関しては部品故障予知と同等な 70%以上が妥当な性能だと自動車 OEM は考えている。

また，Cyber-Security のためには， Security Gateway のような Edge 部分が自律的に Malware を検出する手段，① Attack Surface からの侵入検知，②組み込み済みプログラムのスクリーニングパトロール処理，これら二つの要件を実現させることが重要である。さらに，如何にして Security Gateway で Malware を検出するか，リアルタイムの検出時間，検出率，ハードウェアリソース(コンピュータの消費電力やコンパクトなシステムサイズ）の Computing コスト課題を解決することも重要であると考える。

Security Gateway を構成するコンピュータシステムのアーキテクチャに関しても，前述の Computing コスト課題を踏まえた有効な方法も考える必要がある。

昨今，図 1.4 の Computer System の進化トレンドが示すように，Edge 部分のシステム(AI Edge)では，ヘテロジェニアスドメインスペシフィックコンピュータ(Heterogeneous Domain Specific Computer)と呼ばれる AI などの特定用途の処理を高速に行う複数種のアクセラレータと汎用処理を行うプロセッサを組み合わせ

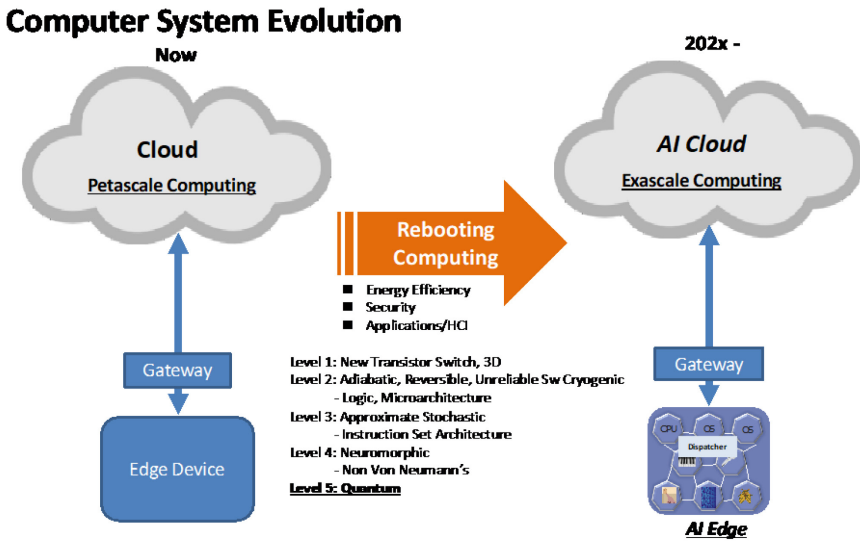



図 1.4 Computer System の進化[6][7]

た並列処理方式が注目されている[6][7]。背景の一つにムーアの法則の終焉(図 1.5)があり，Computing の階層構造の最下層を支えるシリコン半導体の高性能化が難しくなったことから，単体プロセッサで処理できるデータ量に限界が見えたことが大きな要因である[8]。

さらに，どのようなデータ処理やアルゴリズム処理が，何処の機能レイヤーに割り当てて処理することが効率的か，計算処理システム全体の調和を設計することが重要になる(図 1.6)。どのようなデータを Cloud 側で処理すべきか，Edge 側で処理すべきデータは何か，Edge と Cloud 協調すると価値があるデータは何

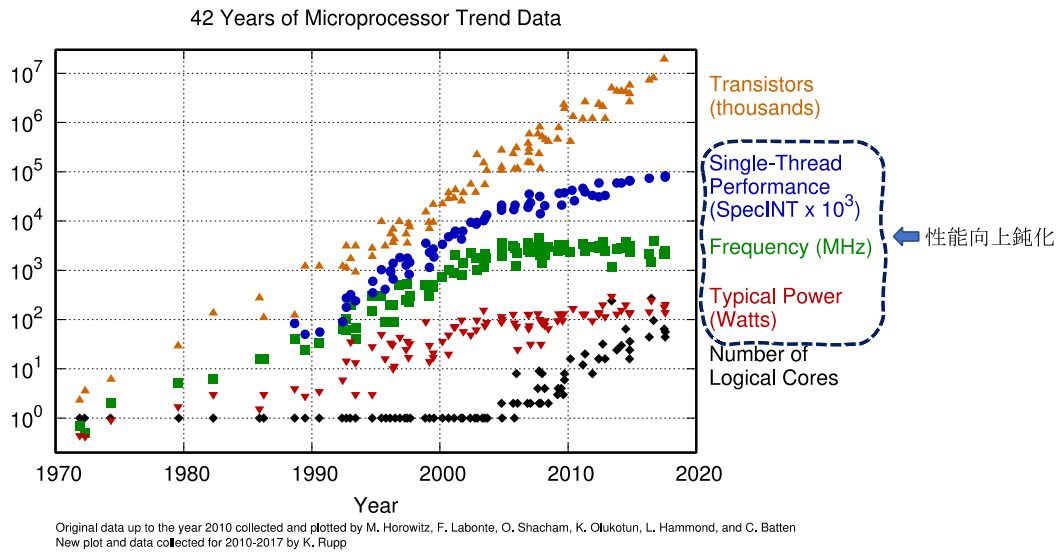



図 1.5 Computer 性能の鈍化[22][23]

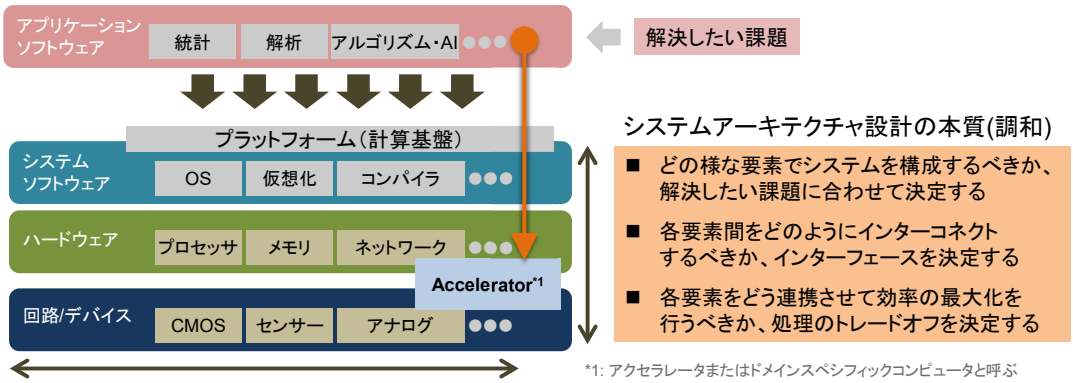



図 1.6 Computing 階層構造と System 設計思想

かなど，解決しようとするアプリケーションに合わせて考えて行くことが必要になる。同様に，システムの組み立て方に関して文献[9]では，システムのボトルネックであるデータパス性能に関して，基本的概念が解説されており，現在でも同じレジュームに従っていると考える。

本研究では，このようなシステムの本質的課題解決の手法から得られるシステム設計思想に従い，車載 Security Gateway の Malware 検出システム構成においては，汎用プロセッサに全ての処理を担わせるので無く，専用アクセラレータを組み合せた SOC を実現し，計算コストの縮退を実現させることが最も有望であると考えた。

以上，本論文の位置は，車載 IoT Edge が抱える深刻な Malware Cyber-Security 問題の解決手法に関して研究したものであり，具体的には，クルマの車載デバイスの進展に起因するセキュリティ課題の対策手段，および車載時に課題となる IoT Edge Computing の計算コスト削減技術に関連した研究である。

## 1.3 本論文の目的と特徴

本論文は，IoT の Edge 部分において，自律的に Malware 検出を行うアルゴリズムの有効性検証を行い，その方式提案と車載実装について考察を行うものである。

本研究の成果を使うことにより，IoT Edge Computing システムの Cyber-Security を高めることが可能となる。

本研究は，過去より多数の実績を蓄積し，高い識別性能を示している Malware の動的コード解析や静的コード解析とは路線を異にし, Edge デバイスに Malware 検出機能を組み込むのであれば，どのような課題を克服すべきか，という観点で取り組んだ研究である。

既往の代表的解析研究の特徴は以下の通りである。

- 動的コード解析は，隔離した専用動作環境を構築し，Malware の振る舞いを観測することで種別を特定する手法である。しかし，隔離環境構築は，VM 上に構築するため，CPU パワーと OS 環境に高い要件を必要とする[1]。

- 静的コード解析は，Malware バイナリーコードの逆アセンブラから実行可能ファイルの制御フローを解析し，共通命令コード(N-gram の組)の頻出度計算により類似度判定する手法である。しかし，この手法もサーバクラスの Computing リソースが必要である[1]。

- これら代表的解析手法は，いずれも 75%以上の高い識別精度を誇るが，Computing 環境のコストが高いことから Edge への適用は難しい。即ち，IoT Edge Computing へ実装する場合，サーバと Edge とのデータ交換オーバーヘッドが大きく，リアルタイム性能が劣る。特に，リアルタイムに解析が要求される場合は，通信やネットワークにかかるレイテンシー，データバンド幅, トランザクションデータ量を考慮すると, Edge の近くに Computing を設けることがセオリーである。

以上の理由より，重要なファクターは，Edge Computing 部分の計算コストを如何に削減させるのか，Malware 検出には，どのような機能やアルゴリズムが有用なのか，を探求する必要性が求められる。

一方，本論文の提案アルゴリズムは，Edge Computing 部のコスト削減に PM Accelerator を用い，そのマスクパターンに高次局所自己相関(HLAC)を応用することで，Texture Image という対象物を多段に抽象化し，局所的特徴と大域的特

徴を抽出することで，そのものが Primitive に持つ特徴を数値として解析するという新規性のあるロジックを実証したものである。

本研究の特徴は，コンピュータハードウェア&アーキテクチャと Texture Image の解析，ベクトルの主成分解析，およびクルマの ICT 技術などの複合分野の融合により，Malware Cyber-Security の効果的手段・方式を提案したものである。

これらの研究開発の着眼ポイントは，以下の通りである。

(1) Malware のコードが悪意を持つというのは，どのような動作をコンピュータやプロセッサに行うから生じる問題なのか。
(2) 機械語命令列は，そもそも何か，何を見つけることが必要か。
(3) 数学や物理学の理論では，何かを見つけるとき，直接そのものを証明するよりも，抽象化したものを観測することで証明できる場合がある。例えば，フェルマーの定理の証明では，2 段階の抽象化を実行している。
(4) パターンマッチングの本質は何か，マスクパターンの役割とは何か，Image の Primitive とは何か, パワースペクトル/時系列データ(Power Spectrum/Time Series Data)とは何か。
(5) Computing の視点では，Edge と Cloud(サーバ側)の役割，およびアクセラレータ，システムバランスなど。

昨今，機械学習を用いた解析は，極めて進化しており，ベクトル演算を加速できる高性能な GPU などを多用した研究が盛んである。しかし，ムーアの法則が限界を迎え，消費電力やサイズなどの課題から，将来の Computing 環境は，大きな変革点にあるといえる。

本論文では，機械学習や高い Computing Power に依存しない新しいアプローチにより Primitive でシンプルなアルゴリズムを導出し，実証した。

## 1.4 本論文の構成

本論文は 6 章から成っており，その構成を以下に簡単に述べる。

まず第 2 章において，関係する既往研究手法と導き出される本論の着想に関して述べる。本研究は，重要な既往研究の提案手法を組み合わせ応用したことと，これら既往研究から得られた発想から新たなアルゴリズムを導き出し，全体を融合することで新しい価値を生み出し課題解決手法を構成したものである。既往研究では，Malware を Texture Image として解析する発想と低計算コストな Edge Computing の応用が重要であり，それらから導出される着想に関して説明する。

第 3 章では，提案手法とアルゴリズムに関して，Malware の特性を考慮した構造的解析手法と，その数値化に用いる高次局所自己相関特徴(HLAC)によるパターンマッチング処理，およびパワースペクトル/時系列データへの数値化に関するアルゴリズムを説明する。

第 4 章では，提案手法を用いた Malware の識別実験に関して，評価サンプルの準備と評価プロセスの説明を行う。さらに，高効率な実験環境構築の概要説明，Malware 識別実験データの解析手法の導出と，その検証結果に関しても説明する。

第 5 章では，提案手法の有効性と課題に関して，未検知や誤検知サンプルおよび難読化に関する定量的推察を与え，識別性能に大きく影響すると考えられているコンピュータアーキテクチャの問題，さらに，車載実装における Edge 適合性能に関して考察する。

最後に第 6 章において，以上の結果をまとめ，結言と今後に残された課題について述べる。

# 第2章　Pattern Matching Accelerator による Malware の構造的解析処理

本研究は，重要な既往研究の提案手法を組み合わせ応用したことと，これら既往研究から得られた発想から新たなアルゴリズムを導き出し，全体を融合することで新しい価値を生み出し課題解決手法を構成したものである。

本論の提案手法に組み合わせた既往研究は，以下の通りである。

(1) Malware と Malware 亜種を含む Malware ファミリーの概念と構成する機械語命令列の相関に関する研究提案からの発想。

(2) Malware ファイルのバイナリーデータ(Binary Data)を画像化することによる分類と識別手法の研究から，Texture Image 解析と Malware 構造の関わりに関する発想。

(3) Malware Image ファイルをパターンマッチングさせるためのマスクパターン創出に応用する Texture Image 解析の研究。

(4) 低計算コストな Edge Computing を実現するための Pattern Matching Accelerator の研究と，そのアクセラレータが持つ性能結果からの概算への適用。

以下，本章では，既往研究手法と導き出される本論の着想に関して述べる。

## 2.1　Malware 機械語命令列の相関

Malware は，ソースコードが公開されたり，コード解析などにより Malware 原種と類似した亜種と呼ばれるファミリー群が多数拡散している。

Malware 検出において，該当する Malware が，これらファミリー群のどの種類に属するか分類することは，その後のトレーサビリティ(Traceability)における効率的な対策に重要であるとされている。さらに，これらの改変された亜種では，難読化などが施され耐解析機能を備える Malware も出現している。文献[10][11]で Iwamura らは, プログラムコードの機械語命令列の類似性に基づいて Malware を種類毎に自動分類する手法を提案している。分類を行う際に重要となるのが，ある Malware と別の Malware の類似度をいかにして計算するかであり，Malware を構成する機械語命令列に着目し，ある 2 つの Malware の機械語命令列を比較しすることで，共通命令列の出現頻度と類似度の相関を求めている。具体的には，機械語を逆アセンブルして得られた逆アセンブルコードから命令を抽出し，N-gram/N-perm のファイル解析により共通な命令列頻度から類似度を求め，機械学習手法を用いて捕獲した Malware の種類を 75%の精度で特定している。

これらのことより，課題解決につなげるための重要なポイントは，Malware ファイルの機械語命令列における共通命令列の出現頻度は，類似度を求める上で重要なファクターであることが理解できる。しかし，当該提案方式は，厳密なプログラムソースコード解析を行うことから大きな計算コストを伴い，主にサーバ側(Cloud)で効果を発揮する識別手法であると考える。

IoT/Edge システムでは, サーバ側と Edge 側が協調し, Edge 側単独でも Malware 検出を可能とし，相互補完により防御効率を上げる試みも必要だと考える。
Malware 検出において，該当する Malware が，これらファミリー群のどの種類に属するか分類することは，その後のトレーサビリティ(Traceability)における効率的な対策に重要であるとされている。さらに，これらの改変された亜種では，難読化などが施され耐解析機能を備える Malware も出現している。

## 2.2 Malware の Texture Image 化による識別

他方，厳密なプログラムソースコード解析と全く異なる手法も提案されている。提案では，Malware ファイルの画像解析から特徴量を求めて識別を行なっている。

文献[12][13]で Nataraj らは，Malware バイナリーファイルをグレースケール画像化し，Texture Image 解析を応用する手法から高い識別率を得ている。文献[12][13]の提案手法は, Malware バイナリーファイルを 1byte(8bit)単位でグレースケールへ変換し，メッシュ濃淡が 256 階調のイメージ(図 2.1)を作成する。これらのイメージに対して，画像全体から求まる大域的画像特徴量(特徴ベクトル)の導出を Olivia らが提唱し文献[14]に示した GIST 記述子(GIST Descriptor) [14]を応用した Texture Image 解析を行っている。

ところで，GIST 記述子は，画像のシーンのような大局的な特徴を求める画像解析手法として提案されたものである。具体的には，画像にガボールフィルタ(Gabor Filter)を適用したモデル化であり，ガウシアンエンベロープ(Gaussian Envelope)をフィルタとして局所の方向と空間周波数を抽出する特徴量である。

Malware バイナリーファイルをグレースケールへ変換し作成された画像(図 2.1)は，一般的な Texture Image(図 2.3)に似ていることから，Texture Image の解析手法を適用することで分類を試みている。Texture Image 解析を応用する理由は，大多数の Malware ファミリーが同じファミリーに属する画像のレイアウトや Texture が非常に類似しており(図 2.2 は，同一ファミリー:Mirai の 3 個のサンプル画像)，パッカー等により難読化された Malware を含んだファイルの場合でも高い識別精度で分類できることからである[12][13]。

文献[15]で Suzuki らは，GIST 記述子を活用した Malware 画像解析手法により 80%程度の識別性能が得られたことを報告している。しかし，GIST 記述子を用いた手法は，80%程度の高い識別精度を誇るが，フーリエ変換など計算コストが大きい処理を多用することからサーバ側(Cloud)での処理に向いていると考えた。

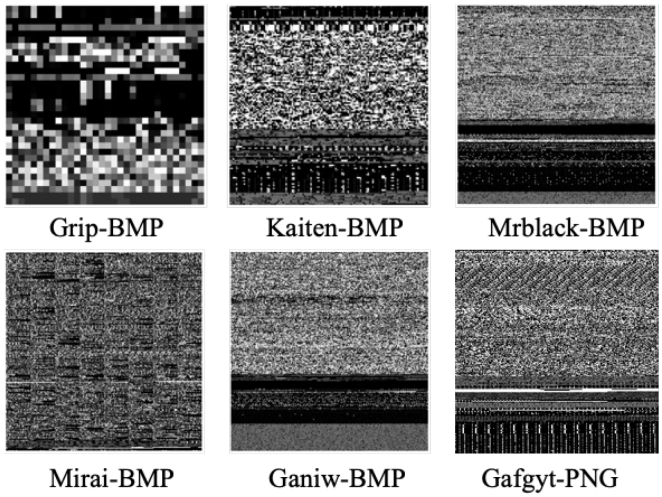



図 2.1 Malware ファミリサンプルの Texture Image (Malware Name: Grip, Kaiten, Mrblack, Mirai, Ganiw, Gafgyt)

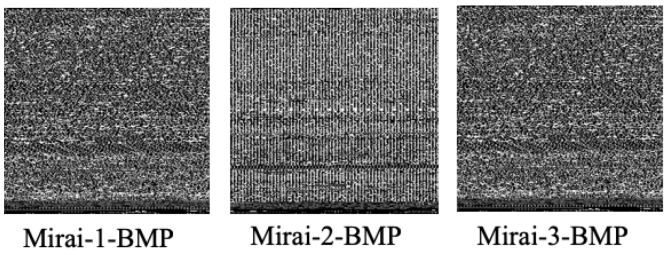



図 2.2 Malware Mirai ファミリの Texture Image (Identical Family: Mirai)

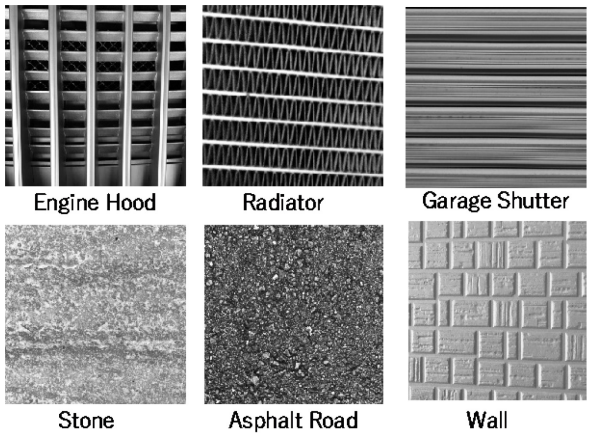



図 2.3 一般的な Texture Image

## 2.3　Texture Image 解析と統計量関係

Texture Image 解析は，構造的な配列規則からパターンマッチングにより要素群の特徴ベクトルを分類していくことであり，シンプルにとらえるとパターンマッチングにより対象が持つ普遍的な数学量の特徴を抽出し分類する技術であると考える。そこで，本研究では Malware の Texture Image 解析をパターンマッチングで行うことで計算コストを大幅に縮退させることが可能であると考えた。以下，本提案の発想につながる手法に関して述べる。

文献[16]で Tomita らは，要素がある種の規則性に従って配列されてできる繰り返しパターンをテクスチャと定義しており，テクスチャの性格を調べることは Texture Image 解析に重要な役割を果たすとある。Texture Image の解析は，大きく構造レベルの解析①(Structural Analysis①)と統計レベルの解析②(Statistical Analysis②)があり，①は，各要素の大きさや形，要素の配列密度などからテクスチャを分類する。さらに，②は，1 次統計量解析と 2 次統計量解析があり，1 次統計量では，繰り返しパターンの局所的性質の統計量を調べるアプローチだと示されており，イメージ内の着目する部分における明度のヒストグラム(Histogram)が基本になると示されている[16]。2 次統計量では，イメージ内の要素の位置関係の組み合わせが重要なファクターになる。これら 2 次統計量解析においては，パワースペクトル(Power Spectrum)や自己回帰モデル(Auto Regressive Model)などの自己相関関数(Auto-Correlation)が重要なファクターとして扱われている。さらに，興味深い関係性として，自己相関とパワースペクトル及び自己回帰モデルが相互ループの関係を形成しており，共通の長所・短所を持つと解説している[16]。

このことより，自己相関関数と自己回帰モデルの関係から抽出されたデータは，Texture Image が持つパワースペクトルと等価な特徴量を導き出すことが可能になり，局所的な明度とも相関を持つと考えた。

## 2.4　Texture Image の構造解析とマスクパターン

Texture Image の解析をパターンマッチングで行おうとする場合，重要なポイントは，何を見つけるためにどのようなマスクパターンを用意するかということである。探すべきオブジェクトが明確な場合は，そのオブジェクトをマスクパターンとして照合すれば良いが，不明確な場合は，適切なマスクパターンを創出する必要がある。

文献[17][18]で Toyoda らは，Texture Image 識別のために高次局所自己相関特徴(HLAC Features)を抽出するマスクパターン群の実験報告がされており，前述の GIST 記述子の特徴抽出手法よりも高い識別性能が確認されている。これらの実験結果が示す有力な仮説は，高次局所自己相関特徴抽出に用いるマスクパターン群には，テクスチャ解析に鍵となる要素が対象とする Texture Image から構造的に得られる特徴を定式化するポテンシャルがあると考えた。そこで，高次局所自己相関の理論から，本研究に適用できると考えた理論式部分を取り出した。

理論の根底にある式は，式(2.1)に示した自己相関関数であり，Texture Image 領域内における位置不変性と加法性が与えられる。この関数を次数 M に拡張した式が式(2.2)になり，これは対象とする Texture Image 領域全体を相対変位$(a_1, \dots, a_M)$に対応するマスクパターンでパターンマッチング・スキャンした結果であると結論付けされる。従って，高次局所自己相関特徴は，Texture Image 領域内に存在する局所マスクパターンのヒストグラムを計算していることに相当していると考える。Texture Image 領域全体を Malware のプログラムファイルと考えると，局所マスクパターンのヒストグラムから，機械語命令列群の出現頻度を求めることが可能になる。さらに，Texture Image 領域全体を次数 M のマスクパターン種でスキャンした結果は，Malware 原種と亜種を含めたテクスチャの画像特質を表現した類似関数になることが理解できる。例えば，HLAC 3×3 サイズのマスクパターンは画像の局所特徴を抽出し，それは画像が持つ高周波成分である。

$$v(r) = \int f(t)\, f(t+r) dt \quad \cdots (2.1)$$

$$v_M(a_1, \dots, a_M) = \int_{\mathcal{R}} I(r) I(r+a_1) \dots I(r+a_M) dr \quad \cdots (2.2)$$

これは，フーリエ変換基底(Fourier Transform)で比較すると，周波数成分はマスクサイズに依存し，方向成分はパターン形状に依存している。方向だけでなく様々な２次元分布を持つパターンを用いることから局所領域の詳細な解析が行える。ところで，次数 M を大きくすれば，詳細な情報量が取得できパターンマッチングの識別に有利に働く。しかし，組み合せ数が指数的に増大し，膨大な計算コストを要する。

本研究の提案手法では，指数的に増大した処理に対して Pattern Matching Accelerator を用いることで，計算コストを縮退させることが可能になる。

## 2.5 Pattern Matching Accelerator 導入による低計算コスト化

Security Gateway を通過する様々なデータファイルに対して，Pattern Matching Accelerator(PM Accelerator)用いて，Malware ファイルのスクリーニング処理をリアルタイム実行する。

PM Accelerator は，ロード/ストア(Load/Store)の回数を縮退させ，負荷の重いパターンマッチング処理を軽減することで，全体の処理効率を改善する。このことにより，リアルタイム処理スピード向上と低消費電力を達成可能となる。ここで，ロード/ストアにかかるコストの縮退重要性は，文献[9]に示した通りである。

従来，パターンマッチングは非常に高負荷な処理と考えられてきたが，汎用プロセッサに PM Accelerator を組み合わせた Heterogeneous Computing 環境は，非常に低計算コストで Malware Texture Image の全スキャン実行が可能である。PM Accelerator は，半導体デバイスの集積度向上に伴い，処理速度と消費電力に優れた照合規模の大きなチップが複数提案されている。図 2.4 は，PM Accelerator の概要を示した図である。元イメージデータと照合するマスクパターンデータを入力し，オペレーションコマンドを発行すると，ワンオペレーションサークルでヒットしたロケーションとヒット回数がマッチング結果データとして出力される機構になっている。内部構造は，連想メモリ CAM とプライオリティエンコー

ダ(Priority Encoder)を組み合わせた構成になっており，各 PE(Processing Element)で Pixel 毎の論理演算を並列実行できる演算器を組み入れた一種の In-Memory 演算プロセッサとなっている。

文献[19]~[21]で Inoue, Pham らは，これらの機構を持ち特質を満たした PM Accelerator を提案している。文献[20][21]では，これら提案技術の性能評価結果と ASIC(Application Specific Integrated Circuit)適用時の性能目標が示されている。提案技術を FPGA(Field Programmable Gate Array)上に実装したケースにおいて，画像サイズ 128×128 Pixels の Matching Time は，約~0.122msec@50MHz オペレーションサイクルとのことである[21]。さらに，ASIC 適用時の性能概算は，画像サイズ 320×240 Pixels の Matching Time は，~5μsec@100MHz オペレーションサイクル，当該処理に要する電力は，6μJ(電源電圧:1.2V, 演算時電流:1.2A, オペレーションサイクル: 100MHz)と見積もっている[20]。これらは，Intel 社の E7600 CPU に比較し，処理性能比で 16 万倍高速との結果が示されている[20]。

ところで，このような In-Memory 演算プロセッサは，従来の半導体デバイスの集積度では実現が難しかった。しかし，メモリとロジックを融合させたデバイス技術の進歩により，様々な用途に向けた In-Memory 演算プロセッサが提案されつつある。特に Deep Learning などの特定の処理を高速実行する AI プロセッサは，今後 Edge Computing 領域へ多用されていくと予測する。

## 2.6　2 章のまとめ

本章では，本論の理論形成に強く影響を与えた発想に関して，その理解と応用価値を説明した。理論形成に重要な既往研究は，Texture Image 解析の定量的解析から導き出される Primitive の本質であり，本論の新しいアルゴリズムである機械学習に依存しないシンプルな解析手法のアイデアを発想させる重要な理論である。

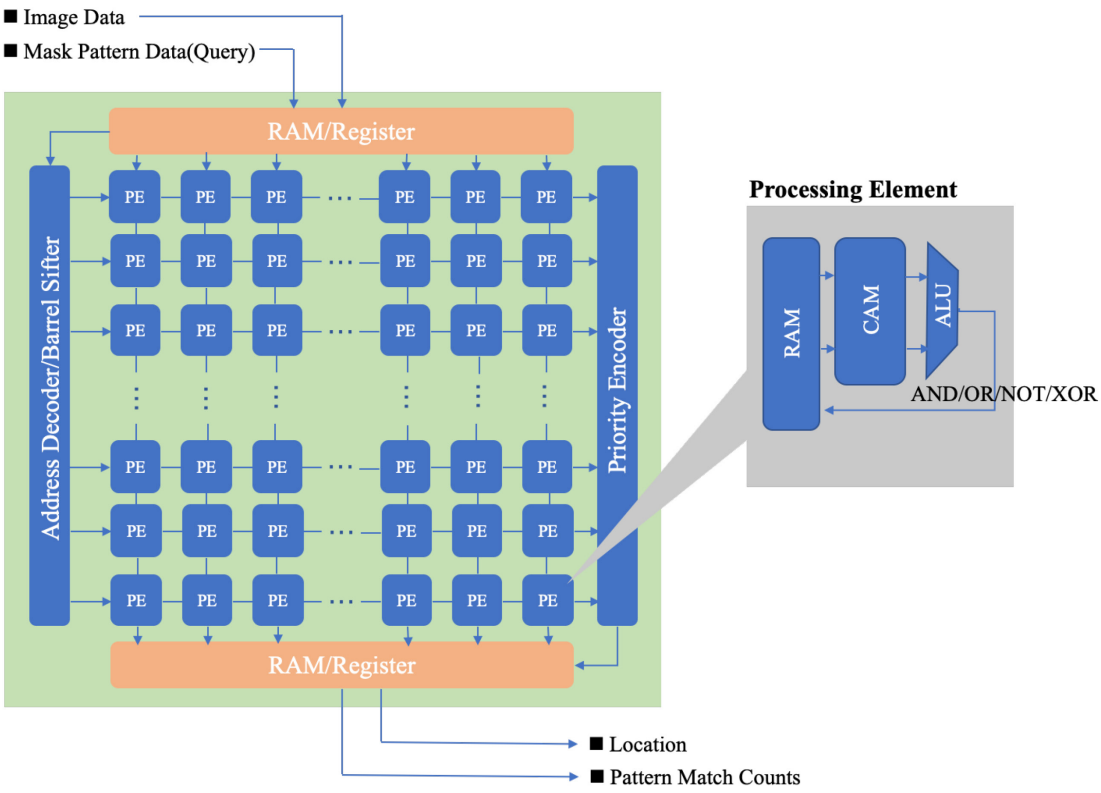



図 2.4 PM Accelerator の概要

# 第3章　提案手法とアルゴリズム

本章では，提案手法と，その Malware 検出技術の新しいアルゴリズムに関して論ずる。

提案手法は，高次局所自己相関(HLAC)から得られるマスクパターンを用いて Malware Texture Image の構造レベル解析を PM Accelerator で行い，パターンマッチングから得られる Malware Texture Image 固有のパワースペクトル特徴量を主成分解析するアルゴリズムを用いて Malware 識別を実現した。

## 3.1　Malware 特性を考慮した構造的解析

Edge 部分において，リアルタイムに近い識別処理を行うためには，プログラムコードの分割や実行を行いながら解析処理を進めるプロセスは現実的では無い。さらに，暗号化や圧縮などの難読化技術への対応策も必要である。

そこで，本研究では Edge 部分の Malware 検出には，リアルタイムで Malware ファイルの画像化を行い Texture Image として，パターンマッチングやパワースペクトル解析を組み合わせる手法を選択した。また，Malware ファイルの機械語命令列は，自然な Texture Image と比較し，明確な配列の規則性を持つことから，構造レベルの解析手法を用いて 1 次の特徴量を抽出することが有効であると考えた。

各々の Malware ファミリーは，原種とコード改変を受けた亜種により構成されており，改変された亜種プログラムコードは，原種のプログラムコードの特質を引き継いでいる。本理論では，Malware コードのコアとなる機械語命令列は，プログラムコードの何処かに必ず記述されており，難読化の順序入れ替えや，データ挿入などのテクニックを用いても，プログラムコード中の何処かのアドレ

ス位置に存在すると推定した。従って，Malware の画像化後の配列規則は，プログラムコードの機械語命令列群が，ある種の目的を持った処理を行うための固有コードとして局所的に類似な Texture Image を形成していると考えた。このことは即ち，局所的なパターンとして Texture Image の中に存在すると言うことであり，逆アセンブラーを用いて，この機械語命令列が何を意味しているのか解析することと，このセクションのパターンは何かという特徴を見つける方法と等価である。さらに，ファミリー毎に原型となる固有パターンのレイアウト (位置)とテクスチャ(パターン)を比較すれば，ファミリー毎の分類が可能であると仮定した。

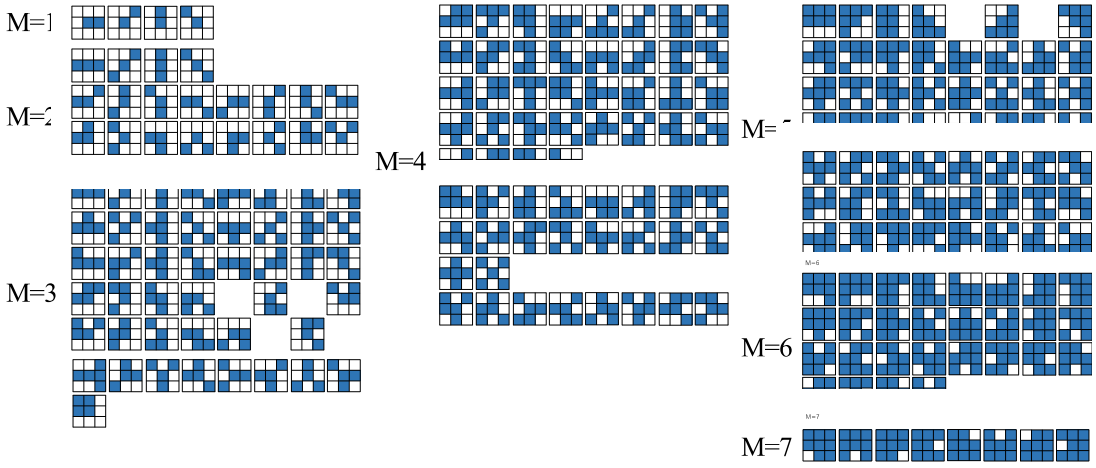
## 3.2　高次局所自己相関特徴によるマスクパターン創出

Malware 原種のプログラムコードから最も特徴を表す固有コードを見つけ，それをマスクパターンとしてパターンマッチングさせることも考えられるが，どの機械語命令列が最適なマスクパターンとなるコードか特定することは困難である。さらに，大きな課題として，正常ファイルとの識別や他の Malware ファミリーとの分類を行うことは不可能であり，より汎用性のあるマスクパターンを考える必要がある。

本提案では，高次局所自己相関特徴の式(2.2)を次数 M=1 から M=7 まで拡張し，算出される 3×3 サイズのマスクパターン 221 種(図 3.1: HLAC マスクパターン 221 種)を用いて，Malware の Texture Image 全体をパターンマッチング処理することにより，局所的類似性と大局的類似性を比較する構造レベルの解析手法を導入する。ここで，次数 M において M=0 と M=8 は，隣接画素との関係から機械語命令列として見た場合有効な意味をなさないと判断し除外した。Malware Texture Image の構造レベル解析における識別能力向上には，多様なパターンが有効であるが，マスクサイズを大きくすると膨大なパターン種を扱うことになり，計算コストの上昇を招きリアルタイム処理には不適である(5×5 マスクサイズでは，$2^{24}$ 通り必要)[18]。さらに，マスクサイズを大きくするとより低周波成分が抽出できるが，機械語命令列の並びを表現するためには，3×3 程度の局所的

な構造解析が適していると判断し，このサイズを導入した。

一方，構造解析による大域的特徴は，加法性と位置不変性の性質から HLAC のマスクパターンで Malware Texture Image 全体をスキャンしたことと等価であり，パターンマッチング結果の頻度(HLAC マスクパターン種毎の累積度数)をヒストグラム化したものは，その画像が持つ性質を数値として表現することが可能である。




Mask Patterns for Malware Pattern Matching.

(221 Mask Patterns, 1st-order to 7th-order HLAC Features, 3×3 pixels)

図 3.1 Malware パターンマッチングのためのマスクパターン

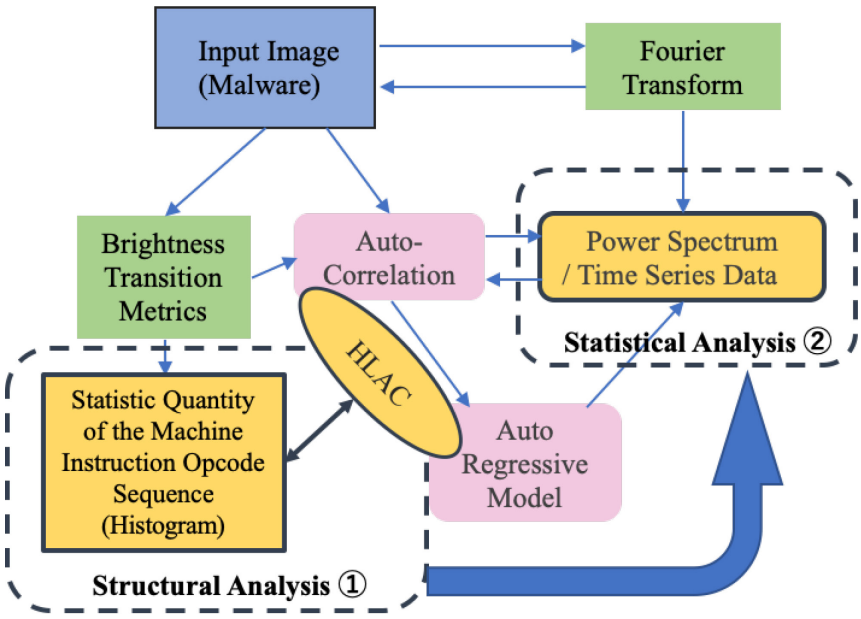
## 3.3 Pattern Matching Accelerator を適用したアルゴリズム

Malware Texture Image 全体を HLAC マスクパターン 221 種(図 3.1)でパターンマッチングさせる処理は, 非常に大きな計算コストが発生する。処理系システム全体は，シンプルな構成と計算アルゴリズムを用い，負荷の重い処理を PM Accelerator に担わせるアルゴリズムを導入する必要がある。

本提案では，高負荷処理に図 2.4 で示した PM Accelerator を導入し，極めてシンプルかつ短時間で完了させる。ここで，システムの計算コストは，目的の処理時間と消費電力である。IoT Edge Computing 実現には，PM Accelerator を組み合せたドメインスペシフィクコンピューティングのシステム構成が有効である。

Malware Texture Image 全体を HLAC マスクパターン 221 種でパターンマッチングさせた結果は，HLAC マスクパターン種毎の累積度数をヒストグラム化させる。図 3.2 で示す相関に従い Malware の解析を行う。構造レベルの解析①(Structural Analysis①)は，テクスチャを構成する Primitive を命令列群と考えて HLAC パターンを用いて構成要素の形と量に分解する。①で算出した形と量は，統計レベルの解析②(Statistical Analysis②)へ入力する数値としてヒストグラムを適用し求める。このヒストグラムの分布は，パワースペクトル/時系列データ(Power Spectrum/Time Series Data)の統計量となっており，ベクトル解析することでデータが持つ性質の分類と識別を行う。これは, HLAC が持つ加法性と位置不変性の性質からヒストグラムは，改変亜種を含む Malware ファミリー群の Malware コードのコアとなる機械語命令列がプログラムコードの何処かに必ず記述されていることを特徴として抽出し，ヒストグラムが表現する数値的分布は，同じ傾向を示すと考えたからである。即ち，Malware Texture Image の局所的な構造解析データの統計結果は, 類似する機械語命令列の分布に等しく, 画像全体が持つ大局的特徴は，パワースペクトル/時系列データ(Power Spectrum/Time Series Data)に類似した特徴を持つと推測する(図 3.2)。

よって, 本提案のアルゴリズムにより計算されたマッチングデータは, パワースペクトル/時系列データの個別ベクトルの相関解析を適用することで Malware




Relationship of Statistics applied to Malware Analysis.

図 3.2 Malware 解析に用いる統計量の相関

ファミリーの分類と Malware 判別が可能になる。

## 3.4　3 章のまとめ

本章では，提案手法と，その Malware 検出技術の新しいアルゴリズムの理論に関して説明した。

具体的には，2 段階の抽象化を行うことで，機械語命令列を構造的に解析し，最終的にスペクトル/時系列データの特徴抽出を可能とする方式理論である。最も難解な課題である「何を観測すべきか，本質的に何を見ているのか」という問題に対して，HLAC を用いたマスクパターンを採用し，PM Accelerator を用いてパターンマッチングさせたマッチ頻度を波と考え，シンプルな固有ベクトル解析へ持ち込むプロセスアイデアを導き出した。

# 第4章　Malware 識別実験

本章では，前章に示した PM Accelerator を適用した Malware 検出の新しいアルゴリズムの有効性検証に関して論ずる。

具体的には，評価サンプルの準備と評価プロセスの説明，高効率な実験環境構築の概要説明，さらには，Edge 適用のターゲットである Security Gateway の要件を満足する Malware 識別性能を有するのかどうか，データ解析手法の導出と，その検証に関して説明する。

## 4.1　評価方法

提案の目的を達成するためには，PM Accelerator を適用した Malware 検出技術の新しいアルゴリズムが，Security Gateway の要件を満たす 70%以上の Malware 識別性能を有するのか検証する必要がある。そのため，表 4.1 に示した複数の代表的な Malware サンプルファミリーのファイルを被検体データとして用意した。さらに，正常ファイルとの比較による識別も必要なことから，Linux 上で動作する通常のアプリケーションを Normal ファイルとして用意した。これらの Malware サンプル(5 種 57 個)は，日本大学理工学部応用情報工学科泉研究室のハニーポッドで収集したものと横浜国立大学理工学部情報・物理セキュリティ拠点吉岡研究室から提供を受けた Malware 検体であり，Symantec 社のウイルススキャンを用いてシグネチャによるファミリー分類を施した検体サンプルである。いずれのサンプルも，IoT デバイスを標的とした Linux 上で動作する Malware ファイルである。

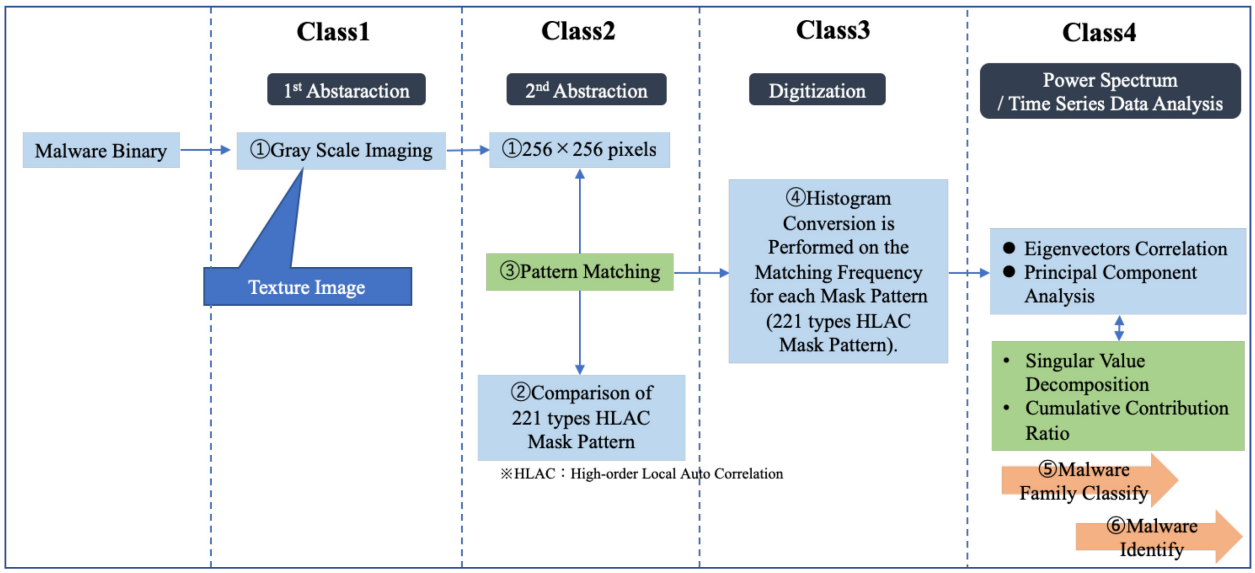
図 4.1 は，評価方法のプロセスフローを示したチャート図である。第 1 次の抽象化(Class1)として，Malware バイナリーファイルを 256×256 Pixels のグレースケール画像化(Texture Image 化)を行いデータベース化する。一方，パターンマッチングのマスクパターンは，HLAC から算出された M=1 から M=7 次元のマスクパターン 221 種をハードコーディングで用意し，定数テーブル化する。第 2 次

の抽象化(Class2)では，PM Accelerator を用いて Malware サンプルファイル毎にパターンマッチングの演算を実行する。

パターンマッチング演算は，1Pixel 毎の Bin で照合処理を実行し，HLAC マスクパターン 221 種毎に対する Malware ファイルのマッチ結果が，ファイル固有の数値データとして生成される。Class3 では，この得られた数値データに基づいて X 軸(横軸)にマスクパターン 221 種をとり，Y 軸(縦軸)に個々のマスクパターンに対応した出現頻度をとる。このようにして得られたヒストグラムは，Malware 固有のパワースペクトル/時系列データとして表現できる。Class4 において，これらパワースペクトル/時系列データの個別ベクトル相関解析および主成分ベクトル抽出による相互相関解析を経て，Malware ファミリーの分類，識別処理を行う。

表 4.1 Malware 評価サンプル

| File | | Feature | Bitmap Type | Sample Quantity |
|---|---|---|---|---|
| Malware Family Name | Grip | Distributed Denial Service Attack. | .bmp | 12 |
| | Kaiten | Botnet for IoT. Closely Related to the Mirai. | .bmp | 12 |
| | Mrblack | Linux Spike Trojan Malware. | .bmp | 12 |
| | Mirai | Botnet (The most famous). | .bmp | 9 |
| | Ganiw | Create a Backdoor. Springboard of DDoS Attacks. | .bmp | 12 |
| Not Malware | Normal | A Normal Application Running on Linux. | .bmp | 12 |




Process Flow Chart of the Algorithm.

図 4.1 アルゴリズムのプロセスフローチャート

## 4.2　実験システムの構成

前節の図 4.1 に示したアルゴリズムのプロセスフローを機能させるための実験環境(有効性検証システム)を構築する。有効性検証システムの PM Accelerator は, 提案処理方式のアルゴリズム検証を目的としたため, 検証システム全体のエミュレータを構築する手段を導入し検証実験を行っている。

さらに，今後の ASIC や FPGA への展開，産業実装応用や機能拡張を見越し，それに適用可能なモジュール構成とした。また，図 1.6 に示した Computing 階層構造と System 設計思想の課題を理解し，本質的要件を満たす方向性を持たせている。これらのことにより, 構築した当該システムのデータ処理能力と解析性能は，高い効率性を発揮すると考える。

### 4.2.1　高効率エミュレータの実現

検証システム全体のエミュレータに用いた PM Accelerator(図 2.4)のソースコードの上位構成概念を図 4.2 に示す。当該ソースコードは，文献[19]~ [21]で提案されているものであり，AOT 社が製品名: SLID として提供しているアプリケーションのプロセッサ部ソースコードである。

図 4.2 で示した SLID_emulator は，大きく 3 つの部位から構成されている。アプリケーションとしてユーザがデータ定義と操作を行う GUI.ap 部と，入出力データおよびコマンドを受け渡す VM.ap，およびパターンマッチングエンジン(SLID_core)から構成されている。本研究に用いる検証システムのエミュレータでは，SLID_core のみを抽出しパターンマッチング・コアとして適用した。

図 4.3 は Malware Detector(検出器)の構成概要を示した図である。上段の Ph.1 構成は, オーソドックスにパターンマッチング・コアへマスクパターンとイメージデータを入力する構成であり，一般的なプロセッサのレジスタファイルと演算器の組み合わせ構成から考え出したものである。しかし, 当該 Malware Detector を Edge 部分へ組み込み実装し目的の要件を満たすためには Malware 検出のリアルタイム性能が重要である。このようなレジスタファイルと演算器の組み合わせにおいて, 最もクリティカルなポイントは, 演算入力データのスループットであり，レジスタファイルからの読み出しパスのバンド幅を演算性能とバランス

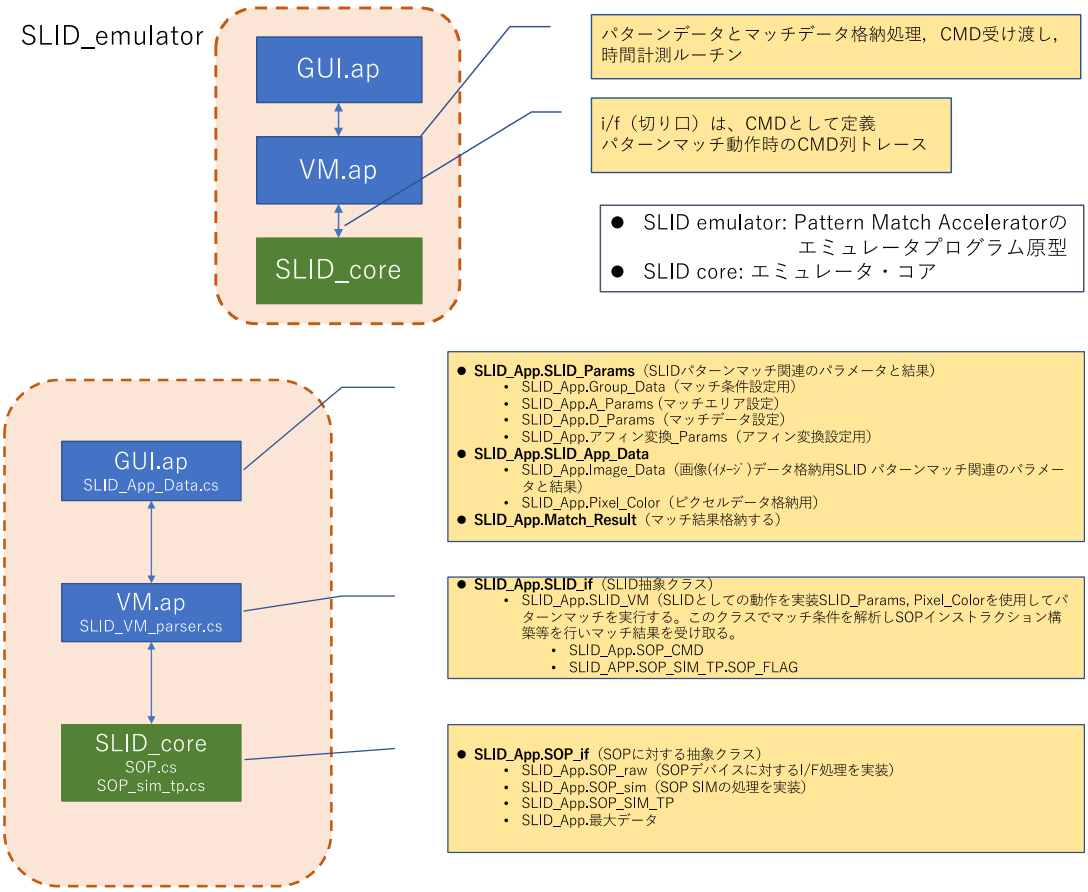



図 4.2 PM Accelerator のソースコード上位構成概念

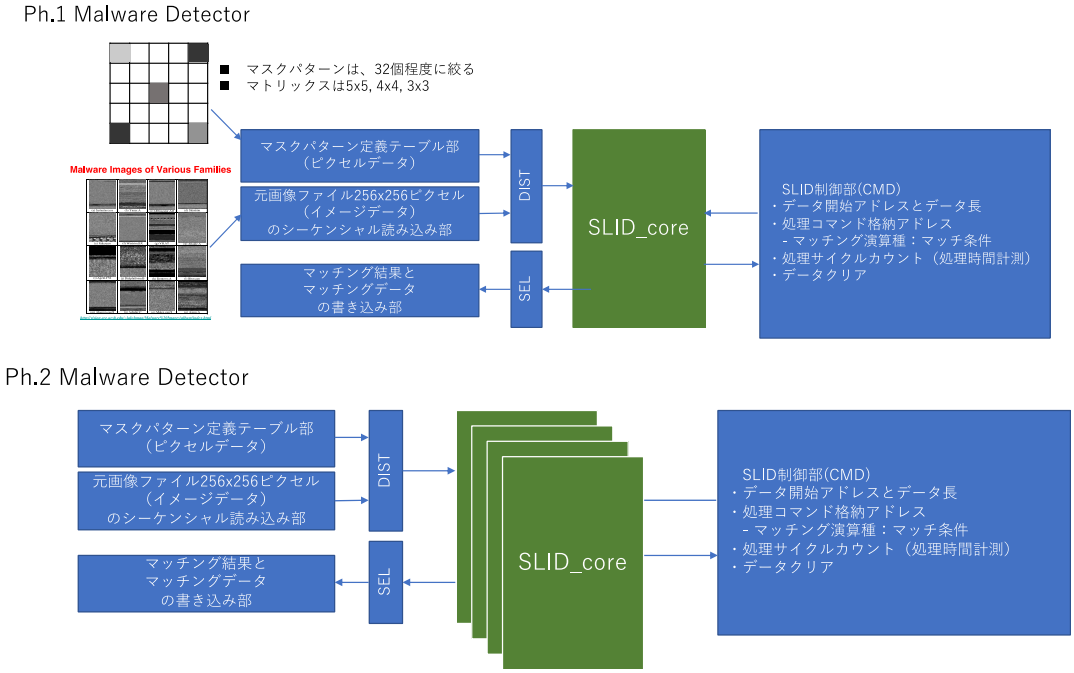



図 4.3 エミュレータ構成検討

した構成にする必要がある。図 4.3 下段の Ph.2 構成は，これらの課題を解決する手法として考え出した SLID_core の多面化案である。この構成方式は，データ振り分け機能である DIST 部の選択論理が重くなる欠点があるが，データスループットに見あう高性能化を得ることができる。

当該有効性検証システムは，アルゴリズム検証を主眼としたため，Ph.1 のオーソドックスな構成を用いて構築した。但し，実際の ASIC 化においては，アルゴリズムが持つ解析性能(演算処理時間)に応じた SLID_core 面数による Ph.2 多面化構成方式を採用すべきである。

### 4.2.2　実験システム

前節で説明した構成方式を用いて構築した本研究の有効性検証システムの概要を図 4.4 に示す。前節の通りエミュレータに適用した PM Accelerator(図 2.4)は，文献[19]~[21]で提案されているエミュレータコア(C#で記述)を適用した。既往研究結果から，Malware 画像全体 256×256 Pixels 画像に対する 1 Query(マスクパターン)の照合処理性能は，実 ASIC チップでは，~5μsec という非常に高速な処理が実現可能である。

PM Accelerator を制御する周辺の Class1 グレースケール画像化処理部，HLAC マスクパターンテーブルからのデータ読みだし処理部，照合指示発行部，パターンマッチング処理結果の統計量処理部などのサブシステムは，Python(約 1,200 行)で記述した。

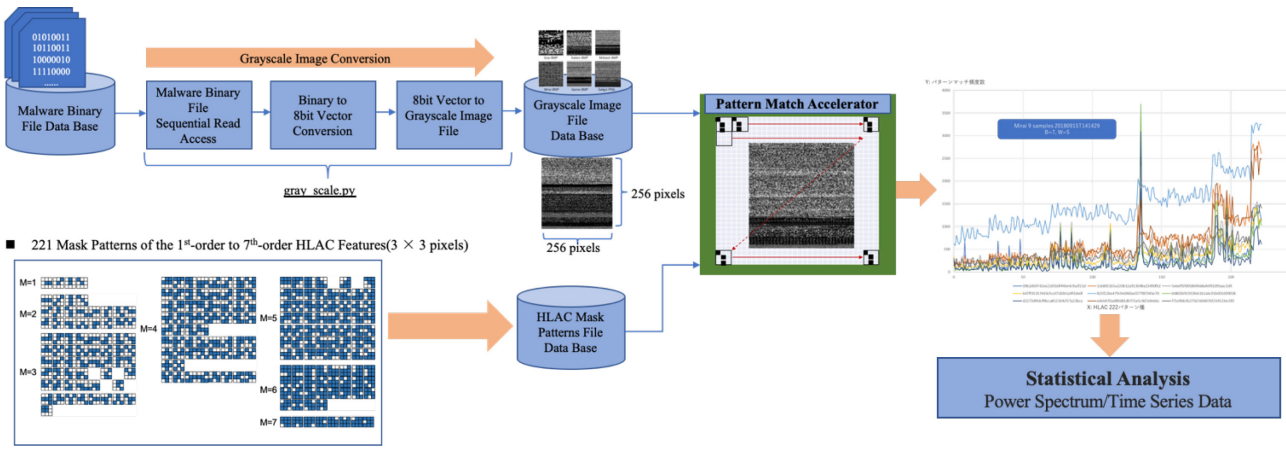



図 4.4 実験システムの構成

実装段階では ARM 系，Intel 系および RISC-V の汎用プロセッサで処理するヘテロジェニアスコンピューティング環境として SOC(System on a Chip)を組むことを想定した構成である。

### 4.2.3　RISC-V プロセッサ適用の考察

特に IoT のエッジ部分へ適用する Security Gateway のコンピュータシステムを考えた場合，消費電力やサイズに効果的な RISC-V をコアプロセッサとして採用することが有効である。図 4.5 は，RISC-V のコプロセッサとして図 2.4 の PM Accelerator を組み合わせたコンセプト図である。RISC-V の Rocket Core は，RoCC(Rocket Custom Coprocessor)と呼ぶアクセラレータ直結インターフェースを持っており，容易に提案手法の Malware 検出システムを構築可能である。

文献[17]で Ajay らは，RISC-V Rocket Core にダイレクトに BNN(Binarized Neural Networks) Specialized Accelerator を接続する構成の並列処理アーキテクチャで SOC を構築している。

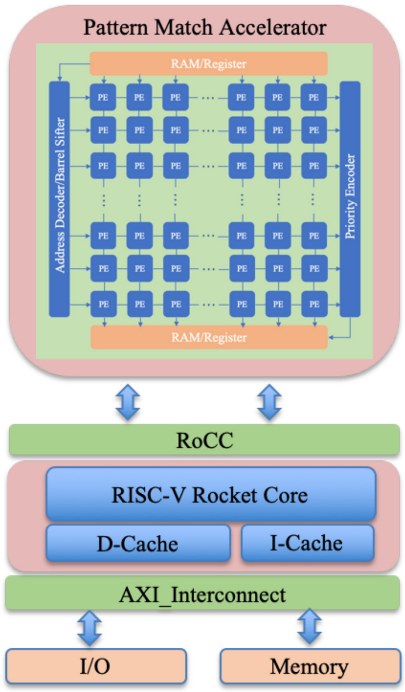



図 4.5 RISC-V プロセッサとコ・プロセッサ接続する Malware Accelerator

## 4.3　実験結果解析

図 4.4 に示す本研究が提案する IoT Edge 部分に組み込み可能な Malware 識別システムが 80%程度の識別精度を有することを，検証システム全体のエミュレータを用いた実験と結果解析から示す。

### 4.3.1　実験データ解析手法に関する考察

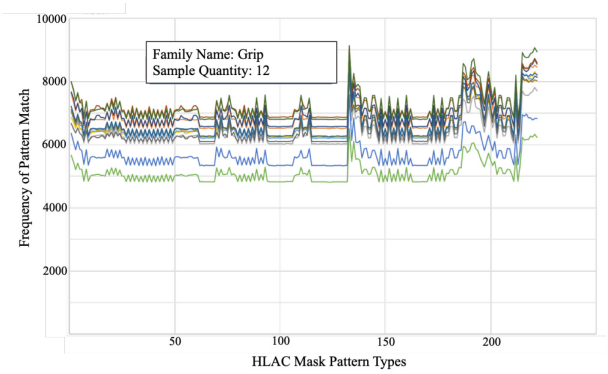
第 2 章の Malware Texture Image(図 2.1)のようにグレースケール画像化した 69 個のサンプル(57 個の Malware ファイル，12 個の Normal ファイル)に対して，第 3 章 図 3.1 に示す 221 種類の HLAC マスクパターンでパターンマッチングした実験結果データの頻度をヒストグラム化したものを，図 4.6 と図 4.7 に示す(HLAC 処理結果)。横軸は HLAC マスクパターンの次数 M=1 から M=7 までの種類番号(HLAC Mask Pattern Types:1~221)，縦軸はパターンマッチング頻度(Frequency of Pattern Match)で, Malware ファミリー毎にまとめて描画している(図 4.6, 4.7 の 6 個のヒストグラム: (a) Grip Samples, (b) Kaiten Samples, (c) Mrblack Samples, (d) Mirai Samples, (e) Ganiw Samples, (f) Normal Samples)。

図 4.6，4.7 のヒストグラムから，それぞれの Malware ファミリー毎に波形やピークに特徴があること，少数ではあるが所属ファミリー内の特徴とは異なる波形を有するサンプルが存在することが視覚的に理解できる。

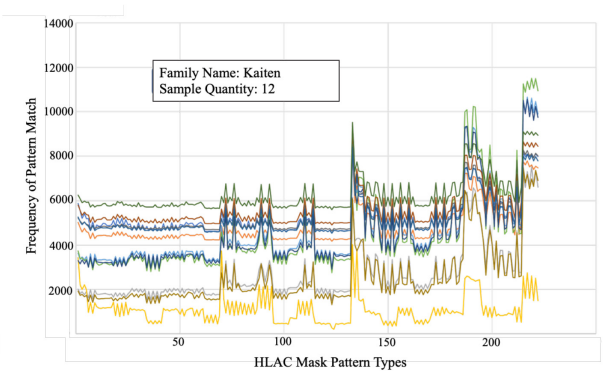
そこで，本研究ではヒストグラムの波形トレンドをパワースペクトル/時系列データとみなし，ファミリー毎の各データ群に対してシンプルなベクトル解析を用いる手法を導入し，相互相関関係に基づいて Malware を識別することを試みた。

シンプルなベクトル解析手法を用いることは, 本研究の位置付けから本質的課題解決の手段として重要なファクターである。

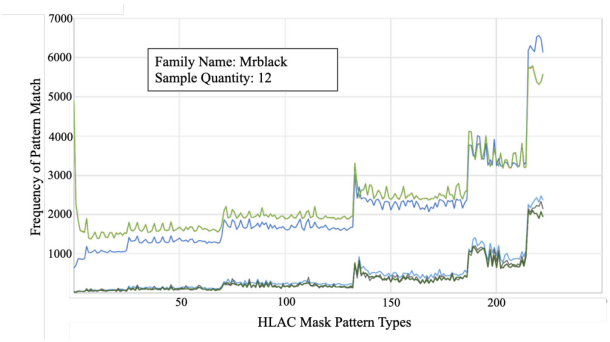
当該研究では，計算コストを縮退させ，Edge 部分でリアルタイムかつ自律的に Malware 検出処理可能なシステムを創造するという方向に合致させるため，複雑な機械学習等の計算リソースを多用する解析方式を用いない。



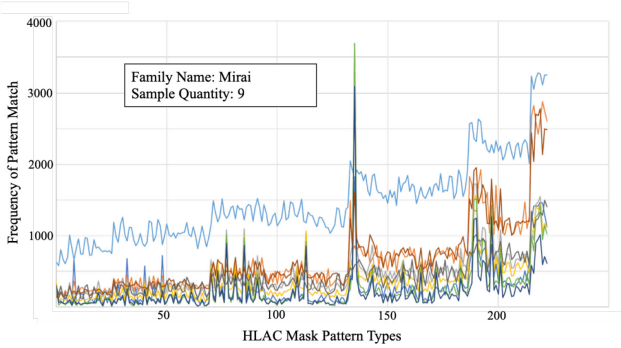

(a) Grip Samples.



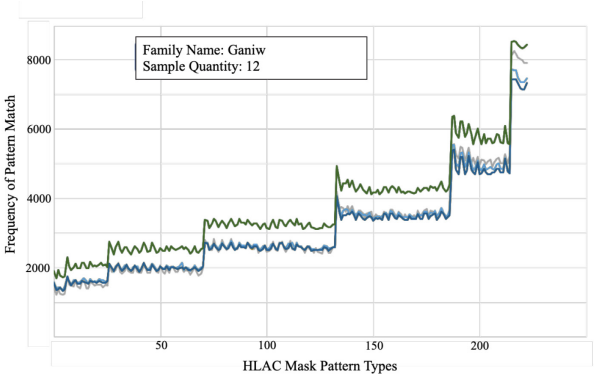

(b) Kaiten Samples.



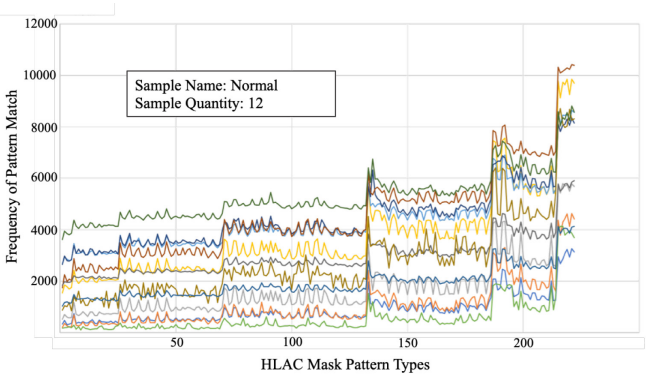

(c) Mrblack Samples.




(d) Mirai Samples.

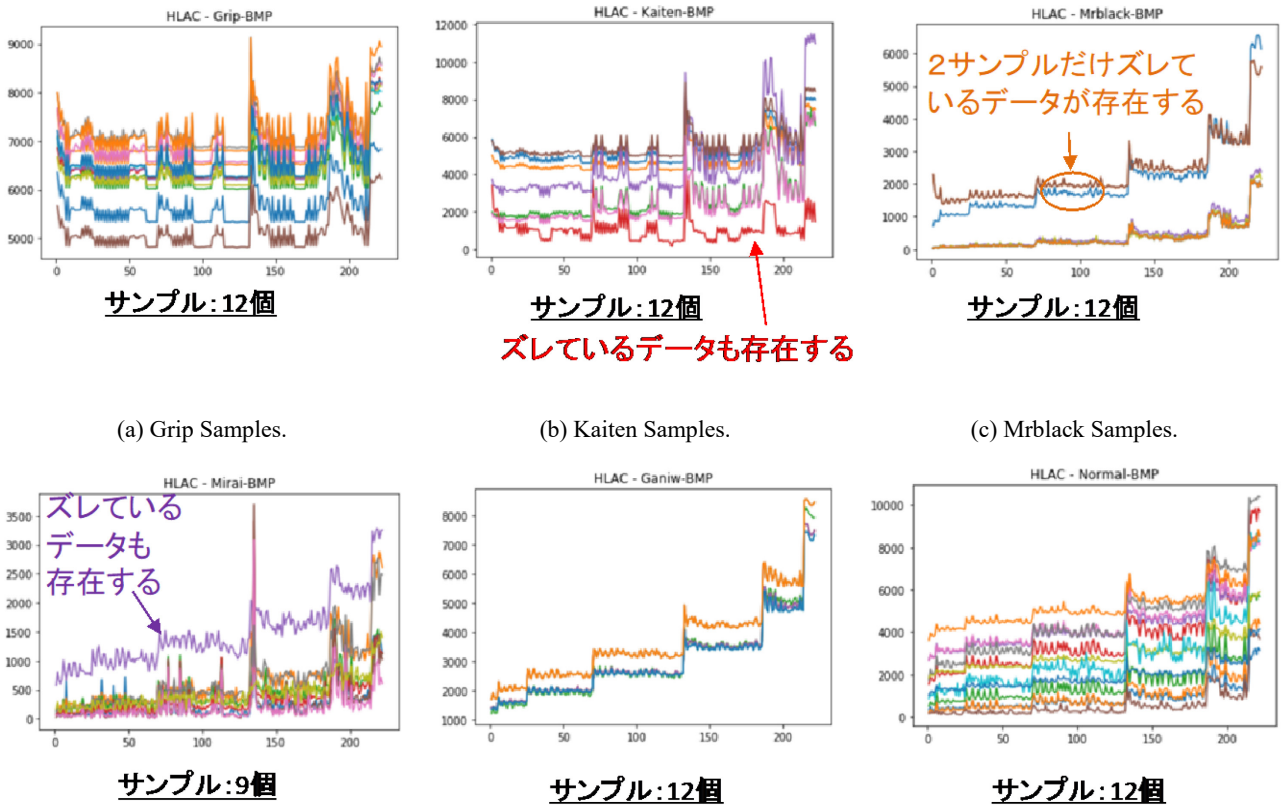



(e) Ganiw Samples.




(f) Normal Samples.

図 4.6 Malware のパターンマッチング頻度のヒストグラム




(a) Grip Samples. (b) Kaiten Samples. (c) Mrblack Samples.

(d) Mirai Samples. (e) Ganiw Samples. (f) Normal Samples.

図 4.7 Malware のパターンマッチング頻度のヒストグラム*1

*1: パターンマッチング度数(縦軸)，マスクパターン種(横軸)

### 4.3.2 モード信頼性評価基準(MAC)の適用

図 4.6 および図 4.7 に示した HLAC 処理結果の傾向から，ベクトルとして分類が可能と推測した。図 4.7 中に図示したようにサンプルの中では，外れ値(ズレているデータ)を含めた困難な全データ判別を検討するか，外れ値を除外して，外れ値以外の主要な情報の推定精度が高い判別を行うか判断が必要である。

HLAC 処理結果は，元のデータの特徴を抽出した波形であり，パワースペクトル/時系列データに類似していることから，各データ群をベクトルとして正規化し，モード信頼性評価基準(MAC: Modal Assurance Criterion)を用いた固有ベクトル相関確認手法を適用する。

ベクトル{⋯ $x_i$ ⋯ }の正規化した要素は,

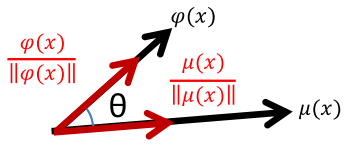
$$x_i = \frac{x_i}{\sqrt{{x_1}^2 + {x_2}^2 + \cdots + {x_n}^2}} \cdots (4.1)$$

式(4.1)と表現でき，一般的に関数で表すと

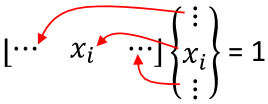
$$\frac{\varphi(x)}{\|\varphi(x)\|} \cdots (4.2)$$

式(4.2)となる。

例えば，下記２つのベクトル$\varphi(x)$と$\mu(x)$の類似性を確認するために内積を取る。内積は下記の各要素の乗算の後に各項を全て加算するため，同じベクトルであれば$\cos\theta = \cos 0 = 1$となる。




$$\frac{\varphi(x)}{\|\varphi(x)\|} \cdot \frac{\mu(x)}{\|\mu(x)\|} = \cos\theta \cdots (4.3)$$




よって，この式(4.3)で示す手法を用いることでベクトルの類似性を確認することができる。

当該手法を用いて，ベクトルを束ねた Matrix に対して適用すると，対角項だけが 1 となり，非対角項が 0 となる Matrix が生成できる。ただし，相互のベクトル間に類似性が見られると 1 に近づくため，非対角項も 1 に近づく。

HLAC 処理結果のファミリー毎に正規化ベクトルを用いた相関関係を図 4.8 に示す。図 4.8 の結果から各ファミリー種類内での相関が高いことを確認できた。

次に，Malware サンプルの全てのデータを基準に相関を計算し，平均値が最も高くなるデータを代表データに選定した。

図 4.9 の Matrix の 5 列目(赤枠)に，Normal データとその他の相互相関を数値で示す。ここで，Mrblack データと Mirai データは相関係数が 0.8 台と相関が低く，さらに Grip データは Normal サンプル間の相関平均値より低いため，閾値を平均値に設定しても判別可能と予測できる。また，Kaiten データが Normal サンプル間の平均値と同じ値を示していることから判別が難しい。一方，Ganiw データとは 0.997 以上と判別が困難であるが，図 4.8 で示したように自己サンプル相関が非常に高いため，0.999 で Ganiw と Grip は先にデータから引き抜くことができる。

以上，これら図 4.9 に示した代表データの相互相関結果より，5 種類は分類可能と考えられる。

次に同種類の中の相関が低いデータが他種類との判別に影響を与えていると仮定し，要因データを基準とした相互相関を検討する(図 4.10)。

サンプル群の相関が低いデータを判別基準より削除することで改善が見込めると予測，さらに，サンプル群の相関が高いデータと低いデータを区別して判別することで両方のデータに対応可能となる。相関関数の閾値を 0.999 にすると全ての相関が消えて，完全に判別が可能となる。しかし，逆に漏れも発生する。

最も相関が高い代表データを用いた相互相関結果を図 4.11 に示す。全サンプルから代表 5 データと全データ群の 2 セットのデータを作成し，判別検討を実施した。結果として，閾値が 0.999 で Grip と Ganiw は完全に判別可能となり，それ以外のサンプルデータでは漏れは発生するが全種類の多くのデータで判別可能となった(50/69 ≒ 72.5% ∽ 70% ≒ 44/63)。

結果として，モード信頼性評価基準(MAC)を用いた結果，約 7 割のデータで分類が可能となった。

さらに，分類性能向上追求し，相関が低い値の代表データ群を作成，1 段目で漏れたデータ群に対して判別を実施した(図 4.12)。相関が低い代表値を用いて判別することで一部のデータで改善した。

多段階の判別を実施した結果，外れ値のデータ群は相関が低いため判別が困難である。さらなる精度向上には，別手法を組み合わせて行く必要がある。

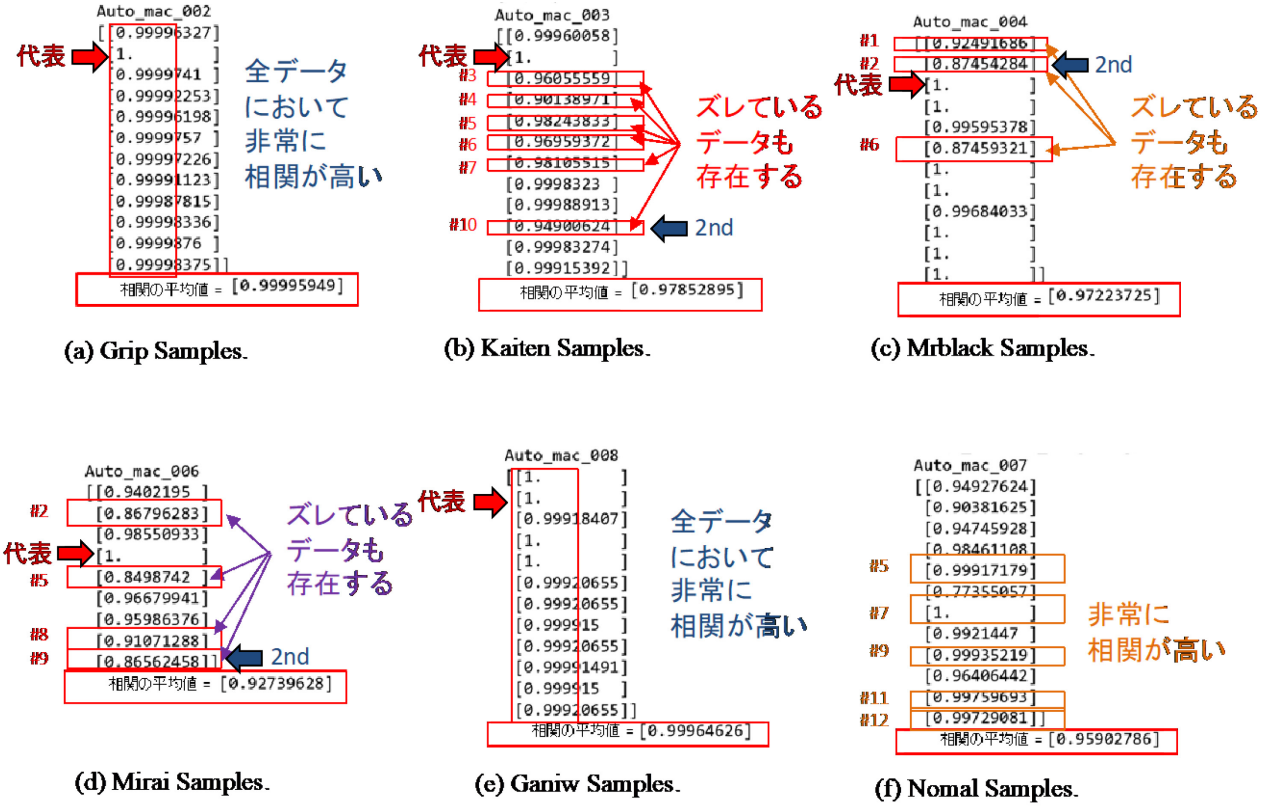



図 4.8 正規化ベクトルの相関結果

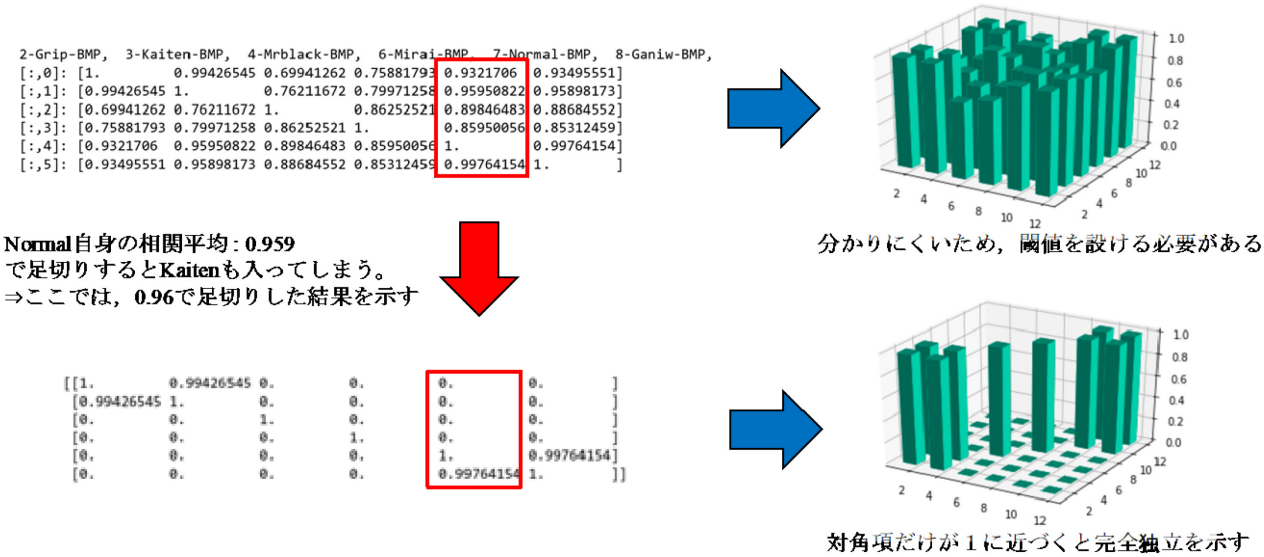
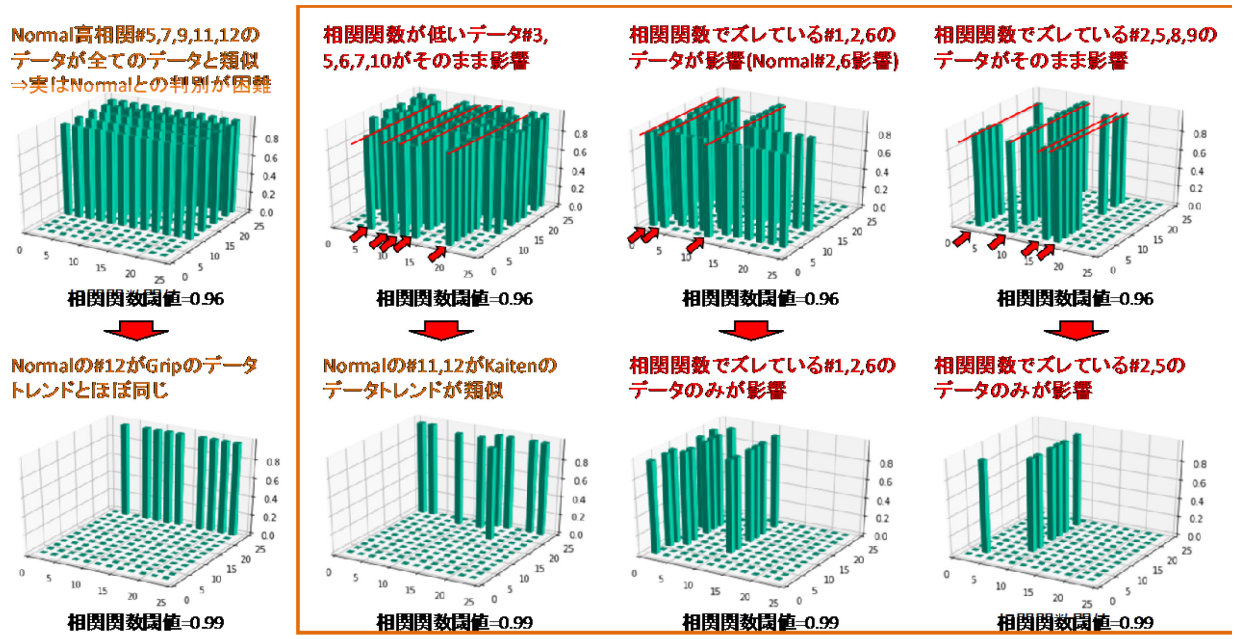



図 4.9 代表データ間の相互相関結果




(a)Grip Samples. (b)Kaiten Samples. (c)Mrblack Samples. (d)Mirai Samples.

図 4.10 要因データを基準とした相互相関結果

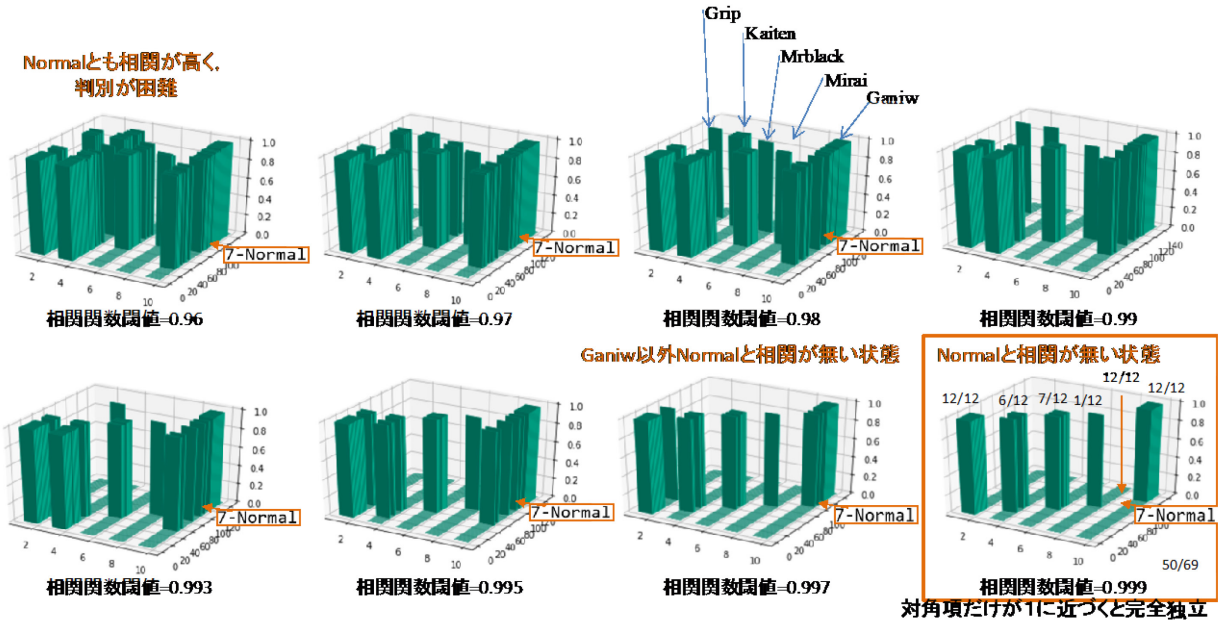



図 4.11 最も相関が高い代表データを用いた相互相関結果

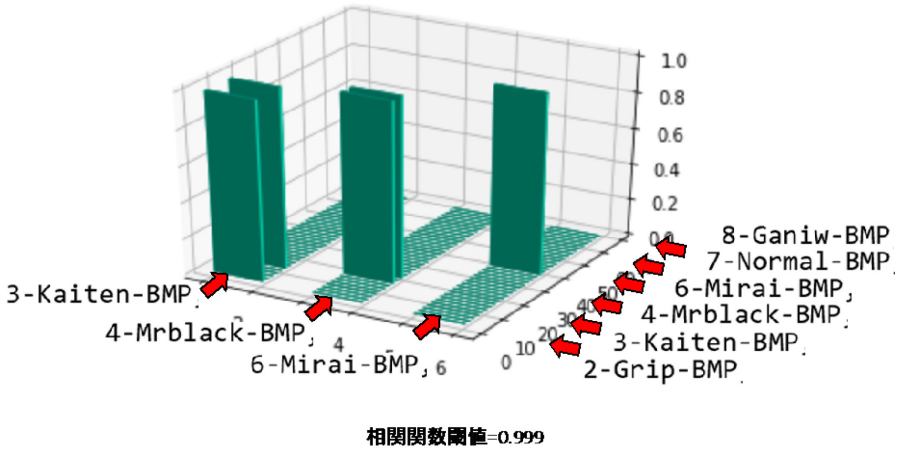



図 4.12 高相関データを除外した相互相関結果

### 4.3.3 特異値分解(SVD)を用いたノイズ除去

ファミリー種類間の相関が高いため，各データ群に対して特異値分解(SVD: Singular Value Decomposition)を用いて，それぞれの主要波形傾向だけを抜き取ったベクトルを構築し判別ベクトルとして用いることとする。

ベクトル$\{\cdots x_i \cdots\}$を束ねた Matrix は，式(4.4)と表され，

$$\mathrm{X} = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{bmatrix} \cdots (4.4)$$

この Matrix を特異値分解すると，式(4.5),(4.6)となり，

$$\mathrm{X} = U\Sigma V^T \cdots (4.5)$$

$$U\Sigma V^T = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ u_{31} & u_{32} \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} \begin{bmatrix} v_{11} & v_{21} \\ v_{12} & v_{22} \end{bmatrix} \cdots (4.6)$$

この要素 Matrix $U$ と $V$ は直行行列であるため逆行列が転置行列として求まる

計算ではこの列が出てくるため削除すると 3 つの要素 Matrix を演算して元の X に戻る

この行列を計算すると式(4.7)となり，第 1 と第 2 の特異値要素の線形結合となる。

$$U\Sigma V^T = \begin{bmatrix} u_{11}\sigma_1 v_{11} + u_{11}\sigma_2 v_{12} & u_{12}\sigma_1 v_{21} + u_{12}\sigma_2 v_{22} \\ u_{21}\sigma_1 v_{11} + u_{21}\sigma_2 v_{12} & u_{22}\sigma_1 v_{21} + u_{22}\sigma_2 v_{22} \\ u_{31}\sigma_1 v_{11} + u_{31}\sigma_2 v_{12} & u_{32}\sigma_1 v_{21} + u_{32}\sigma_2 v_{22} \end{bmatrix} \cdots (4.7)$$

第 1 要素　第 2 要素　第 1 要素　第 2 要素

例えば，データにノイズが乗っており，第 1 成分が 8 割の寄与であった場合，第 2 成分に残りのノイズが乗っていることから，第 1 成分の特異値だけを用いたベクトルを使用するとノイズ除去が可能である。

当該検討では，特異値の累積寄与を確認して採用する特異値の数を決定した。同種類のサンプルをベクトル群として，特異値分解を施した結果から累積寄与率が 9 割以上となるように特異値を選定することでノイズを除去し，主成分ベクトルを抽出した。特異値分解を用いたノイズ除去による相関結果を図 4.13 に示す。元の波形データと比較すると主成分だけを抽出しているため，波形が少し異なるが，元のデータとの相関は非常に高い。

主成分ベクトル群の相関平均が高い代表値を選定し，相関平均より少し高い値を閾値として設定し，8 割のデータが判別可能となり，判別性能が向上した(56/69 ≒ 81.1%)。よって，主成分ベクトルを用いた判別により，約 80%のデータが分類可能となる。

### 4.3.4　実験データ解析手法のまとめ

6 種類のデータ群に対して HLAC 処理した結果を用いて，6 種類のデータの識別が可能か解析した結果，以下の結論を得た。

(1) 6 種類のデータ群(Grip 12 サンプル, Kaiten 12 サンプル, Mrblack 12 サンプル, Mirai 9 サンプル, Ganiw 12 サンプル, Normal 12 サンプル)に対する HLAC 処理結果は，元データの特徴を抽出した波形と考えることができるため，振動の時系列データ解析における振動波形の固有ベクトルを判別手法と同じ処理を用いて，各サンプルを 1 つのベクトルとして正規化し，同じ種類のデータの間では非常に相関が高いことを確認できた。

(2) 6 種類のデータ群の中で，最も相関が高いデータを各種類の代表データとして選定し，6 種類間の相互相関を確認した結果，閾値を 0.999 と置くと，約 70%のデータを正しく分類できた。

(3) 残りの 3 割に関して，外れ値を用いてベクトル判別を行った結果，外れ値を多く選定(ベクトル解析の多段階化)しなければ改善が見込めないことから，当該手法の実用化にはさらなる工夫が必要である。

(4) ベクトル判別の多段階化では大きな改善が見込めないことから，別の手法として，主成分ベクトルを抽出する手法を検討した。同種類のサンプルをベクトル群として，特異値分解を施した結果から累積寄与率が 9 割以上となるように特異値を選定することでノイズを除去し，主成分ベクトルを抽出した。主成分ベクトル群の相関平均が高い代表値を選定し，相関平均を見ながら閾値を設定し，全種類のデータを約 80%のデータを正しく判別でき，判別性能が改善した。

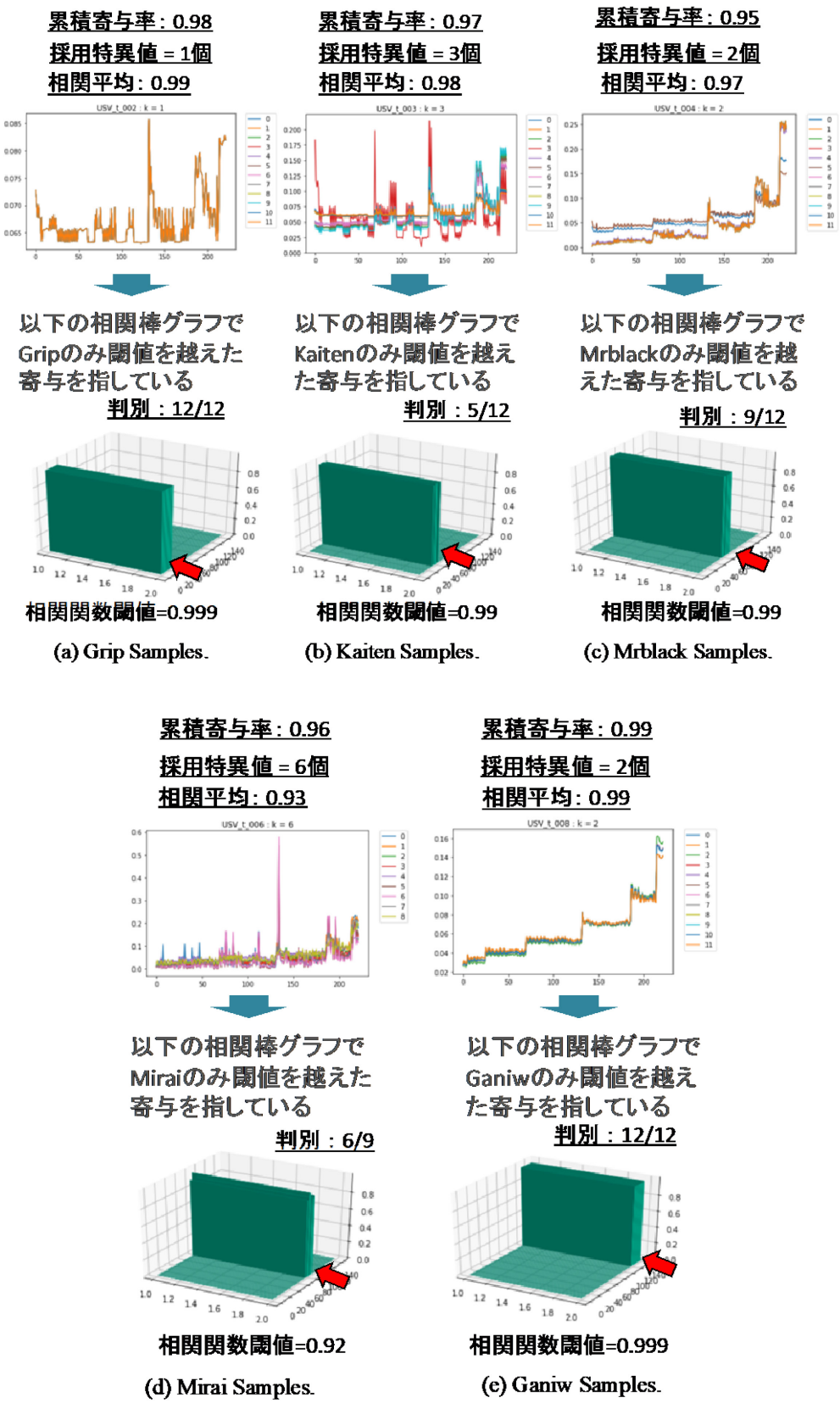



図 4.13 特異値分解によるノイズ除去後の相関結果

### 4.3.5　Malware 識別プログラムへの拡張

前述した解析手法に従い自作した Malware 識別プログラムのフローチャートを図 4.14 に示す。本プログラムは，Malware がファミリー(Gafgyt, Grip, Kaiten, Mrblack, Mirai, Ganiw, Normal)のいずれか，もしくは何にも所属しないと識別する。具体的には，サンプルのヒストグラムの生データとあるファミリー(ファミリーA)の代表データのヒストグラムのモード信頼性評価基準(MAC: Modal Assurance Criterion)値を算出する。ここで，ファミリーの代表データは，ファミリーのサンプルを特異値分解(SVD: Singular Value Decomposition)によって特徴を抽出し，かつその中で MAC 値が最大のものを選定している。算出した MAC 値と手動で設定した閾値を比較することで，MAC 値が閾値と等しいか以上，つまりファミリーA との相関関係が基準と等しいか以上ならばサンプルはファミリーA に所属と判断し，MAC 値が閾値以下，つまりファミリーA との相関関係が基準以下ならばサンプルはファミリーA に非所属と判断する。以上の処理をファミリー全てに対して行う。ここで，全てのサンプルにおいて，所属していないファミリーへの所属判定が出ないように閾値設定を行った。つまり，本プログラムでは全てのサンプルは所属ファミリーもしくは何にも所属しない，と判定され，非所属ファミリーに所属している，とは判定されない。正しい所属ファミリーに所属していると判定されたサンプルの数の割合が識別性能となる。

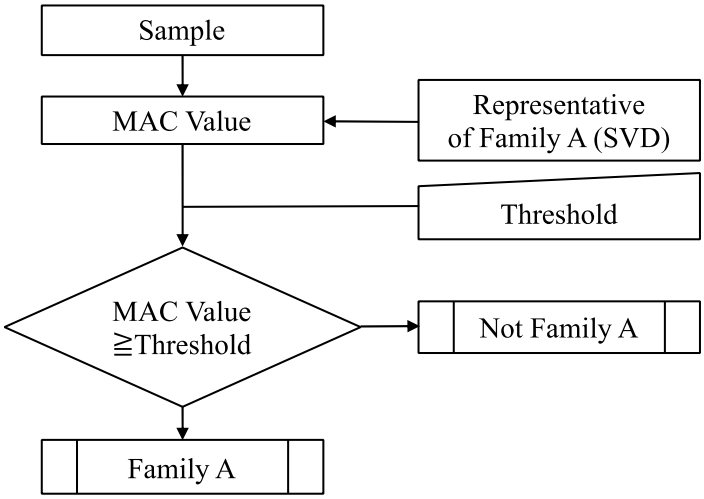



図 4.14 Malware 識別プログラムのフローチャート

まず, サンプルの MAC 値の算出方法について示す。Family(Gafgyt, Grip, Kaiten, Mrblack, Mirai, Ganiw, Normal) の $i(i = 1,2, \ldots, k,\ \ k(k = 9(\text{Family} = \text{Mirai}),\ \ k = 12(\text{Family} \neq \text{Mirai}))$ 番目のサンプルに対して HLAC$j$($j$=1,2,…,221)番目のマスクパターンのパワースペクトル頻度を$y^i_{\text{Family}}(j)$とすると，Family の$i$番目のサンプルのヒストグラムは式(4.8)のようにベクトル$\boldsymbol{\varphi}^{\boldsymbol{i}}_{\textbf{Family}}$で表記できる。その正規化ベクトルは式(4.9)のようにベクトル$\boldsymbol{\omega}^{\boldsymbol{i}}_{\textbf{Family}}$で表記できる。

$$\boldsymbol{\varphi}^{\boldsymbol{i}}_{\textbf{Family}} = \left[y^i_{\text{Family}}(1)\ \cdots\ y^i_{\text{Family}}(221)\right]^T \cdots (4.8)$$

$$\boldsymbol{\omega}^{\boldsymbol{i}}_{\textbf{Family}} = \frac{\boldsymbol{\varphi}^{\boldsymbol{i}}_{\textbf{Family}}}{\left|\boldsymbol{\varphi}^{\boldsymbol{i}}_{\textbf{Family}}\right|} \cdots (4.9)$$

サンプル $i$, $j$ 間の MAC 値$\Omega^{ij}_{\text{Family}}(i \neq j)$は式(4.9)の正規化ベクトルの内積で式(4.10)と表記できる。

$$\Omega^{ij}_{\text{Family}} = \boldsymbol{\omega}^{\boldsymbol{i}}_{\textbf{Family}}{}^T \cdot \boldsymbol{\omega}^{\boldsymbol{j}}_{\textbf{Family}} \cdots (4.10)$$

MAC 値算出の具体的イメージ(図 4.15 参照)は，次の例のようになる。

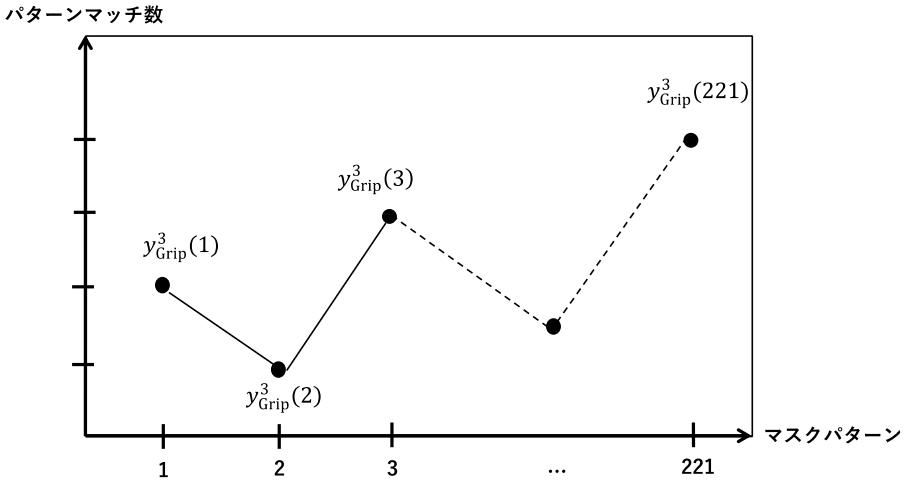



図 4.15 MAC 導出イメージ参照グラフ

Family = Grip の場合，サンプル No.3 の時,

$$\boldsymbol{\varphi}^{\boldsymbol{3}}_{\textbf{Grip}} = \left[y^3_{\text{Grip}}(1)\ y^3_{\text{Grip}}(2)\ \cdots\ y^3_{\text{Grip}}(221)\right]^T \cdots (4.11)$$

$$= \begin{bmatrix} y^3_{\mathrm{Grip}}(1) \\ y^3_{\mathrm{Grip}}(2) \\ \vdots \\ y^3_{\mathrm{Grip}}(221) \end{bmatrix} \cdots (4.12)$$

$$\boldsymbol{\omega^3_{Grip}} = \frac{\boldsymbol{\varphi^3_{Grip}}}{|\boldsymbol{\varphi^3_{Grip}}|} \cdots (4.13)$$

$$\boldsymbol{\omega^3_{Grip}} = \frac{1}{\sqrt{(y^3_{\mathrm{Grip}}(1))^2 + (y^3_{\mathrm{Grip}}(2))^2 + \cdots + y^3_{\mathrm{Grip}}(221)^2}} \begin{bmatrix} y^3_{\mathrm{Grip}}(1) \\ y^3_{\mathrm{Grip}}(2) \\ \vdots \\ y^3_{\mathrm{Grip}}(221) \end{bmatrix} \cdots (4.14)$$

Family = Grip, サンプル No.4 の場合も同様に,

$$\boldsymbol{\omega^4_{Grip}} = \frac{1}{\sqrt{(y^4_{\mathrm{Grip}}(1))^2 + (y^4_{\mathrm{Grip}}(2))^2 + \cdots + y^4_{\mathrm{Grip}}(221)^2}} \begin{bmatrix} y^4_{\mathrm{Grip}}(1) \\ y^4_{\mathrm{Grip}}(2) \\ \vdots \\ y^4_{\mathrm{Grip}}(221) \end{bmatrix} \cdots (4.15)$$

Family = Grip, サンプル No.3, 4 間の MAC 値は,

$$\Omega^{34}_{\mathrm{Grip}} = \boldsymbol{\omega^3_{Grip}}^{T} \cdot \boldsymbol{\omega^4_{Grip}} \cdots (4.16)$$

で与えられる。

次に, ファミリーサンプルの特徴抽出のための SVD の算出方法について示す。式(4.8)から, Family のサンプル全てのヒストグラムは式(4.17)に示す行列 $\boldsymbol{M_{Family}}$(221 行 $k$ 列)と表記できる。

$$\boldsymbol{M_{Family}} = [\boldsymbol{\varphi^1_{Family}} \cdots \quad \boldsymbol{\varphi^k_{Family}}] \cdots (4.17)$$

$\boldsymbol{M_{Family}}$は, 特異値分解によって式(4.18)のように表記できる。ここで, $r$は$\boldsymbol{M_{Family}}$の階数($r = \mathrm{rank}(\boldsymbol{M_{Family}})$)とすると, $\boldsymbol{U_{Family}}$は左特異行列(221 行 $k$ 列), $\boldsymbol{\Sigma_{Family}}$は対角成分が特異値$\sigma_l (l = 1,2, \dots, r)$の式(4.19)に示す行列($k$ 行 $k$ 列), $\boldsymbol{V_{Family}}$は右特異行列($k$ 行 $k$ 列)である。ここで, 式(4.20)を満たす最小の整数$m(m \leq r)$を求める。$\boldsymbol{U_{Family}}$の 0～$m$ 行部分の行列を$\boldsymbol{U'_{Family}}$, $\boldsymbol{\Sigma_{Family}}$の 0～$m$ 行部分の行列を$\boldsymbol{\Sigma'_{Family}}$, $\boldsymbol{V^T_{Family}}$の 0～$m$ 列部分の行列を$\boldsymbol{V^{T'}_{Family}}$, とすると, SVD によってファミリーの特徴 97%を抽出した行列$\boldsymbol{M'_{Family}}$(221 行 $k$ 列)は式(4.21)のようになる。

$$\boldsymbol{M}_{\mathbf{Family}} = \boldsymbol{U}_{\mathbf{Family}}\boldsymbol{\Sigma}_{\mathbf{Family}}\boldsymbol{V}^{\boldsymbol{T}}_{\mathbf{Family}} \cdots (4.18)$$

$$\boldsymbol{\Sigma}_{\mathbf{Family}} = \begin{bmatrix} \sigma_1 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & \sigma_2 & \ddots & & \vdots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & & \ddots & \sigma_{r-1} & 0 & \vdots \\ 0 & \cdots & \cdots & 0 & \sigma_r & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 \end{bmatrix} \cdots (4.19)$$

$$\frac{\sigma_1 + \sigma_2 + \cdots + \sigma_m}{\sigma_1 + \sigma_2 + \cdots + \sigma_r} \geq 0.97 \cdots (4.20)$$

$$\boldsymbol{M}'_{\mathbf{Family}} = \boldsymbol{U}'_{\mathbf{Family}}\boldsymbol{\Sigma}'_{\mathbf{Family}}\boldsymbol{V}^{\boldsymbol{T}'}_{\mathbf{Family}} \cdots (4.21)$$

次に，MAC 値が最大のものをファミリー代表データに選定する。式(4.21)の $\boldsymbol{M}'_{\mathbf{Family}}$を式(4.17)の左辺に代入し，式(4.17)右辺と式(4.9),(4.10)から MAC 値が算出できる。その MAC 値が最大のものをファミリー代表データに選定した。

以上で設定したファミリー代表データと先述した閾値を用いて, 図 4.14 に示すプログラムによって Malware 評価サンプル(表 4.1)の識別を行う。

### 4.3.6　解析プログラムによる実測

前述のアルゴリズムにより作成した解析プログラムにより選定したファミリー代表データと所属ファミリーデータ群との MAC 値を表 4.2 に示す。

Grip，Ganiw はファミリー内の相関が非常に高く，Mirai は相関が低いことがわかる。これら代表データと，表 4.3 に示す閾値を解析プログラムに用いて Malware 識別した結果を表 4.3 に示す。表 4.3 の結果から，ファミリー内の相関が高い Grip，Ganiw は 12 サンプル中 12 個の識別が可能で 100%の識別精度であるが，相関が低い Kaiten と Mirai は，それぞれ 42%，63%の識別精度となった。

総合して，69 個のサンプルに対して 56 個のサンプルが正しく識別され，本 Malware 検出システムは約 81%の識別精度であった。

### 4.3.7　追加サンプルによる検証

前節の結果より，本論のアルゴリズムと当該解析プログラムのポテンシャルを確認することはできた。しかし, 認識率の劣る Kaiten や Mirai 等の特定 Malware ファミリーの分別性能課題解析や新しい Malware 種への対応性を見極める意味合いから，Malware サンプル数追加と新規の Malware 種(Gafgyt)を加えた追加試験を試みた。追加試験に用いたサンプルを表 4.4 に示す。全ての Malware サンプルおよび Normal サンプルは，前節の実験で使用したサンプルとは異なるものであり，無作為に選び出したデータになる。また，同様にこれら Malware サンプル(6 種 641 個)は，日本大学理工学部応用情報工学科泉研究室のハニーポッドで収集したものと横浜国立大学理工学部情報・物理セキュリティ拠点吉岡研究室から提供を受けた Malware 検体であり，Symantec 社のウイルススキャンを用いてシグネチャによるファミリー分類を施した検体サンプルである。いずれのサンプルも，IoT デバイスを標的とした Linux 上で動作する Malware ファイルである。

前節同様に，Malware Texture Image としてグレースケール画像化した 751 個のサンプル(641 個の Malware ファイル，110 個の Normal ファイル)に対して，221 種類の HLAC マスクパターンでパターンマッチングした実験結果データの頻度をヒストグラム化したものを，図 4.15 の①と②に示す。横軸は HLAC マスクパターンの次数 M=1 から M=7 までの種類番号(HLAC Mask Pattern Types:1~221)，縦軸はパターンマッチング頻度(Frequency of Pattern Match)で，Malware ファミリー毎にまとめて描画している(図 4.15①②の 7 つのヒストグラム: (a) Gafgyt Samples, (b) Grip Samples, (c) Kaiten Samples, (d) Mrblack Samples, (e) Mirai Samples, (f) Ganiw Samples, (g) Normal Samples)。

図 4.15①②のヒストグラムも，ファミリー毎に図 4.6 と同様の傾向を持ち，波形やピークに相似な特徴があること，少数ではあるが所属ファミリー内の特徴とは異なる波形を有するサンプルが存在することが視覚的に理解できる。

これらの HLAC 処理結果に対して，当該解析プログラムを用いて Malware 識別処理を実行した。

表 4.2 MAC Value of Representatives on Families.

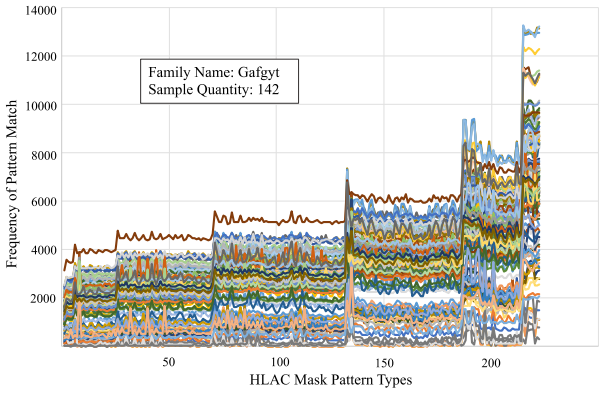
| File | | MAC Value |
|---|---|---|
| Malware Family Name | Grip | 0.999 |
| | Kaiten | 0.980 |
| | Mrblack | 0.974 |
| | Mirai | 0.933 |
| | Ganiw | 0.999 |
| Not Malware | Normal | 0.975 |

表 4.3 Threshold of Mac value after.

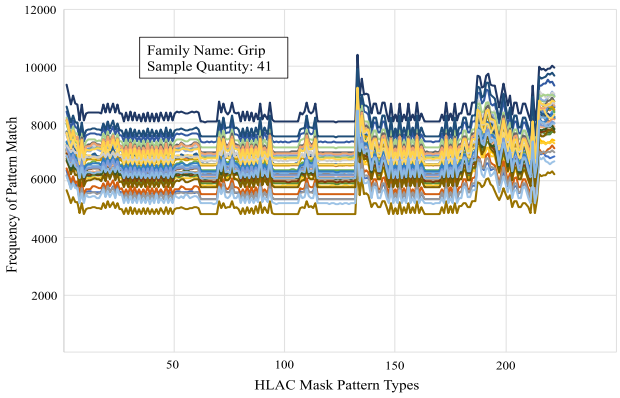
| File | | Threshold Value | Successful Identification Samples |
|---|---|---|---|
| Malware Family Name | Grip | 0.999 | 12/12 |
| | Kaiten | 0.990 | 5/12 |
| | Mrblack | 0.980 | 9/12 |
| | Mirai | 0.933 | 6/9 |
| | Ganiw | 0.999 | 12/12 |
| Not Malware | Normal | 0.980 | 12/12 |
| Sum | | | 56/69 |

表 4.4 Malware 評価サンプル(追加試験向け)

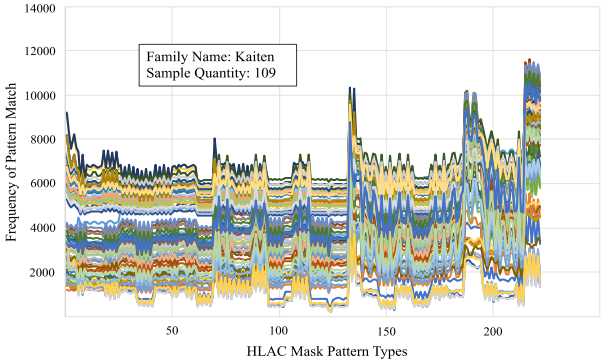
| File | | Feature | Bitmap type | Quantity |
|---|---|---|---|---|
| Malware Family Name | Gafgyt | Botnet (The most famous). | .png | 142 |
| | Grip | Distributed Denial Service Attack. | .bmp | 41 |
| | Kaiten | Botnet for IoT.<br>Closely Related to the Mirai. | .bmp | 109 |
| | Mrblack | Linux Spike Trojan Malware. | .bmp | 99 |
| | Mirai | Botnet (The most famous). | .png | 134 |
| | Ganiw | Create a Backdoor.<br>Springboard of DDoS Attacks. | .bmp | 116 |
| Not Malware | Normal | A Normal Application<br>Running on Linux. | .bmp | 110 |



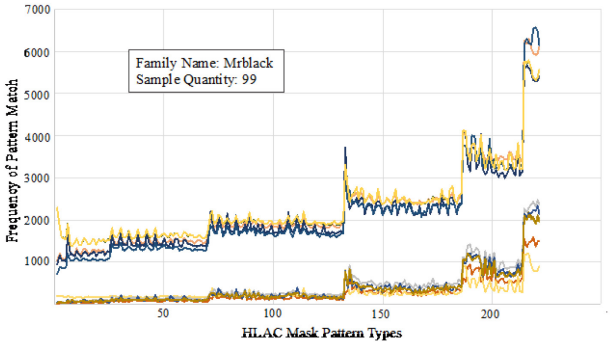

(a) Gafgyt Samples.




(b) Grip Samples.

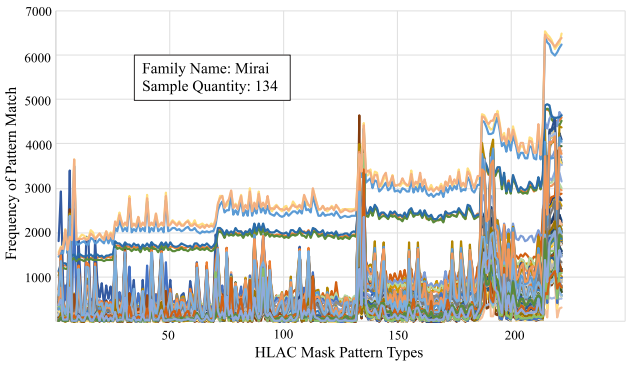



(c) Kaiten Samples.

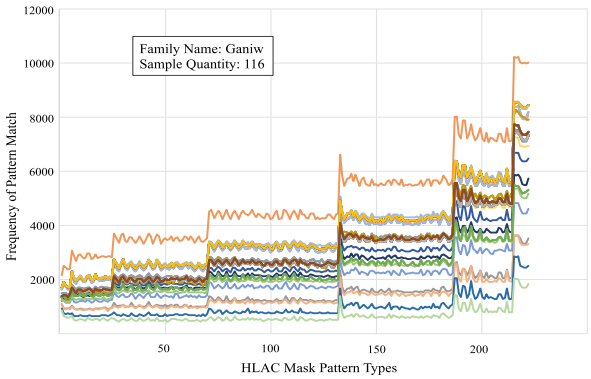



(d) Mrblack Samples.

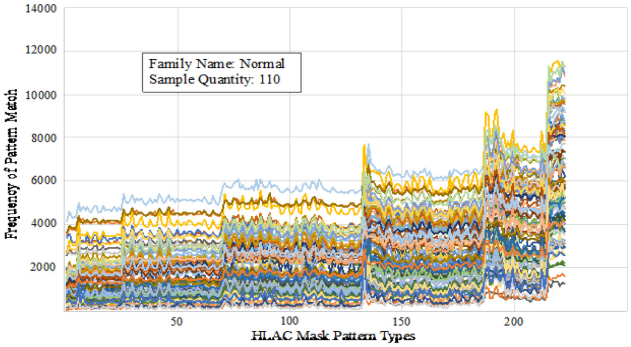
図 4.15 Malware のパターンマッチング頻度のヒストグラム(追加試験サンプル)①




(e) Mirai Samples.




(f) Ganiw Samples.




(g) Normal Samples.

図 4.15 Malware のパターンマッチング致頻度のヒストグラム(追加試験サンプル)②

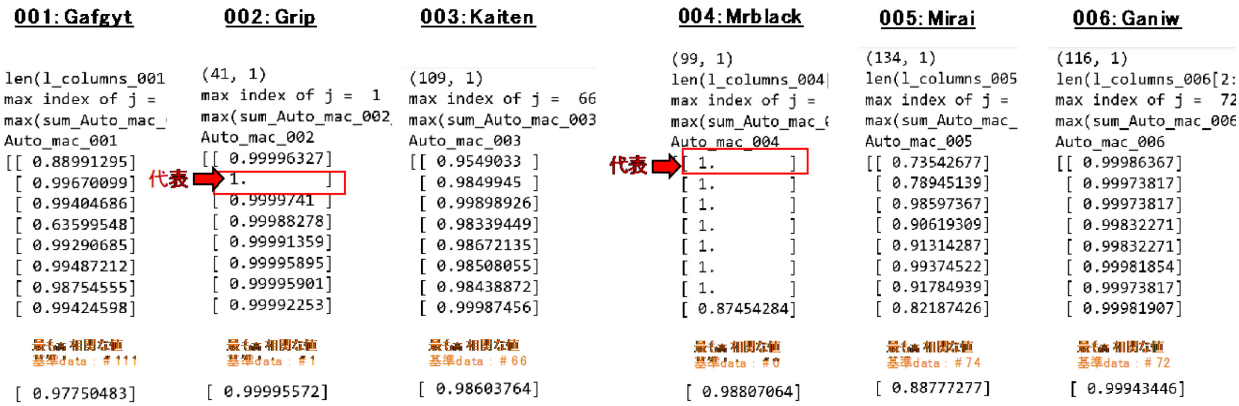



図 4.16 正規化ベクトル相関(追加試験サンプル)

## 4.4　追加試験による Malware 識別結果

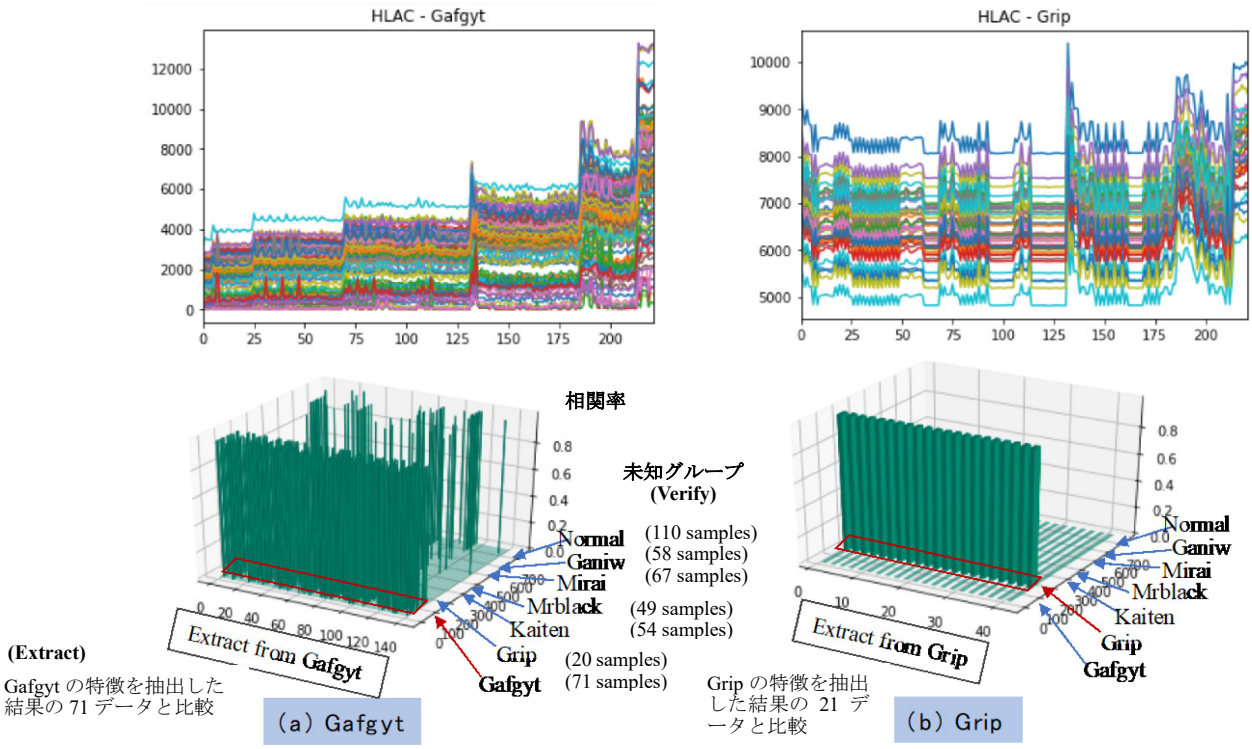
図 4.16 の正規化ベクトル相関から選定したファミリー代表データと所属ファミリーデータ群との MAC 値を表 4.5 に示す。Grip，Ganiw はファミリー内の相関が非常に高く，Mirai は相関が低いことがわかる。これら代表データと，表 4.6 に示す閾値を当該解析プログラムに用いて Malware 識別した結果を表 4.6 に示す。表 4.6 の結果から，ファミリー内の相関が高い Grip は，41 サンプル中 41 個の識別が可能で 100%の識別精度，Ganiw は 116 サンプル中 104 個 90%の識別精度であるが，相関が低い Mirai は，63%の識別精度となった。

総合して，751 個のサンプルに対して 616 個のサンプルが正しく識別され，本 Malware 検出システムは約 82%の識別精度であった。

次に当該アルゴリズムの有効性を検証するために未知データを想定した識別性能について解析した。各 Malware ファミリー種類 641 個のサンプルを 2 つに分け，合計 322 個のサンプルから同様のアルゴリズムに従い特徴を抽出する。特徴抽出に用いなかった残りの Malware 319 個と Normal サンプル 110 個を混ぜ合わせ 429 個の未知グループを作成し，これらのデータグループに対して当該アルゴリズムを用い先述の手法により識別率を算出した。

図 4.17 の①と②は Malware ファミリー毎にパワースペクトル/時系列データの主成分ベクトル解析によるサンプル間の相互相関を視覚化したグラフである。これらの図 4.17 (a)から図 4.17(g)は, 特徴抽出に用いたサンプル(表 4.7 の Extract)の特徴ベクトル群と未知データグループの特徴ベクトル群の特異値分解(SVD)後の相関を表現している。これら図 4.17 のグラフからも(b) Grip，(d) Mrblack，(f) Ganiw に関して，非常に高い分別性能であることが視覚的に読み取れる。

以上述べた解析手法の結果より，未知データを想定した Malware の検出において，本研究の実験システムとアルゴリズム，解析プログラムは 79.6%の識別能力を示した(表 4.7)。



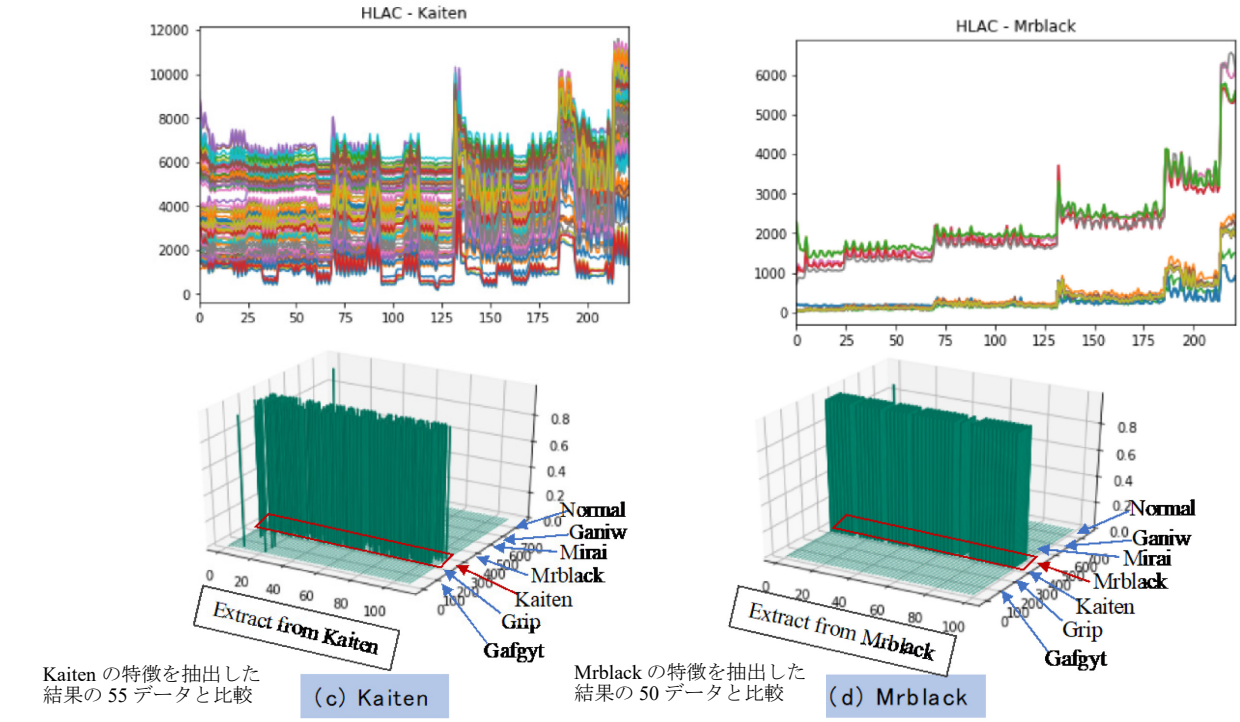




図 4.17 主成分ベクトル解析と相互相関①

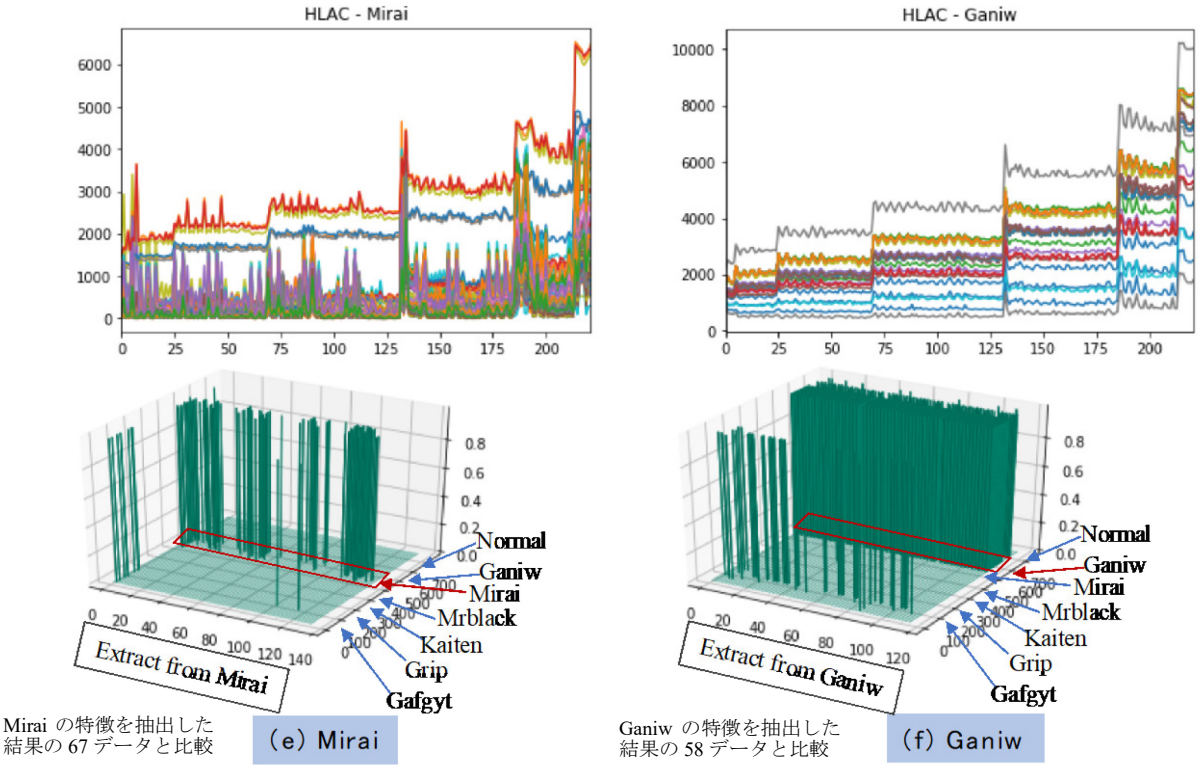



図 4.17 主成分ベクトル解析と相互相関②

表 4.5 MAC Value of Representatives on Families.

| File | | MAC Value |
|---|---|---|
| Malware Family Name | Gafgyt | 0.977 |
| | Grip | 0.999 |
| | Kaiten | 0.986 |
| | Mrblack | 0.988 |
| | Mirai | 0.888 |
| | Ganiw | 0.999 |
| Not Malware | Normal | 0.985 |

表 4.6 Threshold of MAC Value After.

| File | | Threshold Value | Successful Identification Samples |
|---|---|---|---|
| Malware Family Name | Gafgyt | 0.999 | 106/142 |
| | Grip | 0.998 | 41/41 |
| | Kaiten | 0.993 | 88/109 |
| | Mrblack | 0.992 | 86/99 |
| | Mirai | 0.979 | 84/134 |
| | Ganiw | 0.999 | 104/116 |
| Not Malware | Normal | 0.999 | 107/110 |
| Sum | | | 616/751 |

表 4.7 Malware Detection Performance.

| File | | Extract | Verify | Detectable | Detection Rate |
|---|---|---|---|---|---|
| Malware Family Name | Gafgyt | 71/142 | 71/142 | 53/71 | 74.6% |
| | Grip | 21/41 | 20/41 | 20/20 | 100.0% |
| | Kaiten | 55/109 | 54/109 | 44/54 | 81.5% |
| | Mrblack | 50/99 | 49/99 | 43/49 | 87.8% |
| | Mirai | 67/134 | 67/134 | 42/67 | 62.7% |
| | Ganiw | 58/116 | 58/116 | 52/58 | 89.7% |
| Not Malware | Normal | - | 110 | - | - |
| Sum | | | 429 | 254/319 | 79.6% |

## 4.5　4 章のまとめ

本章では, 3 章で示した提案手法とアルゴリズムの有効性検証を実験システムによる実測と，そのデータ解析を行い確認した。

実測値に対する解析は，モード信頼性評価基準(MAC: Model Assurance Criterion)と主成分ベクトルから特異値分解(SVD: Singular Value Decomposition)を用いて特徴抽出する手法により, 実験システムと本研究のアルゴリズム, 解析プログラムは 79.6%の Malware 識別能力を示した。

これらのことにより，車載 Security Gateway への搭載要件である 7 割の検出率をクリアできることを証明した。

# 第5章　提案手法の有効性と課題に関する考察

本章では，前章の Malware 識別実験結果から考察する提案手法の有効性と課題に関して論ずる。具体的には，①未検知や誤検知サンプルおよび難読化に関する定量的推察，②識別性能に大きく影響すると考えられているコンピュータアーキテクチャの問題，さらに，③車載実装における Edge 適合性能に関して説明する。

## 5.1　検知性能と誤検知

### 5.1.1　検知性能に関する考察

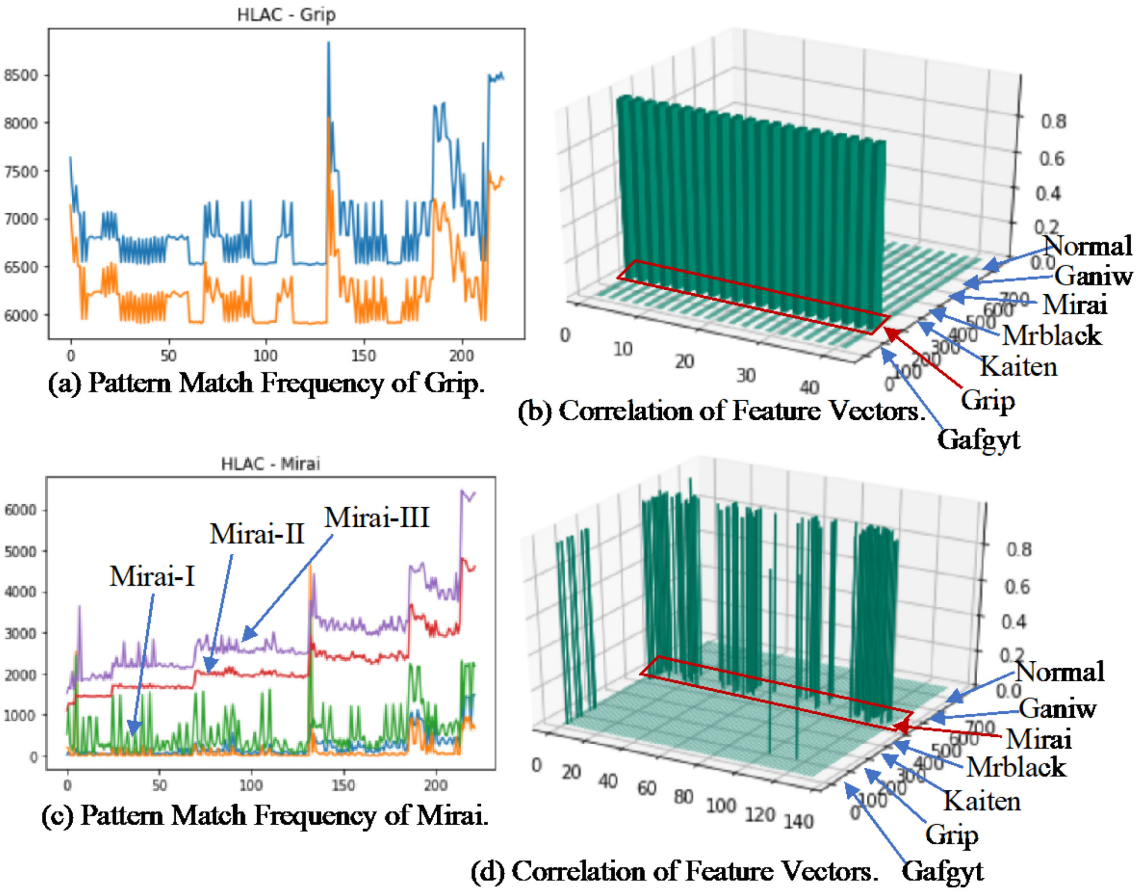
図 4.4 に示した本研究が提案する IoT Edge に組み込み可能な Malware 検出システムの識別性能に関して，性能を支配する要因と誤検知について考察する。尚，考察に適用した解析結果は，追加試験で実施したものである。

Malware ファミリー6 種の中で識別性能が劣る Gafgyt と Mirai は，幾つかのサンプルが近似なデータを持っており，Mirai が Gafgyt と判定される場合がある。閾値を高くし，近似データの識別が可能な閾値とすると，Gafgyt ファミリー内の不判定が増加し，本来の正しく判定を受けるサンプルが減り識別性能は下がる。要因は，Gafgyt と Mirai が，表 4.4 に示したように Botnet という同一攻撃タイプに起因したバイナリーコードの類似性にあると推定される。

図 5.1 は, 識別率の高い Grip と識別率の低い Mirai についてパワースペクトルの特徴を解析した図である。図 5.1(a)は，Grip の High/Low を抜き出した波形トレンドであり，図 5.1(c)は，Mirai の大きく３種に分かれた波形トレンドである。また，図 5.1(b)と図 5.1(d)は，Grip と Mirai に関して，特徴抽出に用いたサンプ

ル(Extract)の特徴ベクトル群と未知データグループの特徴ベクトル群の特異値分解(SVD)後の相関を表現した図である。

Grip に関しては，波形トレンドの High/Low が平行移動しただけの同一波形トレンドであり，特徴ベクトル相関に関しても未知データグループ内に埋もれていた Grip(Verify)と完全に一致することが視覚的に理解できる。他方，Mirai に関しては，難読化によると推測される複数の異なる波形トレンドが存在する。図 5.1(d)の特徴ベクトル相関図を見ると Gafgyt 及び Ganiw と一致する箇所があり，これが識別性能の低下要因だと考える。ところで，識別率の向上策に関しては，図 5.1(c)に図示したように複数の波形トレンド群を Mirai-I，Mirai-II，Mirai-III と新しい異種ファミリーと再定義し，特徴ベクトルの再構築を行えば，識別性能の向上が可能であると考える。




Texture Image Based Features (Grip and Mirai).

図 5.1 Malware Texture Image の基本的特徴

### 5.1.2　誤検知に関する考察

誤検知に関しては，Malware が Normal と判定される場合が，Gafgyt:23.7%，Kaiten:1.8%，Ganiw:12.7%となった。他方，識別性能が低い Mirai が Normal と判定される誤検知は，0%であった。図 5.2 は，Normal への誤判定率が高い Gafgyt について，パワースペクトルの特徴を解析した図である。図 5.2(a)は，Gafgyt の Verify に用いたサンプル 71 個の波形トレンドであり，図 5.2(b)は，特徴抽出に用いた Gafgyt(Extract)の特徴ベクトル群と未知データグループの特徴ベクトル群の相関(SVD 後)を表現した図である。図 5.2(b)からも，未知データグループに埋もれた Normal を Gafgyt 自身であると誤判定している割合の多さが視覚的に分かる。さらに，図 5.2(c)は，正判定 53 個の波形トレンドであり，図 5.2(d)は，Normal と誤判定した 17 個の波形トレンドである。図 5.2(d)より，視覚的には正判定の波形トレンドと類似していると見えるため，これら誤判定したサンプルの波形トレンドと正判定の波形トレンドから識別用の特徴ベクトル群を再構築することで，誤検知率を下げることが可能であると考えている。

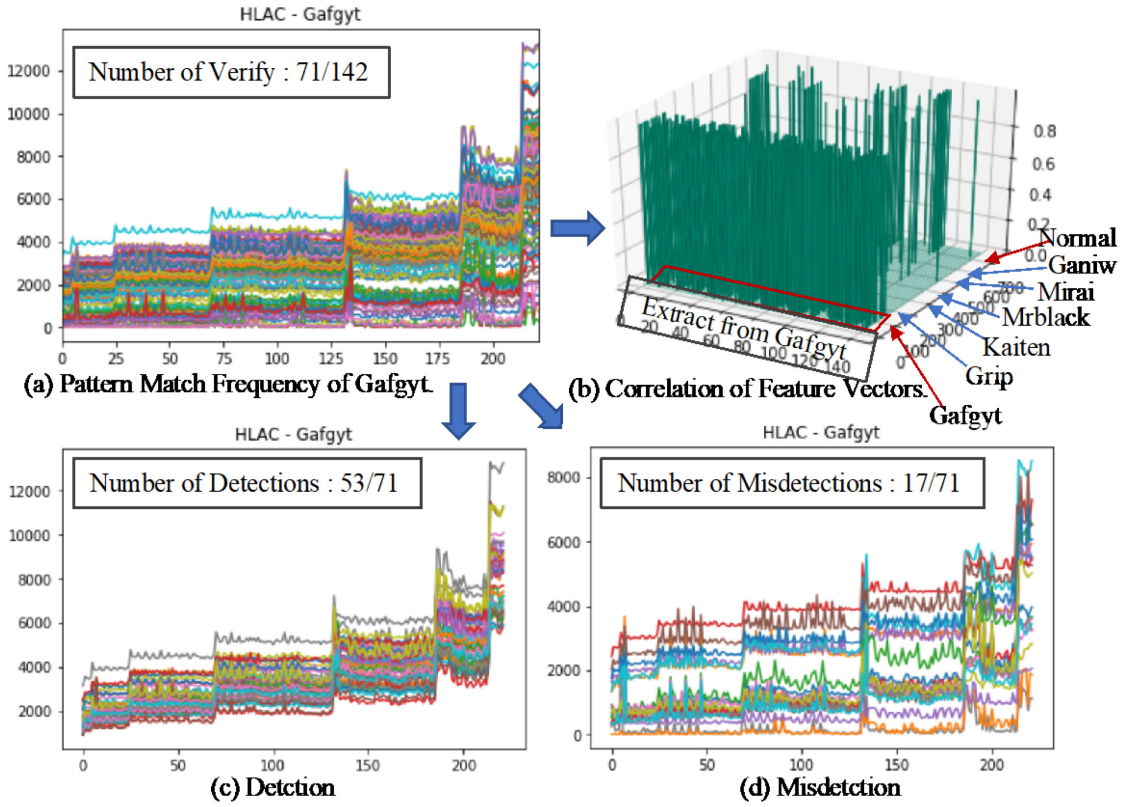



図 5.2 Malware Gafgyt ファミリの誤検知に関する考察

ところで，識別性能を可能な限り向上させることは，閾値解析部分のプログラムの複雑さを招く。適用システムが必要とする識別性能と計算コストとの兼ね合いの中で最適な追加プログラム処理量を決める必要があると考える。

これらとは異なるアプローチであるが，低い識別性能のファミリーに関して，Texture Image の領域分割手法(Segmentation)を導入し，領域毎のパワースペクトルを計算するように提案アルゴリズムを改良することで，性能向上も可能だと考えている。Texture Image の領域分割は，分割の境界を何処に，何にするのかが，難しい。文献[23]のボロノイダイアグラム(Voronoi Diagram)の理論を参考例として分割手法を導出可能であると考えている。具体的には，本研究のプロセスフロー(図 4.1)の Class2 処理に Voronoi Polygons による Texture Image の領域分割を融合させ，分割した領域毎に HLAC マスクパターンによるパターンマッチングを行い複数領域の Class3，Class4 処理を行うことで，より高精度の識別処理ができると考えている。さらに，このような同じアルゴリズムの並列処理プロセッサは，文献[22]の Celerity Chip が現行半導体技術での実現性を実証済みである。

## 5.2 難読化 Malware への対応

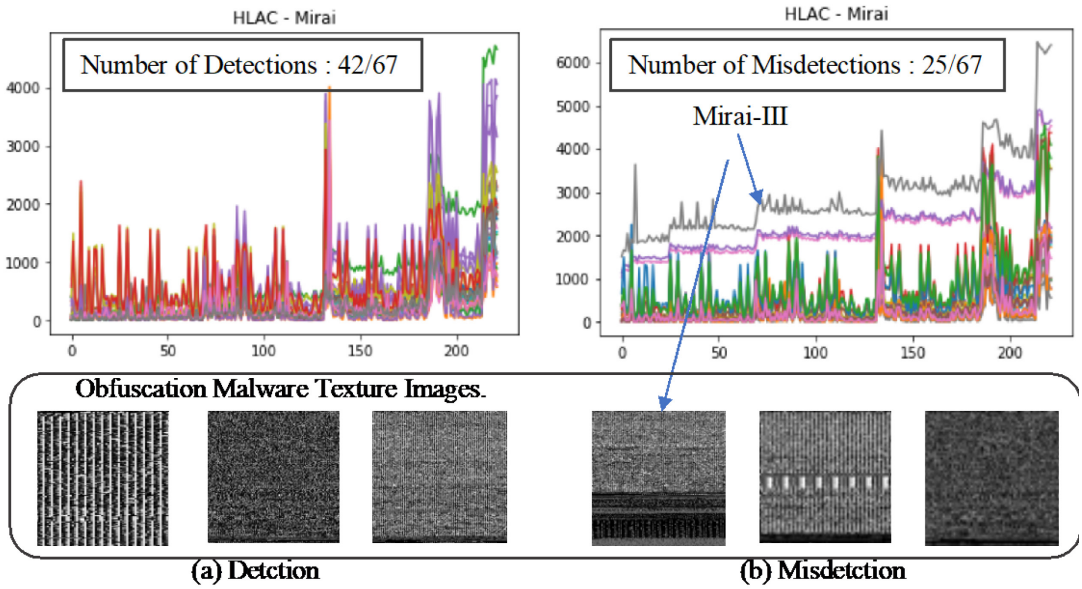
多くの Malware は，コード解析を妨害する目的でパッキングという難読化処理が施されている。しかし，本研究で適用した Malware の Texture Image 化による手法の最大の利点は，パッキングなどの難読化された Malware であってもアンパッキングを必要とせず解析が可能なことである。これは，文献[13]の既往研究において，難読化への弾力性として示されている。

本論文でも，本提案のアルゴリズムが，文献[13]同様に難読化への弾力性を持つことを考察する必要がある。

図 5.3 は，Mirai に関して，パッキングなどによる難読化処理を受けたサンプルの識別結果を表した図である。Mirai の識別率は，他のファミリーサンプルよりも大幅に低い。これは，Mirai のソースコードが公開され，高度な難読化手法を施した亜種が多数存在することと無関係ではない。図 5.3(a)は，正判定サンプルの波形トレンド図とその一部サンプル(難読化後)の Malware Texture Image で

あり，図 5.3(b)は，誤判定したサンプルの波形トレンド図とその一部サンプル(難読化後)の Malware Texture Image データである。これら図 5.3(a)と図 5.3(b)において，パターンマッチング後の波形トレンドを比較すると，誤判定されたサンプルの波形トレンドは，異なる波形トレンド (例えば Mirai-III と定義できる異種 Malware ファミリー)に変化しており，識別を逃れたことが分かる。文献[13]においてもパッキングによる難読化後サンプルは，同一のパッキング手法を用いた異なるファミリーと誤認識される場合があると指摘されている。

結論として，本提案のアルゴリズムは，難読化後サンプルであっても識別可能であるが，パッキングなどの難読化によるコード変更の複雑さの程度によっては，誤判定となるケースもあり課題が残る。さらに，この部分は，静的コード解析手法と並行して解析を行う必要があると考える。




Discussion of Malware Obfuscation (Mirai).

図 5.3 Malware Mirai ファミリの難読化

## 5.3　計算コスト比較

本研究の Malware 検出システムのスコープにおいて重要な要件は，Security Gateway のような IoT Edge システムへ搭載可能なサイズと消費電力，処理プログラム量，および処理時間である。さらに，ユースケースに適した識別性能を低コストで実現させることである。

Malware の解析には，動的コード解析手法と静的コード解析手法があり，動的コード解析手法は，隔離環境を OS 仮想化上に構築することから CPU パワーと OS 環境に高い要件を必要とする。そのため，CPU 環境コストの観点から Edge 搭載は難しい。また，静的コード解析は，厳密なコード解析を通じて高い識別性能を提供する。昨今はサーバ側の高性能 CPU や GPU などの AI 専用プロセッサによる豊富なコンピューティングリソース環境下で機械学習(ML)を活用し高精度な解析を行うことが可能であり，結果として高い識別率を提供している。しかし，Edge 側に高性能 CPU(例:8th Generation Intel Core Processor の Thermal Design Power: 28W~95W)[24]をインテグレートすることは，消費電力・冷却・サイズの観点から難しい。さらに，自律的に Malware 識別が可能な要件も重要視しており，Edge 側とサーバが連携することは，異なるスコープであると考えている。

本研究のアルゴリズムを動作させる Malware 検出システムのサイズと消費電力，実行性能，プログラムサイズは，PM Accelerator チップコアのダイサイズが，65nm プロセス適用ケースにおいて，85 平方 mm, 256×256 Pixels の Matching Time は，~5μsec@100MHz オペレーションサイクル，識別処理に要する電力は，6μJ(電源電圧:1.2V, 演算時電流:1.2A@100MHz)，約 1.4W である。この数字は，NVIDIA 社の Edge 搭載 GPU Jetson Nano[25]と比較して 1/3~1/7 程度である。さらに，主記憶領域に格納するプログラムサイズは，約 2Mbyte(Matrix 領域含む)，識別解析のプログラム処理時間が約 5.8msec と見積もることができ，導出された数値は，十分に Edge 側に搭載可能な諸元だと考える。

ところで，256×256 Pixels 構成のサイズ拡大は，マッチング時間短縮に有利である。しかし，PM Accelerator のパターンマッチング演算器は，1Pixel 当たりの

回路数が重く，集積度とチップサイズのコスト制約が厳しい。故に本研究では Texture Image を 256×256 Pixels と固定した。何故なら，実験では 1Pixel 単位(Bin)で照合処理することから HLAC マスクパターンの特徴である加法性と位置不変性により，サイズ拡大をしても殆ど等価な局所的特徴と大局的特徴のパワースペクトル抽出が可能である。さらに，識別処理の特徴ベクトル相関で特異値分解によるノイズ除去により，影響の無いレベルに低減できると考えたからである。

## 5.4 異なるコンパイラと ISA

Malware のコードは，生成したコンパイラやプロセッサの ISA(Instruction Set Architecture)が異なればコード自体も異なる。これらの差異を本論文の提案手法が吸収でき，同等の識別性能を発揮できるかどうか考察する必要がある。

図 5.4 は，コンピューティングスタックを示したものである。コンパイラは，機械語生成処理の過程で，実行コードに必要なライブラリと API などの Runtime コードを組み込む。Malware は，レジストリ設定ファイルの修正・変更，他のプロセスや OS コマンドを実行させるコードインジェクション，通信機能などの API コールなどのため，Runtime コードやライブラリを多数取り込む。さらに，これら Runtime コードやライブラリは，処理速度やコードサイズの効率を上げるためにプロセッサに最適化(Micro Architecture の設計性能を引き出す)された機械語で直接コーディングされており，プロセッサ開発元が共通ライブラリとして提供しているケースが多い。リンケージなどのツール類は，コード実行効率化のために，これらの提供ライブラリコードを共通利用する。このような背景から，異なるコンパイラでコンパイルした Malware であっても実行コード間の類似度は高く，本提案のアルゴリズム手法への影響は少ないと推定する。

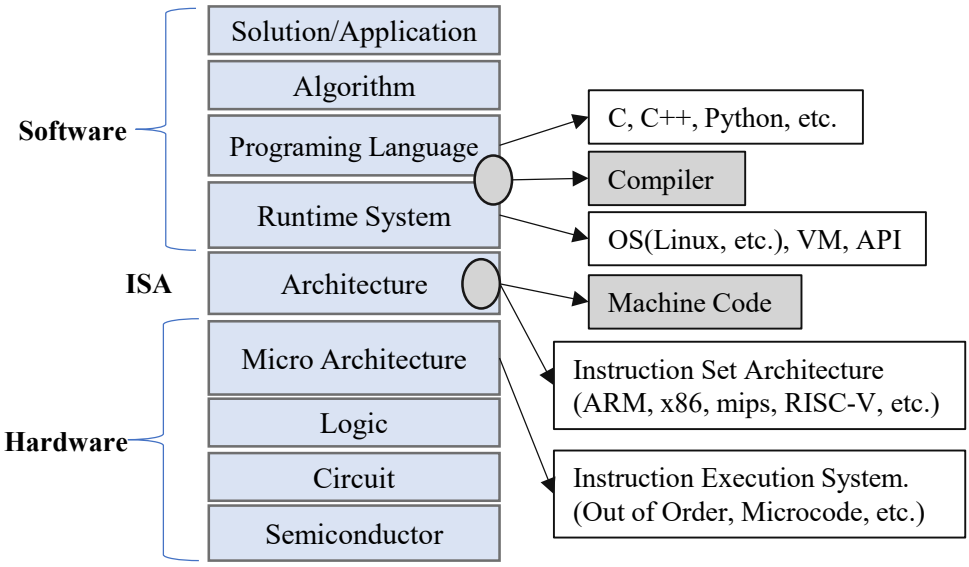
異なるプロセッサ ISA の課題[1]に関しては，対象とする Edge へ内蔵されたプロセッサ ISA を標的とした Malware ファミリー群から特徴抽出したデータを学習し，識別処理を実行することで解決できる。特に IoT Edge システムに使用されるプロセッサは，ARM 系を基本とした拡張アーキテクチャが多数を占めており，ARM 対象の Malware コードをカバーするように Malware サンプル評価結果

から特徴ベクトルを調整したシステムを用いれば良いと考える。

しかし，今後は，RISC-V のような新しい ISA のプロセッサの台頭も考慮すると，ISA の差異をどのように吸収すべきか課題が残る。この課題解決の糸口として押さえておきたい論理は，現在のコンピュータの基本的アーキテクチャはフォンノイマン型の Load/Store アーキテクチャを踏襲していることである(図 5.5)。これは，機械語実行レベルから見えるリソースが，記憶機構(Memory)，演算機構(Arithmetic Logic Unit)，命令やデータを取り出し格納を行う記憶制御機構(Memory Access)，命令レジスタと演算レジスタ(Register File)と割り込み関係の特権命令に使う(Flag Register)などであり，異なる ISA 間であっても基本的動作を行う機械語命令のコード列の波形トレンドは，類似していると推定する。

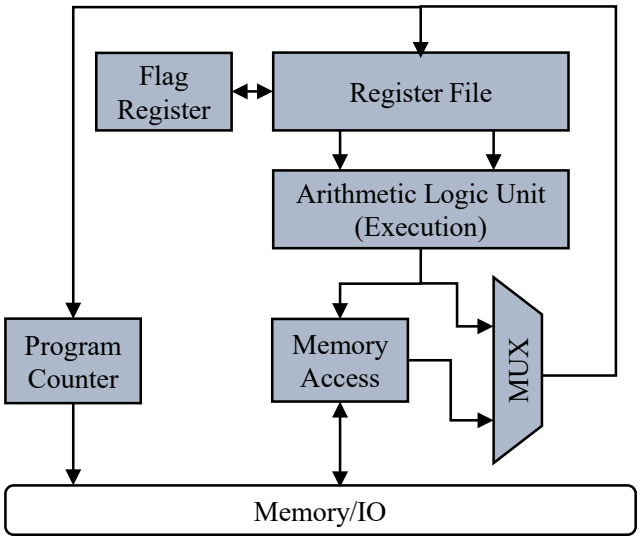
これは，Malware Texture Image を構造的に抽象化したパワースペクトルの解析を行う本アルゴリズムは，共通機械語命令列の累積(局所的特徴)を計算するだけで無く，Malware コード全体が持つ大局的特徴をヒストグラムとして表現し，それが持つパワースペクトルの波形トレンドを特徴量として解析することができるからだと考えている。

但し，これら仮説に関しては，静的コード解析手法などの厳密なコード解析との実験比較により実証できることから，本研究においては，課題として残る。




The Concept of Computing System Stack.

図 5.4 一般的なコンピューティングスタック構成




Fundamentally CPU Function Diagram.

図 5.5 基本的プロセッサの機能構成

## 5.5　車載実装

図 5.6 は，クルマなどの Security Gateway に適用が進む TEE(Trusted Execution Environment)という暗号鍵生成・管理・認証機能を備えた Edge デバイスである。通常の動作領域(Normal World）からセキュリティ保護領域(Secure World)へのアクセスが不可能な構成となっており，クリティカルデータの格納や処理を提供できる。このような Cyber-Security に優れたシステムにおいても，システムの立ち上がりシーケンスや通常稼働中の攻撃検知など，定常的にセキュリティチェックを実行する機能実装が求められている。特に NHTSA(米国運輸省道路交通安全局)から表 5.1 に示す，クルマの Cyber-Security 保護要求(Fundamental Vehicle Cybersecurity Protections)という指令が発効されており，クルマの動作にクリティカルなファームウェアの不正更新に対する Cyber-Security 対策(監視と記録)を強く求めている[26]。

これらの必要性は，メンテナンスアップデートなどのライフサイクルに渡り，ファームウェアなどへの Malware 混入を完全に担保することが難しいからである。さらに，クルマの Cyber-Security 要件として，これら TEE 機構のセキュアブート時間は，500msec 以内に完結することが求められており，ブートシーケンスに組み入れられる Malware の検知処理もリアルタイムに近い性能が必要である。

本研究の Malware 検出システムは，Security Gateway に求められる処理スピード，サイズ，消費電力を備えることから，図 5.6 の TEE 機構を支えるメインプロセッサに連結するコプロセッサとして活用できると考える。

## 5.6　5 章のまとめ

本章では，提案手法の有効性と課題に関して，重要と判断した①検知・誤検知の性能，②難読化ファイルへのアルゴリズム弾力性，③コンピュータの命令セット(ISA)やコンパイラの差異，④Edge 組み込みの計算コスト比較，⑤期待される

クルマの実装領域に関して説明を行った。

結果として，解決すべき課題や実証すべき仮説は残るが，実フィールドにおける Edge 実装を想定した Malware 識別アルゴリズムにおいては，本研究の提案は非常にポテンシャルが高いと考える。

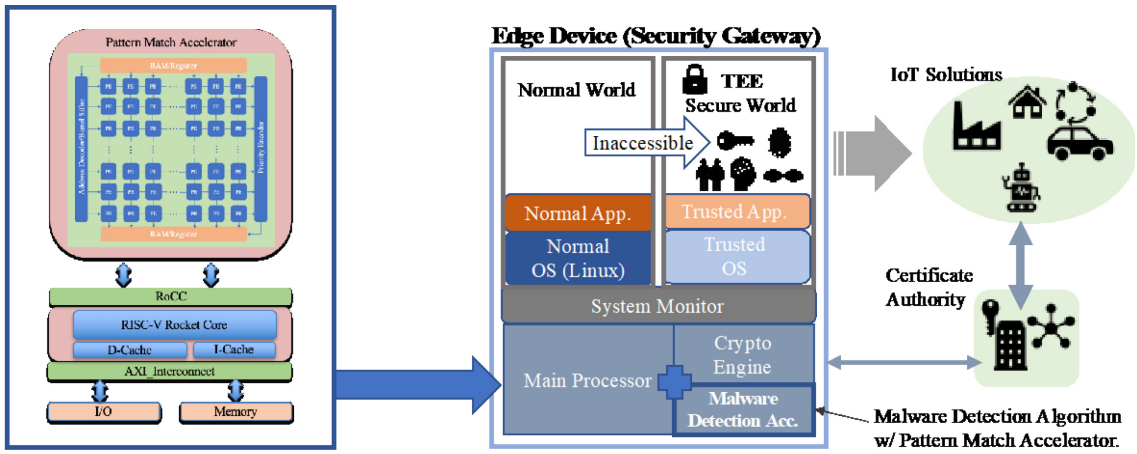



図 5.6　Cyber-Security 向け Edge Computing.

表 5.1 米国運輸省道路交通安全局発行のクルマのセキュリティ要件[26]

| セキュリティ対策技術 | NHTSA セキュリティ保護要求 Fundamental Vehicle Cybersecurity Protections |
|---|---|
| アクセスの認証・鍵管理 | ✓ 車両のデバッグインターフェース/データに対するアクセス制限・排除<br>✓ 車両のメンテナンス/診断操作に対するアクセス制御<br>✓ ファームウェアの分析・改竄排除のためのアクセス制御<br>✓ ネットワークポート/プロトコル/サービスの制限・削除<br>✓ 車両の内部・外部におけるセキュアなワイヤレス通信制御 |
| 機密性確保 | ✓ 暗号化によるファームウェアの保護とセキュアブートアップ<br>✓ 鍵・証明書付き暗号データによる車両とサーバ間の通信 |
| 完全性確保 | ✓ デジタル署名を用いた正当性のあるファームウェア更新<br>✓ 一般通信経路と分離した車両内メッセージ通信制御の導入<br>✓ 車両ユニークに紐付いた認証用暗号鍵の管理 |
| 可用性確保 | ✓ 車両アーキテクチャ設計におけるセグメンテーション分離技術適用 |
| 監視・記録 | ✓ サイバー攻撃を記録するログの仕組み(トレーサビリティ確保) |

# 第6章　結言

## 6.1　本研究で得られた成果

本研究で提案した Malware 識別アルゴリズムを用い, 実験システムによる有効性検証により，以下の 3 点の研究成果を得た。

(1) HLAC 理論を適用したマスクパターンと Pattern Matching Accelerator(PM Accelerator) を用いた構造的解析手法のアルゴリズムにより画像化 Malware(Texture Image)から Malware 固有のパワースペクトル/時系列データの抽出ができることを確認した。

(2) Malware 固有のパワースペクトル/時系列データをモード信頼性評価基準(MAC: Modal Assurance Criterion)を用いて識別処理を行った結果，平均で 8 割近い精度で Malware ファイルを検出可能であると確認できた。但し，難読化やコンパイラと ISA の差異など，識別性能低下につながる課題解決に向けた，さらなる改良が必要である。

(3) PM Accelerator の既往研究結果[20][21]から導出した計算コスト概算によると，本研究のアルゴリズムを適用する PM Accelerator は，一つの Malware ファイルの識別処理を，1.1msec以下(~5μsec×221 マスクパターン)で実行する。汎用プロセッサ単体を用いたパターンマッチングに比較し，約 $10^5$ 倍の高速処理が可能である。この処理時間の短縮により，電力使用量の低減が期待できる。

以上，3 点の効果により，高い Malware 検出能力を持った低計算コスト(処理時間，消費電力，システムサイズ)の車載 IoT Edge Computing の実現が可能である。

## 6.2　今後に残された問題

本研究を通じて，非常に重要な幾つかの問題が残された。これらは，将来の有益な研究テーマを生み出すと考えており，それら問題に関して下記にまとめる。

(1) 検出率の向上問題

検出率の向上策に関しては，識別性能の劣る特定 Malware ファミリー群に対して複数の波形トレンドを新しい異種ファミリーと再定義することで，特徴ベクトルの再構築を行う必要がある。さらに，リアルタイム処理性能を損なわない範囲で，ダイナミックに閾値を設定し直す追加プログラム処理が必要であると考えている。誤検知に関しては，誤判定したサンプルの波形トレンドと正判定の波形トレンドから識別用の特徴ベクトル群を再構築する Malware 識別アルゴリズムの追加検討を進めていく。

これらとは異なるアプローチであるが，低い識別性能のファミリーに関して，ボロノイダイアグラム(Voronoi Diagram)の理論を応用し，Texture Image の領域分割(Segmentation)[23]を導入することで，領域毎のパワースペクトルを計算するよう提案アルゴリズムを改良することも並行して検討していく。

(2) コンピュータアーキテクチャ問題(異なる命令セット)

現在のコンピュータアーキテクチャは，基本的に IBM/360 の命令セットのエンハンスであり，50 年間大きなジャンプは無い。特に RISC プロセッサの登場以降，ISA(Instruction Set Architecture)のバリエーションに大きな特徴差異は少なくなったと考えている。その結果，異なる ISA 間であっても基本的動作を行う機械語命令のコード列の波形トレンドは，類似していると推定する。これは，提案アルゴリズムに汎用性を持たせる意味で大きなファクターである。今後，検証の仕方に新たなアイデアを持ち込み ISA の差異に左右されない提案アルゴリズムの汎用化研究に取り組みたい。

(3) 照会ベンダーからのフィードバックへの対応

本研究の成果は，独自動車 OEM や国内自動車部品メーカ，国内インフラ

会社など，複数の企業から注目され，プレゼンテーションと Q&A を求められ対応した。

これらのベンダーからのフィードバックで，注目に値する課題として，未知の Malware への対応力，プロセス型 Malware への対応，Malware 以外の分野への応用，例えば，異常検知やプログラムバグ検出，特定のコード，特定のプログラム開発者の癖などを見つける技術への応用など，非常に興味深い応用研究要求が出ている。今後は，これら分野への提案手法の発展も検討していく予定である。

# 付録

## A.1 Connected CAR

近年，クルマとモバイルネットワークがつながり，新たな価値やビジネスを創出する Connected Car が注目されている。Connected Car というシステムは，複雑な Mobility 社会を構成する上で必須な技術として認識されており，これら多様な V2X システムを組み合わせたシステムオブシステムズ(System of Systems)は，データを収集・活用することで Society5.0 が目指す高効率で持続可能な社会実現へ向けた重要なインフラシステムであることが理解できる(図 A.1.1)。

Connected Car と Cloud 側とのつながりを表現したシステム概要を図 A.1.2 に示す[27]。クルマはモバイルネットワークを介して，利用目的に応じた様々な Cloud と接続するパスを持つことが理解できる。例えば，クルマのナビゲーションシステムなど IVI-Infotainment(In Vehicle Information + Entertainment)領域の接続パスは，地図データやコンテンツデータをポイントクラウドと呼ぶ Cloud へアクセスする。OEM Oriented 領域では，クルマのエンジンなど運転に直接結び付く制御を行う車載コンピュータ(ECU)と Cloud 間がデータを送受信し，ファームウェアのアップデート(Secure-OTA)やデータセンシングを行うために独自のパスを持つ。最近では，コネクテッド技術の進展に伴い，クルマが IoT の Edge として進化していく必然性が認識されている。これは，自動運転(Autonomous Drive)や先進的運転支援機構(ADAS)では，必要な System of Systems である[28]。

具体的な例として，文献[29]で Kiyama らは，Telematics システムにおいて，電気自動車(EV: Electric Vehicle)の電池残量をリアルタイムに計測し，Cloud 側で電池残量に応じた航続距離と最適な充電スタンド(Charging Station)のマッチングを行い，ナビゲーションにより最適ルートを指示する手法を提案している。実現したシステムは，2010 年よりビジネス運用に入っている。このようなコネクテッドシステムは，EV 電池消費量と充電スタンドの関係をマッチングさせ，都市全体の電気エネルギー資源を効率的に運用することが可能になる。

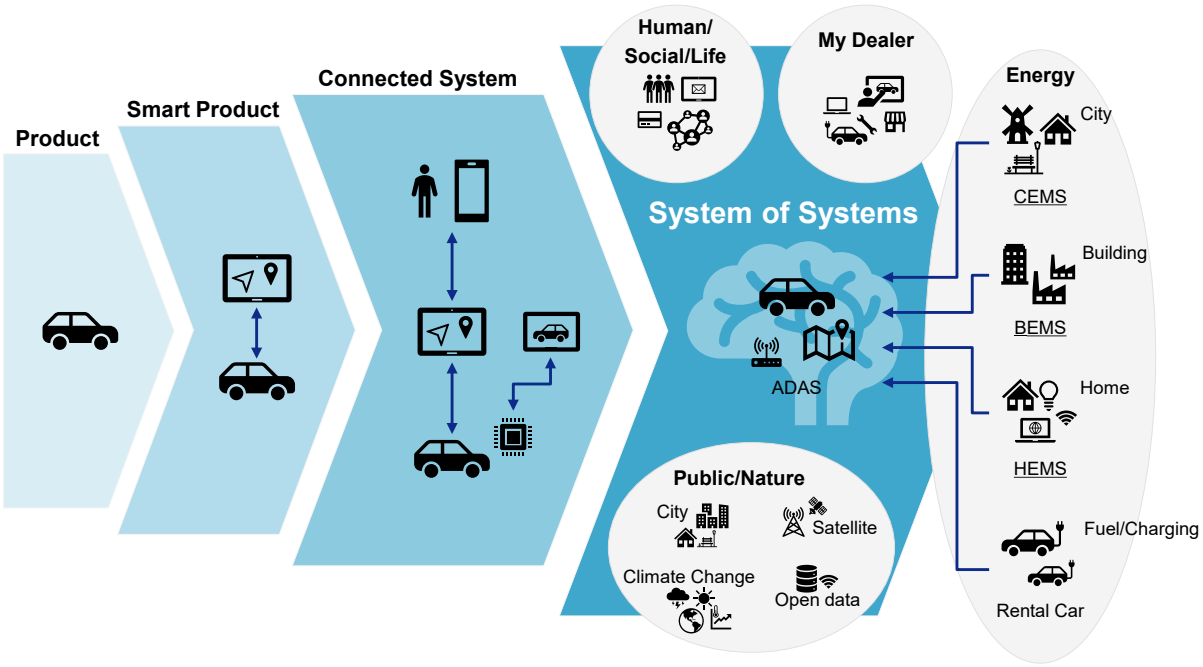



図 A.1.1 Connected の進展による System of Systems

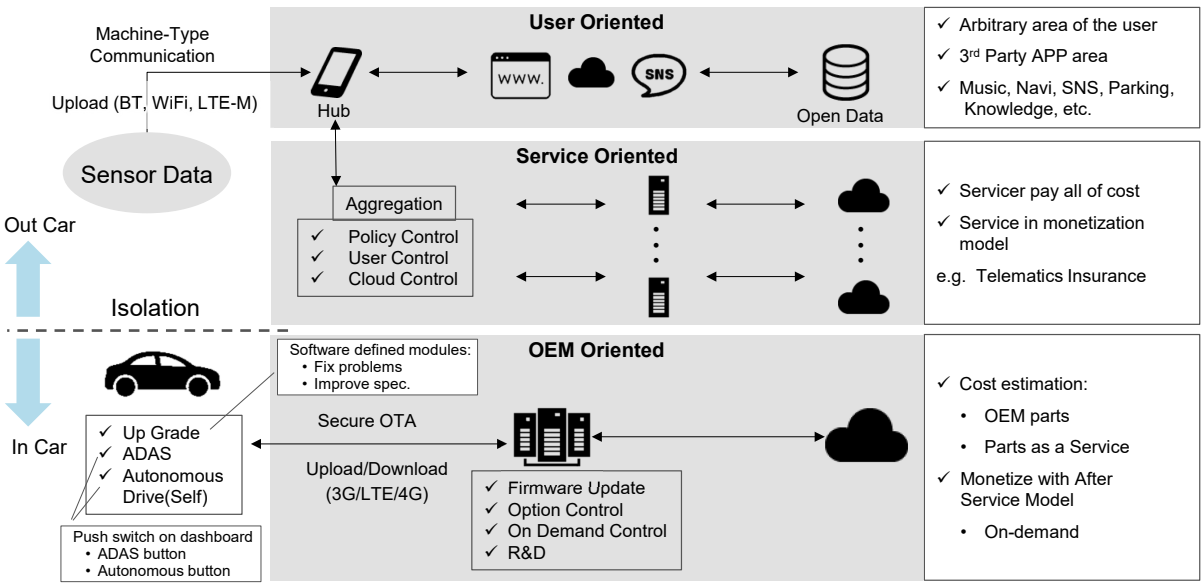



図 A.1.2 Connected Car の仕組みとデータ管理

## A.2 Trusted Execution Environment

付録 A.2 では，IoT のセキュリティ脅威と課題について，有効な解決策の一つであるハードウェアセキュリティモジュール TEE(Trusted Execution Environment)の概要を解説する。TEE を実装することで，Edge デバイスが攻撃されることを前提に考えたセキュリティ対策が可能となる。

TEE は，ソフトウェアとハードウェアが協調することによって，アプリケーションの安全な実行環境を実現するための技術仕様であり，ソフトウェアレベルの脅威を効率的に防ぐことができる。技術仕様は，GlobalPlatform という団体(Apple, Samsung, Huawei, NTT, Toshiba などが参画)により，各種 API(Application Programming Interface)や，ユーザーインターフェースである TUI(Trusted User Interface)などの仕様策定が進められている[30]。安全な実行環境を実現する同様の技術として，他に SE(Secure Element)などがある。具体的実装としては，英国 ARM 社の Trust Zone テクノロジーや RISC-V プロセッサをベースとした UCB の Keystone がある。

図 A.2.1 に概要を示す。この TEE 技術は，Normal World と呼ぶ，通常アプリケーション実行環境と，System Monitor により分離された Secure World と呼ぶ安全な実行環境を実現する。Normal World では，Rich OS と呼ぶ通常の Linux などの汎用 OS を稼動させる。他方，TEE 環境である Secure World では，生体認証データ，鍵・証明書など，特にセキュリティが要求される処理を行う Trusted OS を稼動させる。ハイパーバイザー機能をサポートするプロセッサにより Normal World から Secure World の環境にはアクセスできないよう制御される。これらのハードウェアアシスト機能により Secure World で実行されるクリティカル処理やデータを保護している。Malware などが，Normal World で稼動する汎用 OS の管理者権限を奪取したとしても，Normal World から Secure World に侵入することはできないことから，高度の保護すべきデータの侵害は阻止できる。これらの

機能により，デバイスの所有者も容易にアクセスできない仕掛けとしているため，Cyber-Security に一定の防御効果が期待できる。

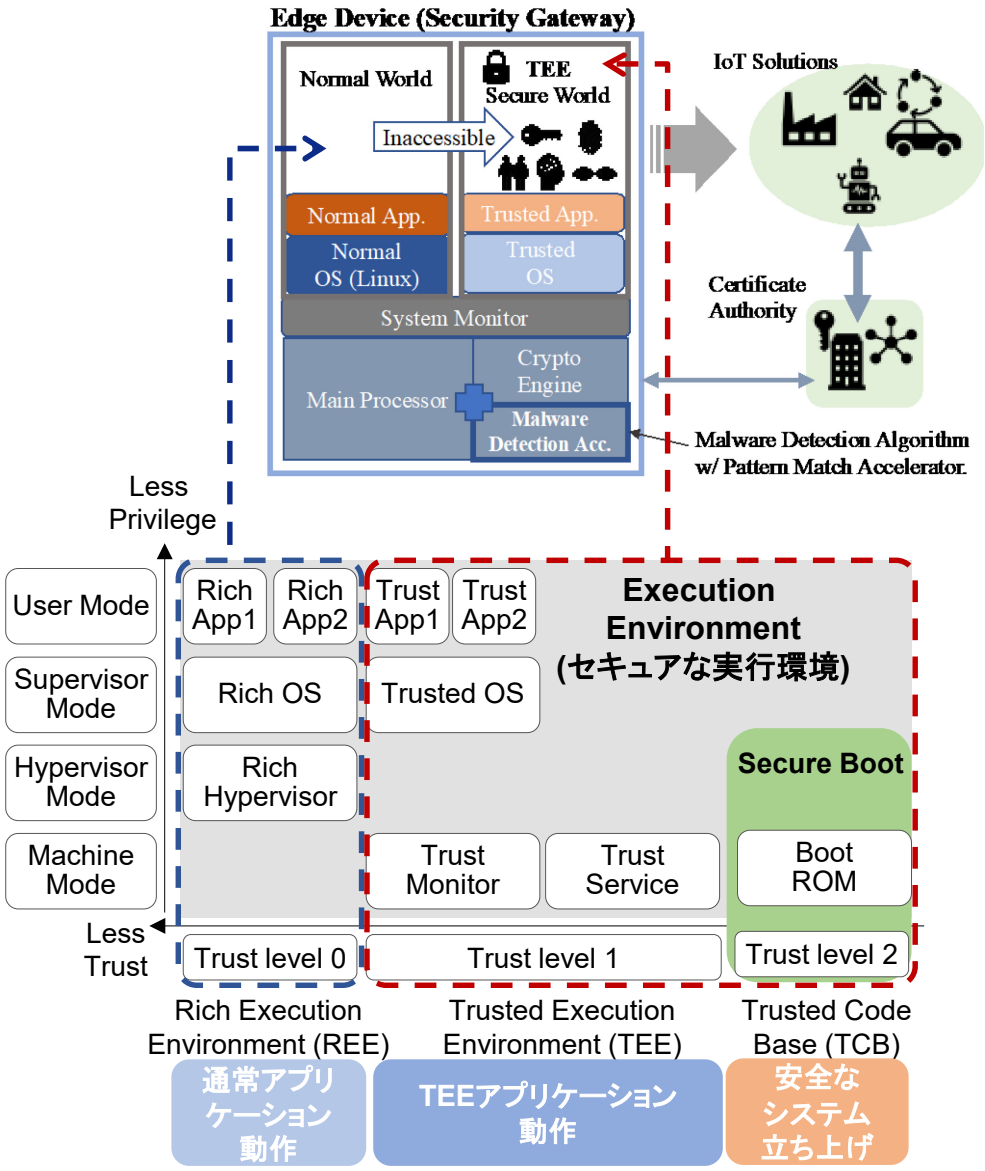



図 A.2.1 Trusted Execution Environment の概要

# 謝　　辞

本研究は，日本大学理工学部応用情報工学科において，泉 隆 特任教授，細野 裕行 教授，関 弘翔 助手の指導の下に著者が行ったものである。近年，ICT の進化により，様々なデバイスとクラウドがつながる IoT が進展しており，これら Edge 環境の Cyber-Security 性を担保する重要な技術研究を進められたことは，ひとえに先生方のご指導の賜である。ここに謹んで感謝申し上げます。

本研究のために Malware サンプルをご提供いただいた横浜国立大学理工学部情報・物理セキュリティ研究拠点の吉岡准教授，ならびに日本大学理工学部応用情報工学科の小寺君，三田君にお礼を申し上げます。特に，三田康一郎君は，本研究と同じエミュレータを用い，ピアソン積率相関による Malware 分類実験から高い分別率を達成しており，アルゴリズムや識別性能に関して，貴重なディスカッション機会を持つことができました。

PM Accelerator の構造と性能に関して，ご指導いただいた電気通信大学大学院情報理工学研究科の範 公可(Cong-Kha Pham)教授と株式会社 AOT 代表取締役の井上克己氏に感謝申し上げます。尚，当該 PM Accelerator は，2019 年度 NEDO(国立研究開発法人新エネルギー・産業技術総合開発機構)事業「画像集合演算プロセッサ(2D-SOP)による高度画像認識基盤の開発」として採択されました。

パワースペクトル/時系列データ解析に関して，メカニカル振動論の観点からご指導いただいた株式会社 Z2One 代表取締役の辻 宏実智氏に感謝申し上げます。

クルマの Cyber-Security の技術動向や TEE アーキテクチャと Malware 検出アクセラレータの融合などに関して，著者が参画する 2018 年度 NEDO 事業「セキュアオープンアーキテクチャ基盤技術とその AI エッジ応用研究開発」の成果を参照しました。事業参画メンバーとのディスカッション機会に感謝します。

# 文　献


[1] National Institute of Information and Communications Technology NICT NEWS : “A State-of-the-Art R&D on Cybersecurity”, Vol.472, No.6 (2018-11)
国立研究開発法人情報通信研究機構 NICT NEWS : 「サイバーセキュリティの研究開発最前線」, Vol.472, No.6 (2018-11)

[2] Y. Taniwaki : “Cybersecurity in the Age of Digital Economy – Toward Establishing a Foundation for Digital Transformation -: 4. Spreading IoT Devices and Cybersecurity Policy”, *IPSJ Magazine,* Vol.59, No.12, pp.1090-1094 (2018-11) (in Japanese)
谷脇康彦 :「ディジタルエコノミー時代のサイバーセキュリティ –ディジタルトランスフォーメーション促進の基盤確立に向けて- : 4. IoT 機器の普及とサイバーセキュリティ政策」, 情報処理学会誌, Vol.59, No.12, pp.1090-1094 (2018-11)

[3] 一般社団法人 日本クラウドセキュリティアライアンス(CSA ジャパン) IoT ワーキンググループ :「IoT へのサイバー攻撃仮想ストーリー集」, A-7 自動車システムからの情報混乱,Vol.1, pp.23-26 (2017-8)
http://www.cloudsecurityalliance.jp/IoT_WG.html

[4] National Highway Traffic Safety Administration (NHTSA) : “NHTSA and Vehicle Cybersecurity”, https://www.nhtsa.gov/technology-innovation/vehicle-cybersecurity, (2018-1)

[5] Apple Inc. : “Apple T2 Security Chip Security Overview”, https://www.apple.com/mac/docs/Apple_T2_Security_Chip_Overview.pdf, (2018-10)

[6] Tom Conte : “IEEE Rebooting Computing Initiative & International Roadmap of Devices and Systems”, *2015 IEEE Computing Society President, Co-Chair, IEEE Rebooting Computing Initiative*, Schools of CS & ECE, Georgia Institute of Technology.

[7] David Patterson : “50 Years of computer architecture: From the mainframe CPU to the domain-specific tpu and the open RISC-V instruction set”, *2018 IEEE International Solid-State Circuits Conference (ISSCC)*, (2018-2)

[8] Karl Rupp : “42 Years of Microprocessor Trend Data”, GitHub, 2018.

https://www.karlrupp.net/2018/02/42-years-of-microprocessor-trend-data/

[9] M. Kashiyama : “Explanation of Data Bus Technologies for High Speed Processing”, *IPSJ Magazine*, Vol.38, No.6, pp.457-459 (1997-6) (in Japanese)
柏山正守 :「「高速プロセッシングデータバス技術」の編集にあたって」, 情報処理学会誌, Vol.38, No.6, pp.457-459 (1997-6)

[10] M. Iwamura, M. Itoh, and Y. Muraoka : “Automatic Malware Classification System Based on Similarity of Machine Code Instructions” , *IPSJ Journal*, Vol.51, No.9, pp.1622-1632 (2010-9) (in Japanese)
岩村誠 ・ 伊藤光恭 ・ 岡村洋一 :「機械語命令列の類似性に基づく自動マルウェア分類システム」, 情報処理学会論文誌, Vol.51, No.9, pp.1622-1632 (2010-9)

[11] K. Tanaka, and A. Goto : “Study of Countermeasures based on the Characteristic of Recent Attack Code*”, IEICE, SCIS2014*, The 31st Symposium on Cryptography and Information Security Kagoshima, Japan, (2014-1) (in Japanese)
田中恭之・後藤厚宏 :「攻撃コードの特徴からみた対策の検討」, 電子情報通信学会, 2014 年 暗号と情報セキュリティシンポジウム, (2014-1)

[12] L. Nataraj, S. Karthikeyan, G. Jacob, and B. S. Manjunath : “Malware Images: Visualization and Automatic Classification”, *VizSec’ 11 Proceeding of the 8th International Symposium on Visualization for Cyber Security,* Article No.4, (2011-7)

[13] L. Nataraj, V. Yegneswaran, P. Porras, and J. Zhang : “A Comparative Assessment of Malware Classification using Binary Texture Analysis and Dynamic Analysis”, *Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence,* USA, (2011-10)

[14] A. Olivia, and A. Torralba : “Modeling the shape of a scene: a holistic representation of the spatial envelope”, *Intl. Journal of Computer Vision*, Vol.42, No.3, pp.145-175 (2001-5)

[15] T. Suzuki, T. Kasama, and N. Miyaho : “Study of Malware Classification Method Utilizing Image of Binary Data”, *The Institute of Electronics, Information and Communication Engineers (IEICS)* ,Tokyo Student Branch Conference D-19 paper 205, (2015-1) (in Japanese)
鈴木貴之・笠間貴弘・宮保憲治 :「バイナリーデータの画像化を活用したマ

ルウェア分類の検討」,平成 27 年度電子情報通信学会東京支部学生会研究発表会, D-19 講演 205（2015)

[16] F. Tomita, Y. Shirai, and S. Tsuji : “Texture Analysis*”, IPSJ Magazine*, Vol.19, No.2, pp.173-182 (1978-2) (in Japanese)
富田文明・白井良明・辻 三郎 :「テクスチャ解析」,情報処理学会誌, Vol.19, No.2, pp.173-182 (1978-2)

[17] T. Toyoda, and O. Hasegawa :“Feature Extraction for Texture Classification Using Mask Patterns”, *IPSJ SIG Technical Reports*, Vol.2004, No.91, pp.77-84 (2004-9) (in Japanese)
豊田崇弘・長谷川修 :「テクスチャ識別のためのマスクパターンによる特徴抽出法」,情報処理学会研究報告コンピュータビジョンとイメージメディア, Vol.2004, No.91, pp.77-84 (2004-9)

[18] T. Toyoda, O. Hasegawa : “Extension of Higher Order Local Autocorrelation Features”, *Imaging & Visual Computing The Journal of the Institute of Image Electronics Engineers of Japan*, Vol.34, No.4, pp.390-397 (2005-7) (in Japanese)
豊田崇弘・長谷川修 :「高次局所自己相関特徴の拡張」, 画像電子学会誌論文, Vol.34, No.4, pp.390-397 (2005-7)

[19] K. Inoue, and C.-K. Pham : “The Memorism Processor: Towards a Memory-Based Artificially Intelligence Complementing the von Neumann Architecture”, *SICE Journal of Control, Measurement, and System Integration*, Vol.10, No.6, pp.544-550 (2017-11)

[20] K. Inoue, D.-H. Le, M. Sowa, and C.-K. Pham : “Set Operating Processor(SOP) : Application for Image recognition”, *The Institute of Electronics, Information and Communication Engineers, IEICE Technical Report*, Vol.113, No.236, pp.35-40 (2013-10) (in Japanese)
井上克己・レ ドゥクフン・曽和将容・範公可 :「集合演算プロセッサー(SOP) – 画像認識への応用ー」, 電子情報通信学会技術研究報告, Vol. 113, No. 236, pp.35-40 (2013-10)

[21] D.-H. Le, T.-B.-T. Cao, K. Inoue, and C.-K. Pham : “A CAM-Based Information Detection Hardware System for Fast Image Matching on FPGA”, *IEICE Trans Electron.,* Vol. E97-C, No.1, pp.65-76 (2014-1)

[22] T. Ajay, K. Al-Hawai, A. Amarnath, S. Dai, S. Davison, P. Gao, G. Liu, A. Lotfi, J. Puscar, A. Rao, A. Rovinski, L. Salem, N. Sun, C. Torng, L. Vega, B. Veluri, X.

Wang, S. Xie, C. Zhao, R. Zhao, C. Batten, R. G. Dreslinski, I. Galton, R. K. Gupta, P. P. Mercier, M. Srivastava, M. B. Taylor, and Z. Zhang : “Celerity: An Open Source RISC-V Tiered Accelerator Fabric”, *IEEE Hot Chips: A Symposium on High Performance Chips 2017* (Hot Chips29), (2017-8)

[23] M. Tuceryan, A. K. Jain, and Y. Lee : “Texture Segmentation using Voronoi Polygons”, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.94-99 (1988-7)

[24] Intel Corporation : “8th Gen Intel Core Processor Families”, Datasheet, Volume 1 of 2, (2018-4)

[25] NVIDIA Corporation : “Nvidia Jetson Nano Technical Specifications”, 2019. https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/

[26] National Highway Traffic Safety Administration. (NHTSA) : “Cybersecurity best practices for modern vehicles.” (Report No. DOT HS 812 333), pp.17-20, Washington, DC: Author, (2017-12)

[27] M. Kashiyama : “Innovative Data Management and Utilization for Connected Car” , TU Automotive JAPAN, (2015-10)

[28] 柏山正守：「シリコンバレーがトレンドを創り世界がアジャストする」, JAISA 日本自動認識システム協会 自動認識セミナー, (2014-9)

[29] N. Kiyama, Y. Kobayashi, H. Aoshima, K. Shirai, and M. Kashiyama : “A Route Search Method for Electric Vehicles in Consideration of Cruising Range and Charging Time”, *IPSJ Journal*, Vol.54, No.1, pp.156-165 (2013-1) (in Japanese)
木山　昇・小林雄一・青島弘和・白井啓介・柏山正守：「航続可能距離と充電時間を考慮した電気自動車向けルート探索手法」, 情報処理学会論文誌, Vol.54, No.1, pp.156-165 (2013-1)

[30] GlobalPlatform Technology : “TEE Management Framework including ASN.1 Profile Version 1.0.1”, (2019-5)

# 著者発表論文

## （研究業績）

■ 学術論文誌論文 (査読付き)

1. Masato Iwabuchi, Masami Usami, Masamori Kashiyama, Takashi Oomori, Shigeharu Murata, Toshiro Hiramoto, Takashi Hashimoto, and Yasuhiro Nakagima, “A 1.5-ns Cycle-Time 18-kb Pseudo-Dual-Port RAM with 9K Logic ates,” *IEEE Journal of Solid-State Circuits*, vol.29, Issue4, pp.419-425, April 1994.
2. Katsuyuki Umezawa, Seiichi Susaki, Masamori Kashiyama and Satoru Tezuka, “A Proposal of User Authentication Infrastructure for Next-Generation Telematics,” *IEEJ Transactions on Electrical and Electronic Engineering (TEEE)*, vol.5, Issue 4, pp.439-449, Jun. 2010.
3. 木山昇，小林雄一，青島弘和，白井啓介，柏山正守，“航続可能距離と充電時間を考慮した電気自動車向けルート探索手法,”，情報処理学会論文誌, Vol.54, No.1, pp.156-165, Jan. 2013.
4. 柏山正守，柏山礼興，関弘翔，細野裕行，“Pattern Match Accelerator を用いた IoT Edge 向け Cyber-Security の研究, ”，電気学会論文誌 D (産業応用部門誌)， *IEEJ Transactions on Industry Applications*, Vol.140, No.1, pp.15-28, Jan. 2020.

■ 国際学会論文 (査読付き)

1. Masami Usami, Masato Iwabuchi, Masamori Kashiyama, Takashi Oomori, Shigeharu Murata, Toshiro Hiramoto, Takashi Hashimoto, Yasuhiro Nakajima, “A 1.5ns cycle-time 18kb pseudo-dual-port RAM,” *IEEE Symposium 1993 on VLSI Circuits*, pp.109-110, May, 1993.
2. Hiroaki Fujii, Yoshiko Yasuda, Hideya Akashi, Yasuhiro Inagami, Makoto Koga, Osamu Ishihara, Masamori Kashiyama, Hideo Wada, and Tsutomu Sumimoto, “Architecture and Performance of the Hitachi SR2201 Massively Parallel Processor System,” *IEEE 11$^{th}$ International Parallel Processing Symposium 97(IPPS’97)*, pp.233-241, April, 1997.
3. Katsuyuki Umezawa, Seiichi Susaki, Masamori Kashiyama, and Satoru Tezuka, “A Studay on an Authentication Infrastructure between Terminal and ASP for

Next Generation Telematic," *Intelligent Transport System Telecommunications (ITST2008)*. Pp.67-71, Oct. 2008.

4. Katsuyuki Umezawa, Seiichi Susaki, Masamori Kashiyama, and Satoru Tezuka, "A Studay on user authentication infrastructure for next generation telematic," *IEEE International Conferrence on Vehicular Electronics and Safety 2008(ICVES 2008)*, pp. 38-44, Sep. 2008.
5. Yuichi Kobayashi, Noboru Kiyama, Hirokazu Aoshima and Masamori Kashiyama, "A route search method for electric vehicle in consideration of range and locations of charging stations," *2011 IEEE Intelligent Vehicles Symposium (IV)*, pp.920-925, Jun, 2011.

■ 研究報告・解説

1. 齋藤拡二, 橋本眞宏, 澤本英雄, 熊谷多加史, 山縣良, 釜田栄樹, 松原健二, 柏山正守, 礒部敏子, 堀田多加志, 中野哲夫, 清水照久, 中澤喜三郎, "《招待論文》擬似ベクトル機構を有する並列コンピュータ向け RISC プロセッサ,", 電子情報通信学会技術研究報告, pp.1-6, 1995 年 10 月 20 日.
2. 藤井啓明, 保田淑子, 明石英也, 稲上泰弘, 柏山正守, 和田英雄, 住本勉, 河辺峻, "並列計算機 SR2201 の方式と評価,", 電子情報通信学会技術研究報告, pp.1-8, 1996 年 8 月 26 日.
3. 柏山正守, "《特集》高速プロセッシングデータバス技術の編集にあたって,", 情報処理, Vol.38, No.6, pp.457-459, 1997 年 6 月.
4. 澤本英雄, 石原修, 柏山正守, "RISC 超並列スーパーコンピュータのデータバス技術,", 情報処理, Vol. 38, No.6, pp.485-492, 1997 年 6 月.
5. 柏山正守, 北野昌宏, "日立アドバンストサーバ「HA8000-ex/880」,", 計算工学, Vol.6, No.4, pp.42-44, 2001 年 10 月 31 日.
6. 大黒浩, 庄山貴彦, 田中輝雄, 柏山正守, "Harmonious Computing を実現するサーバ製品,", 日立評論, Vol.85, No.7, pp.33-36, 2003 年 7 月 1 日.
7. 梅澤克之, 洲崎誠一, 柏山正守, "テレマティクス向け個人認証基盤の提案,", 電子情報通信学会技術研究報告, ITS, pp.133-138, 2008 年 2 月 18 日.
8. 木山昇, 小林雄一, 青島弘和, 柏山正守, "航続可能距離と充電スタンドの位置を考慮した電気自動車向けルート探索手法,", 電子情報通信学会技術研究報告, ITS, pp.1-8, 2011 年 9 月 21 日.

9. 木山昇, 小林雄一, 青島弘和, 柏山正守, “航続可能距離と充電スタンドの位置を考慮した電気自動車向けルート探索手法,”, 電気学会研究会資料, ITS 研究会, pp.1-15, 2011 年 9 月 28 日.
10. 小林雄一, 木山昇, 白井啓介, 柏山正守, “ドライバに安心を提供する電気自動車向けテレマティクスシステム,”, 自動車技術, Vol.66, No.12, pp.42-47, 2012 年 12 月 1 日.
11. 柏山正守, 宮澤浩久, “車内空間へのリアルタイムな情報提供を可能にする自動車向けクラウド型テレマティクスサービス,”, 日立評論, Vol.95, No.1, pp.119, 2013 年 1 月 1 日.
12. 木山昇, 小林雄一, 長船辰昭, 白井啓介, 柏山正守, “テレマティクスシステムを活用した電気自動車向け最短経路探索手法,”, 日立評論, Vol.95, No.6-7, pp.35-41, 2013 年 7 月 1 日.
13. Masamori Kashiyama, Hirohisa Miyazawa, “Cloud Telematics Service for Realtime Information in Vehicles,”, HITACHI REVIEW, Vol.62, No.5, pp.64, 13, Aug. 2013.
14. 高橋利光, 木山昇, 祖父江恒夫, 柏山正守, “テレマティクスサービス向けクラウド基盤の検討,”, 電気情報通信学会総合大会講演論文集, A-17 ITS, pp.230, 2014 年 3 月 19 日.
15. 柏山正守, 関弘翔, 細野裕行, “Pattern Match Accelerator を用いた Mobility IoT 向け Malware 検出の検討,”, 電気学会産業応用部門大会シンポジウム「ITS 技術とその応用」, 4-S6-4, 2019 年 8 月 21 日.
16. 柏山正守, 関弘翔, 細野裕行, “Pattern Match Accelerator を用いた Edge 向け Malware Cyber-Security の検討,”, 電気学会研究会報告, ITS 研究会, IEEJ-ITS, ITS-19-019, 2019 年 9 月 6 日.
17. 柏山正守, 関弘翔, 細野裕行, “CASE クルマの Cyber Security 実装検討,”, ITS-Japan, 第 17 回 ITS シンポジウム 2019, 1-A-09, 2019 年 12 月 12 日.