博 士 論 文

IoT 環境におけるサイバー攻撃に対する時刻同期技術を利用したイベント検知手法に関する研究

(Study on Event Detection Methods Using Time Synchronization Technique for Cyber Attacks on IoT Environment)

平成 31 年 3 月

川村　保

山口大学大学院理工学研究科

概要

現代社会において、IoT（Internet of Things） の利活用が急速に進んでいる。IoT は、従来のインターネット上のサービスだけでなく、産業用オートメーションや制御システムなどの社会インフラやビジネス、さらに個人生活にも浸透し始めており、実世界とインターネット世界が一体化して、公共や生命、財産に関わる情報を手軽に取り扱うことが可能になっている。その反面、インターネットと同様に、分散サービス拒否（DDoS）攻撃や不正アクセスなどのサイバー攻撃が目立ち始め、被害も深刻化している。このため、IoT に対するセキュリティ対策は緊急の課題となっている。

セキュリティ対策に関する国際規格では、対策の規範として、抑止、防止、検知、回復の 4 段階の対応を求めている。抑止や防止が対策の基本ではあるが、それらの対策に不備があったり、未知のサイバー攻撃を受けたりするとシステムや機密情報を守れないため、重篤な被害に陥るのを阻止し迅速な回復を行うためには、検知が重要な役割を果たす。国際規格では、システムの周辺状況が出現または変化した状態をイベントとして検知し、より深刻な状態であるインシデントやアクシデントに至るのを阻止することが求められている。

一般的なコンピュータネットワークに対する従来の検知手法として、パケットキャプチャやファイヤーウォール、侵入検知システム、ウィルス対策ソフトなどが利用されている。しかし、それらの検知手法は、IoT デバイスのシステムリソース量の制約やリアルタイム検知と専門家による保守作業の困難さなどの問題から、IoT 環境には不向きである。府省の IoT セキュリティガイドラインでは、IoT デバイスのリソース使用量やネットワーク負荷などの状態を把握することが提唱されているが、それらの情報をいかに役立てて精度よくイベント検知を行うかの方法論は未だに確立されていない。

そこで本論文では、コンピュータや IoT デバイスで一般的なネットワーク時刻

同期技術を利用するアプローチから、IoT 環境に適するイベント検知手法を明らかにすることを目的とする。本論文で利用するネットワーク時刻同期技術は、時刻同期サーバと NTP（Network Time Protocol）パケットを交換することにより、システムクロックを標準時刻に同期させるものである。本論文では、IoT デバイスがサイバー攻撃を受けた場合に、CPU やネットワークインタフェースの割り込み処理が平時と比較して多発し過負荷になることで、ネットワーク通信時間が遅延する現象と、システムクロックがずれ時刻補正値が大きくなる現象に着目する。すなわち、本論文における検知の原理は、被攻撃時に多発する現象に着目することで、インシデントの予兆を示すイベントか否かを精度よく区別することにある。

本論文では、IoT デバイスで直接的に情報を収集し判断を行うホスト型と、IoT デバイスが接続される通信処理装置で間接的に情報を収集し判断を行うネットワーク型の両イベント検知手法を提案する。さらに、本提案手法の有用性を実証するために、本手法に基づくイベント検知モジュールを開発し、擬似サイバー攻撃や不正アクセスを発生させて、イベント検知の実証実験を行った。その結果、検知率が 97〜99%、誤検知率が 2〜7% と正確性、網羅性、信頼性が高く、バランスがよいことがわかった。また、検知モジュールは、バイナリファイルのサイズが約 160KiB、物理メモリと仮想メモリの消費量が、それぞれ、約 7MiB、8MiB と非常に軽量であることも分かった。提案手法では、2 秒単位で直近の 1 分間の判定が可能であり、暗号化通信にも対応しており、定期的な更新や保守作業を必要としないことから、IoT 環境で利用可能なイベント検知の先駆的な研究として、その応用やさらなる発展が期待される。

目次

表目次

図目次

第 1 章

緒言

1.1 研究背景

近年、インターネットに様々なものを接続する IoT（Internet of Things）が普及し始めている。IoT はモノのインターネットと呼ばれており、IoT デバイス、IoT ネットワーク、IoT アプリケーションの 3 つの要素で構成される。1 つ目の IoT デバイスは、IoT の Things にあたり、情報を収集・計測または処理してデータ化するモノ自身である。例として、一般的なコンピュータやルータをはじめ、スマートデバイス、家電、ゲーム機、車両、医療機器、センサー、スマートメータ、制御モジュールなどがある。2 つ目の IoT ネットワークは、IoT の Internet にあたり、モノを相互接続する通信ネットワークである。例として、インターネットに加えて、インターネットに接続するための WAN 接続回線や LPWA（Low-Power Wide-Area Network）[1]、有線・無線 LAN、超近距離無線の 6LoWPAN（IPv6 over Low power Wireless Personal Area）[2]、Bluetooth や ZigBee[3] などがある。3 つ目の IoT アプリケーションは、Things と Internet を組み合わせて提供されるサービスである。例として、自動車分野における自動運転システム、交通分野におけるナビゲーションシステム、物流分野における自動配送・在庫管理システム、医療分野における遠隔診療システムを含めた IoMT（Internet of Medical Things）[4]、農業分野におけるスマート農業システム、金融分野におけるスマート認証システムやスマート決済システムなどがある。IoT では、IoT デバイスと IoT ネットワークを介して実世界とインターネット世界が一体化し、IoT アプリケーションを用いることにより、公共や生命、財産に関わる情報を手軽に取り扱うことが可能になっている。

IoT が本格的に普及し、人々の生活をより便利なものにするには、セキュリティ対策が不可欠となる。コンピュータの接続網としてのインターネットでは、分散サービス妨害攻撃（以降、DDoS（Distributed Denial of Service）攻撃）[5] や不正アクセスなどのサイバー攻撃 [6] と、その対策の攻防が繰り返されてきたが、IoT においても例外ではない。例えば、2016 年 9 月に、マルウェア Mirai[7, 8] に感染した IoT デバイスから、米国のセキュリティ情報サイトである Krebs on Security[9] に対して、当時史上最大規模である 665Gbp の帯域の DDoS 攻撃 [10, 11] が行われた。続いて、同年 9 月に、フランスのサーバレンタル企業である OVH 社に対して、1.1Tbps と 901Gpbs の 2 回の DDoS 攻撃が行われ、サイバー攻撃の帯域記録を更新した [12]。一般的な企業のサーバでは 10Gbps 程度、大企業の大容量サーバでは 100Gbps 程度の攻撃によりシステムがダウンすると言われており、これらの DDoS 攻撃はその数倍から数十倍の威力を持つものである。また、同年 10 月に、DNS サーバプロバイダーのダインに対しても大規模な DDoS 攻撃が行われ [13]、GitHub、Twitter、Reddit、Airbnb、ネットフリックスなどの国際的に利用されている Web サイトを含む多くの Web サイトでアクセス障害が発生した。この攻撃では、10 万台以上の Web カメラやデジタル・ビデオ・レコーダーなどの IoT デバイスが加担したと言われている [14]。このように、IoT に対するサイバー攻撃が目立ち始め、被害も深刻化しており、その対策は緊急の課題となっている [15]。

IoT は、インターネット上のアプリケーションのみならず、産業用オートメーション及び制御システム（以降、IACS（Industrial Automation and Control System））にも浸透し始めており、同様に、セキュリティ対策が求められている。IACS には、エネルギー分野における電力やガスなどの配電・配給システム、石油・化学や鉄鋼などのプラント、鉄道などの交通インフラ、機械や食品などの生産・加工ライン、ビルの管理システムなどがある。従来の IACS は、固定のホストコンピュータと専用端末、X.25[16, 17] や ATM（Asynchronous Transfer Mode）[18] などの高信頼性の通信手順、保証された専用の伝送路や閉域網、文字データやビットパータンを含む限定された情報媒体などにより、外部ネットワークから遮断された専用システムで構成されていた。そのため、セキュリティ上の脅威はほとんど意識されず、その対策はあまり検討されてこなかった。しかし、近年の IACS は、IoT に代表されるような民生用の PC やサーバコンピュータなどのハードウェア、IP（インター

ネットプロトコル）の低信頼性の通信手順、遠隔操作や遠隔保守に利用する無保証のインターネット、画像や音声データなどを含む多種多様なデータ形式の情報媒体などにより、誰でも利用できる公開されたネットワークで構成される。このような IACS がサイバー攻撃を受けると、社会インフラやビジネスだけでなく、個人の HSE（Health: 健康、Safety: 安全、Environment: 環境）に対しても深刻な影響が生じる。このため、IoT が備えるべき技術的なセキュリティ対策を具体化し、実装を急がなければならない段階にきている。

セキュリティ対策に関する国際規格として、情報セキュリティマネジメントシステム（以降、ISMS（Information Security Management System））[19, 20] や IACS 向けのサイバーセキュリティマネジメントシステム（以降、CSMS（Cyber Security Management System））[21, 22] がある。これらの国際規格では、セキュリティ対策の規範として、以下に示す 4 段階の対応を求めている。

抑止: システムや機密情報など資産として価値のある情報（以降、情報資産 [23, 24]）に内在する弱さである脆弱性を低減させること。

防止: 情報資産の脆弱性を突く攻撃である脅威から保護すること。

検知: 情報資産の危険を監視し、危険の顕在化を検出して人に通知すること。

回復: 情報資産が被害を受けた場合に、損害を局所化し、速やかに原状に復帰させること。

Web サイトを例にとると、抑止は、脆弱性であるセキュリティホールに対して、セキュリティパッチをあてることであり、防止は、セキュリティホールを突いて侵入しようとするウイルスに対して、ウイルス対策ソフトで対応すること、検知は、不審な通信や振る舞いに対して、不正アクセス検知システムで検知すること、回復は、サービス停止などの損害に対して、バックアップサイトへの切り替えにより対応することに相当する。セキュリティを高めるには、抑止と防止による対策が重要であることは言うまでもない。しかし、それらの対策に不備があったり、未知のサイバー攻撃を受けたりした場合、システムや機密情報を完璧に守ることは困難である。このため、多様に進化し続けるサイバー攻撃に対応し、重篤な被害に陥ることを食い止め、迅速な回復を行うためにも、最後の砦として、検知が最も重要な役割

を果たす。

検知に関しては、ISMS や CSMS では、攻撃の予兆から被害に至るまでの過程で発生する事象を検知し、それが以下のいずれであるかを判断して通知することが求められている。

イベント： 一連の周辺状況が出現または変化した状態。

インシデント： 潜在的なリスクが顕在化して事件に至った状態。

アクシデント： インシデントが金銭的な損害や社会的な信用の失墜にまで至った状態。

Web サイトへの DDoS 攻撃を例にとると、イベントは、Web サイトへの同時アクセス量が通常時と比べて継続的に急増している状態であり、インシデントは、同時アクセス量が許容限界を超えてサービスが停止した状態、アクシデントは、アクセス障害により利益の損失などの実被害が発生した状態に相当する。当初の ISMS の標準規格 [25, 26, 27] においては、インシデントの検知が中心であった。しかし、インシデントの検知に手間取ると、インシデントが拡大してアクシデントに至ることを防止できず、これが原因でアクシデントが引き起こされる場合が数多く発生していた。そこで、最新の ISMS の標準規格 [28] においては、イベントの検知を強化して、インシデントの防止や発生後の対応に備えることで、セキュリティ対策を確実にすることが求められている。

DDoS 攻撃や不正アクセスなどのサイバー攻撃に対する従来の検知手法として、パケットキャプチャ [29, 30]、ファイアウォール [31, 32]、侵入検知システム（以降、IDS（Intrusion Detection System））[33, 34]、ウイルス対策ソフトウェア [35, 36] などがある。パケットキャプチャは、通信路上の通信データを傍受して解析、集計したり、通信量の変化を監視したりするものであり、異常パケットや通信障害を検知する。ファイアウォールは、コンピュータやネットワークへの通信の許可または拒否などのアクセス制御を行うものであり、通信データやログに記録された情報から不正探査や侵入を検知する。IDS は、コンピュータやルータへの侵入やその兆候を監視するものであり、通信路上の通信データやログに記録された通信データ、通信手順と予めデータベースに登録された情報を照合し、異常を検知する。シグネチャ型の IDS[37, 38] では、数千種類を超える既知の攻撃手口と一致するか照合し、

アノマリ型 [39, 40] の IDS では、許可された正しいルールから逸脱していないかを照合する。ウイルス対策ソフトウェアは、コンピュータやルータへの侵入や攻撃を行うウイルスやワームなどの不正ソフトウェアから保護するものであり、不正ソフトウェアをパターンマッチング法 [41, 42] やビヘイビア（振る舞い）法 [43, 44] などにより検知する。前者は、不正ソフトウェアのプログラムコードの特徴を記録したパターンファイルと検知対象のプログラムコードと一致するか、後者は、既知の不正ソフトウェアの不審な振る舞いと検知対象のソフトウェアの振る舞いと一致するかを照合する。

上記の従来の検知手法を IoT のイベント検知に適用しようとする場合、以下の問題が挙げられる。パケットキャプチャ、ファイアウォール、侵入検知システムに共通する問題として、検知期間中は、常時、通信データやシステムのログファイルを記録するため、メモリまたはディスクの記憶領域を大量に消費する。また、それらのファイルは、通常、専門家が数日または数時間単位の作業時間をかけて解析する。ファイアウォール、侵入検知システム、ウイルス対策ソフトウェアに共通する問題として、検知期間中は、検知プロセスにより、常時、CPU やメモリの使用率が高くなる。また、解析結果に基づいて、設定ファイルやパターンファイルを緊急的または定期的に調整する作業する必要がある。IoT デバイスのシステムリソースは一般的な PC と比べると極端に少なく、また、専門家による関与も困難である場合が多い。以上のように、これらの従来の検知手法は、IoT を対象に侵入または攻撃を受ける前のタイミングでイベントを検知し、それがインシデントの予兆であるか否かを判断する目的には不向きである。

1.2 研究の目的

IoT 向けのイベント検知として、総務省と経済産業省の IoT セキュリティガイドライン [45] では、IoT デバイスの状態や通信状態を記録し、システムリソースの使用量やネットワーク負荷などの情報を監視することを提唱している。これら情報やその統計量は、Linux 系 OS では top[46]、sar[47]、netstat[48] などのコマンドにより取得できる。しかし、これらの情報や統計量だけで、平時の使用量や負荷であるか、攻撃や不正アクセスによる使用量や負荷なのか、を判断することは容易ではない。すなわち、IoT デバイスの情報監視の必要性は示されているものの、取得

した情報をいかに役立てて精度よくイベント検知を行うかの方法論は確立されていない。

そこで、本論文では、IoT 環境におけるサイバー攻撃のイベントを検知するために、IoT デバイスに組み込み可能な精度のよいイベント検知の実現方法を明らかにすることを目的とする [49, 50, 51]。一般的に、検知手法には、検知が行われる場所により、ホスト型 [52] とネットワーク型 [53] の 2 種類がある [54]。ホスト型は、ホスト（IoT デバイス）で直接的に情報を収集し、自身に発生するイベントやインシデントを検知する手法であり、ネットワーク型は、ルータや無線アクセスポイントなどのネットワーク機器で間接的に情報を収集し、アクセスドメイン内のホストに発生するイベントやインシデントを検知する手法である。本論文では、IoT デバイスが使用される環境の違いに対応するために、ホスト型とネットワーク型の双方に対するイベント検知手法を検討する。本論文では、インシデントの予兆としてのイベントを精度よく検知するために、コンピュータや IoT デバイスで一般的に利用されるネットワーク時刻同期技術を利用するアプローチをとる。これは、ネットワークに接続されるシステムのシステムクロックを標準時刻に同期させる技術であり、NTP（Network Time Protocol）を用いて時刻同期サーバと NTP パケットを交換することで、時刻の調整を行う技術である。本論文では、IoT デバイスがサイバー攻撃を受けた場合に、CPU やネットワークインタフェースの割り込み処理が平時と比較して多発して過負荷になることで、システムクロックが大きくずれ、ネットワーク通信時間が遅延することに着目する。すなわち、標準時刻を検知の基準とし、NTP パケットから得られる時刻調整量やネットワーク通信時間のずれの量を用いてイベント検知を行うことで、IoT デバイスの本来の利用目的に及ぼす影響を軽減する。

本論文では、IoT 環境におけるサイバー攻撃に対して、ネットワーク時刻同期技術を利用したホスト型とネットワーク型のイベント検知手法を提案する。また、本提案手法の有用性を実証するために、本手法に基づくイベント検知モジュールを開発し、IoT デバイスに組み込み、擬似サイバー攻撃を発生させて、イベント検知の実証実験を行う。本論文で明らかにするイベント検知手法には、以下の特徴がある。本検知手法は、ネットワーク時刻同期技術を利用し、検知尺度を標準時刻とするため、インシデントに至る前にイベントを精度よく検知できる。また、検知は簡

単なメカニズムに基づいているため、システムリソースの消費が少なく、専門家の関与を前提とした更新や調整作業も不要である。さらに、暗号化通信にも対応しているため、幅広い用途での応用が期待される。

1.3　本論文の構成

本論文の構成は、以下のとおりである。

第 1 章は緒言であり、本研究の背景と目的について述べる。

第 2 章では、従来のイベント検知手法を分類し、IoT のイベント検知の要件を明らかにする。まず、情報セキュリティに関する標準規格について、IoT のセキュリティ対策の指針や規範となる国際規格、各国政府のガイドラインにおける要求される対策を述べ、検知の位置づけを述べる。つぎに、サイバー攻撃と不正アクセスについて、攻撃の段階毎に手口の内容をまとめ、それらのサイバー攻撃に対する従来の検知手法を、検知が行われるタイミングとシステムリソースの観点で分類する。また、それらの従来の検知手法の問題点を考察し、IoT に求められるイベント検知の機能に対する要件を明らかにする。そして、その要件について、システム監視による異常検知による予備実験の結果を示す。

第 3 章では、ネットワーク時刻同期技術を用いたホスト型のイベント検知手法を提案し、実証実験を行う。本手法は、IoT デバイス自身が直接検知を行える場合の手法である。まず、本手法で着目するネットワーク時刻同期技術について、NTP による時刻補正の仕組みと時刻同期モジュールによるローカルクロックの補正方法を述べる。つぎに、ホスト型のイベント検知手法について、検知の原理と検知手法を詳しく述べる。さらに、本手法を実装した検知モジュールを用いた実証実験について、実験目的と方法、実験結果を述べ、本提案手法の有効性を評価する。

第 4 章では、ネットワーク時刻同期技術を用いたネットワーク型のイベント検知手法を提案し、実証実験を行う。本手法は、IoT デバイスが直接検知を行えない場合の手法である。まず、ネットワーク型のイベント検知手法について、検知の原理と検知手法を述べる。つぎに、実証実験について、実験目的と方法、実験結果を述べ、本提案手法の有効性を評価する。

最後に、第 6 章で、本研究のまとめと今後の課題、将来性の展望について述べる。なお、付録 A として、第 3 章で述べた予備実験の結果の詳細について述べる。

第 2 章

情報セキュリティ対策に用いられる従来の検知手法と問題点

2.1 はじめに

本章では、本論文が IoT 環境におけるサイバー攻撃のイベント検知手法を提案するにあたって、IoT 環境のサイバーセキュリティに関連した国際規格やガイドラインの規範と要求事項を洗い出し、セキュリティ対策における検知の位置づけと、イベント検知の重要性を明確にする。次に、サイバー攻撃と不正アクセスの手口の種類と特徴を示し、それらに対する従来の検知手法について述べる。特に、府省の IoT セキュリティガイドラインでは、システム資源を記録し把握することを求めていることから、これに最も近い従来の検知手法として、異常検知によるイベント検知について述べる。また、従来のイベント検知手法に関して、検知のタイミングとシステムリソース量の観点で分類しながら研究動向をまとめる。それらの従来手法を IoT 環境に適用した場合の問題点を考察した上で、IoT で求められるイベント検知に対する要件を明らかにする。本章の最後に、IoT デバイスにおいて、異常検知で汎用的なシステム監視を用いた予備実験を行い、有効性および問題点を検証する。なお、本予備実験の詳細な内容は、付録 A で述べる。

2.2 情報セキュリティに関する国際規格やガイドラインの指針と規範

国際規格（Global standard）とは、製品の品質、性能、安全性、寸法、試験方法などに関する国際的な取極めのことである [55]。国際規格の制定機関としては、

ISO（国際標準化機構、International Organization for Standardization）、IEC（国際電気技術会議、International Electro-technical Commission）、ITU（国際電気通信連合、International Telecommunication Union）などがある。情報セキュリティに関する国際規格として、各国際規格で共通なリスクマネジメント、組織全体に向けられたセキュリティ、ネットワークセキュリティ、サイバーセキュリティ、製品のセキュリティなどがある。主要なガイドラインとして、米国と日本国政府のガイドラインがある。これらの国際規格やガイドラインは、社会インフラや産業インフラで利活用される IoT 環境のセキュリティ対策にとって、安全性や信頼性、整合性を担保するための定義、実践、見直し評価の規範であり、国際規格の適合評価制度の認証基準でもある。

まず、ISO/IEC の国際規格について述べる。

- ISO 31000:2018（Risk management – Guidelines)
  本国際規格 [56]（JIS 版規格 [57]）は、リスクマネジメントの国際規格で、情報セキュリティ（ISMS)、サイバーセキュリティ（CSMS)、IT サービス（ITSMS)、品質（QMS)、環境（EMS）などにとって共通基盤であり、それらのリスクマネジメントに関連する国際規格の統一性と整合性を図っている。本国際規格は、リスクアセスメントの過程（プロセス）を、リスクの特定、分析、評価、対応、モニタリング及びレビューの一連の活動として規定し、そのプロセス全体のマネジメント方針、手順及び実務の体系的な適用を求めている [58]。情報セキュリティの検知は、リスクアセスメントにおけるリスクの特定にあたり、リスクの原因及びリスク源、リスクの好ましい結果及び好ましくない結果、これらの結果が発生することがある起こりやすさを考慮して、結果及び起こりやすさに影響を与える要素を特定することである。

- ISO/IEC 27000（Information technology – Security techniques – Information security management systems）seriase
  本国際規格シリーズは、ISO/IEC JTC 1/SC 27 による情報セキュリティ（ISMS）に関する規格群である。
  – ISO/IEC 27001:2013（Information technology – Security techniques

– Information security management systems – Requirements）

本国際規格 [59] は、組織の情報セキュリティマネジメントシステム（ISMS）の適合評価制度の認証規格で、組織の事業リスク全般を考慮して、文書化した ISMS を確立、導入、運用、監視、レビュー、維持及び改善するための要求事項を規定している。

– ISO/IEC 27002:2013（Information technology – Security techniques – Information gurity management systems – Code of practice for information security controls）

本国際規格 [60] は、ISMS の実践のための規範で、ISMS の導入、実施、維持及び改善に関する規範（ベストプラクティス）を示している。ISMS において、事件事故に至らせない対応をセキュリティコントロールと呼び、それを具現化した対策が管理策である。管理策には、抑止、防止、検知および回復改善の 4 つの段階がある。このうち、検知に関しては、前版の ISO/IEC 27001:2005[27] では、インシデント検知が中心であった。しかし、インシデントの検知に手間取ると、インシデントが拡大してアクシデントに至る場合が数多く発生していた。そこで、最新版の ISO/IEC 27001:2013[59] においては、イベント検知を強化して、インシデント検知とその後の対応に備えることで、セキュリティ対策を確実にすることを求めている。

– ISO/IEC 27033（Information technology – Security techniques – Network security）series

本国際規格 [61] は、ネットワークセキュリティに関する概念、リスク、実装時の設計手法や管理策などのガイダンスである。

– ISO/IEC 27032:2012（Information technology – Security techniques – Guidelines for cybersecurity）

本国際規格 [62] は、サイバーセキュリティの指針と実践のためのベストプラクティスである。本国際規格は、サイバーセキュリティの概要、他のセキュリティ分野との関係、担当者や関係者の役割分担、共通課題から構成されており、これを図 2.1 に示す。この図は、サイバーセキュリティは、情報セキュリティの範囲内で、アプリケーションセ

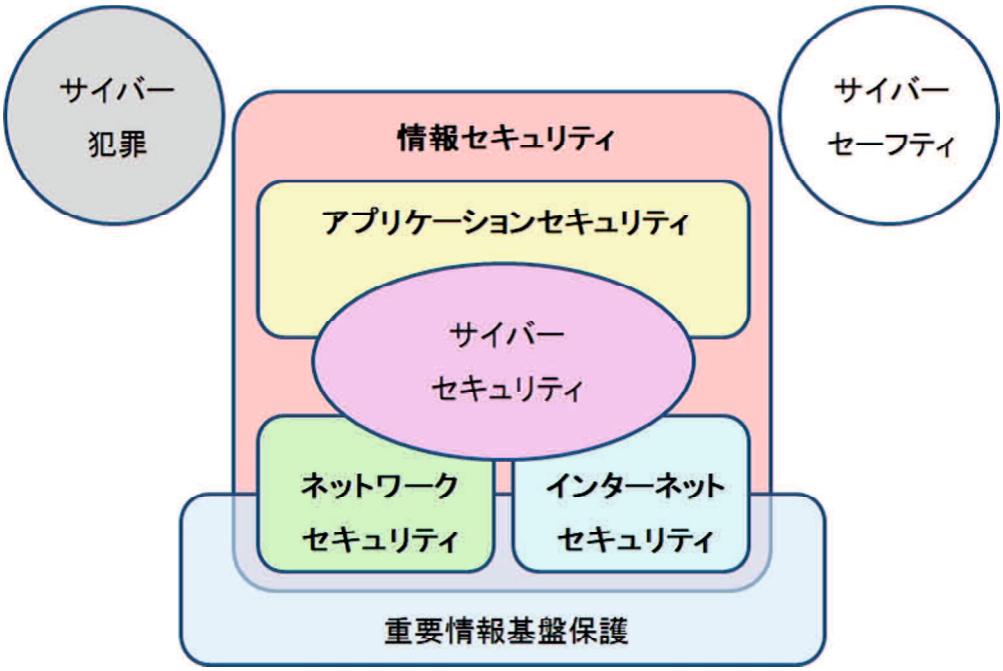

図 2.1　サイバーセキュリティとその他セキュリティ分野の関係

キュリティ、ネットワークセキュリティおよびインターネットセキュリティから構成され、重要情報基盤と、サイバー犯罪対策とサイバーセーフティーとの関係を示している。この構成は、IoT 環境の 3 つの要素である IoT アプリケーション、IoT ネットワーク、IoT デバイスと重なる。本国際規格では、特に、サイバー空間で多様化し高度化する脅威対策として、サイバー空間の脆弱性や攻撃手法の表現・格納方法を標準化して、国境を越えた世界規模で交換し共有化することを提唱している。本国際規格が参照するサイバーセキュリティの情報共有の技術的フレームワークとして、ITU-T SG17 のプロジェクトによる CYBEX（The Cybersecurity Information Exchange Framework）により、X.1500（Overview of cybersecurity information exchange）が勧告されている。本国際規格は、脆弱性データベースに反映され、WWW により共有化され、一般的な脆弱性試験に利用されている。

- ISO/IEC 15408（Information technology – Security techniques – Evaluation criteria for IT security）series

  本国際規格 [63] は、ハードウェア製品及びソフトウェア製品のセキュリティ機能の機能要件と保証要件の有効性について、企画、開発から製造、運用及

び保守に至るまで適切であるかの評価基準を規定する適合審査基準である。適合審査基準は、製品やシステムのセキュリティ要件に対して、軍事、政府調達から一般家庭向けに 7 段階でセキュリティ強度の評価保証レベル（EAL : Evaluation Assurance Level）が規定されており、その実装の確かさを評価する。本国際規格を相互認証する加盟国にとって、コモンクライテリア承認アレンジメント（CCRA, Common Criteria Recognition Arrangement）は条約に準ずる国際協定である。この　CCRA の加盟国では、国内外政府調達のハードウェア製品及びソフトウェア製品に対して本国際規格の評価保証レベルを満たす必要があり、製品の輸出入の条件となっている。よって、IoT デバイスも、本国際規格に基づいた評価保証レベルを満たす必要がある。

- IEC 62443-2-1:2010（Industrial communication networks - Network and system security - Part 2-1: Establishing an industrial automation and control system security program）

  本国際規格 [64] は、CSMS（Cyber security management system for IACS）の認証基準である。CSMS とは、IACS（Industrial Automation and Control System）は、産業用オートメーション及び制御システム（IACS : Industrial Automation and Control System）を対象としたサイバーセキュリティのマネジメントシステムである。最近の IACS は、これまでの専用システムとは異なり、民生用の機器や IoT と公開された網であるインターネットから構成されるため、サイバー攻撃の対象となっている。CSMS の対象者は、制御システムのライフサイクルを考慮し、制御システムのオーナーである事業者に加え、システムの構築や運用開始後のシステム改修、維持保全を分担する事業者及びシステムインテグレータとしている。CSMS の目的を以下に述べる。CSMS に基づくセキュリティ対策を実施することで、サイバー攻撃に対するリスクを低減すること。IACS の運用担当者に対して、行動指針を徹底することで、ヒューマンエラーや組織に起因するセキュリティインシデントの発生可能性を低減すること。CSMS 構築・運用により、企業内セキュリティガイドラインの改訂とともに、企業内の各事業所の間で運用実態が明確になり、継続的な改善を行うこと。規格に基づくセキュリティ管理策を実施

することを認証基準としている。よって、IoT デバイスも、本国際規格に基づいた認証基準を満たす必要がある。

- ISO/IEC 20000（Information technology – Service management）seriase 本国際規格 [65] は、IT サービスに関する、IT サービスを提供する組織の IT サービスマネジメントが適切であるかどうかを評価するための認証基準である。本規格は、サービスレベル管理（SLM: Service Level Management) として、通信サービスや IT サービスなどで、提供者がサービスの品質について継続的・定期的に点検・検証し、品質を維持あるいは改善する仕組みを示している。これに準拠して、実際のシステムの稼働状況や対応状況を継続的に記録し、事前に定めた目標水準に照らして十分な品質を保っているか定期的に点検することを求めている。さらに、本規格は、サービス品質保証（SLA: Service Level Agreement）として、サービス品質の目標水準はサービスを提供する組織と提供を受ける組織の間で事前に文書などの形で合意・契約することを求めている。よって、IoT 環境に関連した IT サービスは、本国際規格に基づいた認証基準を満たす必要がある。

次に、重要な日米政府のガイドラインについて述べる。

- NIST SP 800-94（Guide to Intrusion Detection and Prevention Systems）本ガイド [66] は、NIST（National Institute of Standards and Technology : 米国国立標準技術研究所）のサイバー攻撃に対するリスクマネジメントに関するレポートの SP800 シリーズ [67] のうち、侵入検知及び侵入防止システム（IDPS）に関するガイド [66] である。本ガイドでは、IDS に関する技術を、ネットワークベース、無線、NBA（Network Behavior Analysis : ネットワーク挙動解析）及びホストベースの 4 つの種類に分類し、各 IDPS テクノロジーを実践するための設計、導入、構成、保護、監視、保守の推奨事項を示している。IDPS テクノロジーは、監視対象のイベントの種類とテクノロジーの導入方法により、根本的に異なる情報収集、ログ、検知、防止機能を提供する。それぞれ異なる長所を備えており、多種多様な環境に堅牢で実効性のある DPS ソリューションを実現するは、複数種類の IDPS テクノロジーの併用と統合が不可欠であるとしている。

- NIST SP 800-160 Volume2（Systems Security Engineering: Cyber Resiliency Considerations for the Engineering of Trustworthy Secure Systems）

  本ガイド [68] は、NIST SP800 シリーズのうち、システムセキュリティエンジニアリングとして、特にセキュアな IoT デバイスを開発する上で必要な作業についてのガイドラインである。

- NIST IR 8200（DRAFT）（Interagency Report on Status of International Cybersecurity Standardization for the Internet of Things（IoT））

  本ガイド [69] は、米国土安全保障省（DHS）と米国立標準技術研究所（NIST）がリリースした、IoT デバイスの開発や導入担当者に向けて、セキュアな IoT 開発のフレームワークを構築するための IoT セキュリティのガイドラインである。すでに、DHS がリリースした Internet of Things Fact Sheet（IoT ファクトシート）と Strategic Principles for Securing the Internet of Things（セキュアな IoT のための戦略的原則）と共に、セキュアな IoT を実現するために、以下の 6 つの戦略的原則を示している。設計段階でセキュリティを組み込む、セキュリティの改善と脆弱性対策を進める、従来のセキュリティ手法を拡張する、潜在的な影響に応じてセキュリティ対策の優先度を決める、IoT 全体を通じて透明性を促進する、接続には慎重を期する、である。

- IoT セキュリティガイドライン

  本ガイドライン [45] は、経済産業省及び総務省の配下の IoT 推進コンソーシアム IoT セキュリティワーキンググループにより、IoT やシステム、サービスの提供にあたってのライフサイクル（方針、分析、設計、構築・接続、運用・保守）における指針と、一般利用者に向けたルールを示している。例示として次の要点 13 がある。要点 13. 機器等がどのような状態かを把握し、記録する機能を設ける。ポイント: 機器等の状態や他の機器との通信状況を把握して記録する。

  ・各 IoT 機器・システムで動作をログとして記録する。

  （記録する内容の例）

  - セキュリティ解析用：攻撃、ユーザ認証、データアクセス、構

成管理情報更新、アプリケーション実行、ログの記録開始・停止、通信、扉の開閉、チェックサム、移動履歴

- セーフティ解析用：故障情報（ハードウェア／ソフトウェア）
- リライアビリティ解析用：結果情報、状態情報、動作環境情報（温度、湿度、CPU 負荷、ネットワーク負荷、リソース使用量等）、ソフトウェアの更新

・ログを保管するためのリソースは有限であるため、保管方針を策定する。

・関連する IoT 機器・システム間でログの記録時間が整合するように、時刻の同期を行う。

・ログに記録するタイミングは機器毎に設計するのではなくて、IoT 機器・システム全体で考慮する。

・ログの記録が IoT 機器・システムの保全のためであることをマニュアル等に記載する。

- 制御システムのセキュリティリスク分析ガイド 第 2 版

  本ガイドライン [70] は、セキュリティリスク分析を実施するための手順や手引きを示したガイドラインある。また、本ガイドラインは、重要なセキュリティ対策項目をベースにチェックリスト化しており、

  1. ネットワークセキュリティ
  2. サーバ・端末セキュリティ
  3. ベンダー管理
  4. 人的セキュリティ
  5. 物理セキュリティ
  6. セキュリティ監視
  7. インシデント対応
  8. 第三者評価

  の 8 要素から構成されている。よって、制御システムの IoT は、本ガイドラインに基づいたセキュリティリスク分析を実施する必要がある。

以上の国際規格やガイドラインにより、IoT 環境には、以下のセキュリティ対策および検知が求められている。

- 多様化し高度化する新たなサイバー攻撃の特徴や手口を、世界規模で標準化し共有化して、対策や検知に反映すること。
- IoT 環境を利用した最近の IACS に向けたサイバーセキュリティ対策を強化するために、IoT デバイスを含めた製品のセキュリティ機能の機能要件を明確にして保証すること。
- 監視対象のイベントの種類を明確にし、それに対応したテクノロジーにより、情報収集、検知、緊急対応の機能を提供すること。
- 特に、IoT デバイスがどのような状態かを継続的に把握し、記録する機能を設けること。記録から、インシデントに至る前に、インシデントの予兆をイベントとして検知して、その後の緊急対応を速やかかつ確実にすること。

IoT 向けのイベント検知手法を開発する上では、特に、最後の 2 点を満たすことが重要になる。

2.3 サイバー攻撃と検知手法

2.3.1 サイバー攻撃の段階と攻撃・不正アクセスの手口

サイバー攻撃とは、ネットワークを通じて各国の国防、治安等をはじめとする各種分野の情報システムに侵入し、データを破壊、改竄するなどの手段で国家等の重要システムを機能不全に陥れる行為 [71] である。一般的に、サイバー攻撃には以下の段階があり、プロジェクト体制下で、専門化、分業化され、巧妙かつ計画的に実施される。

事前調査段階: 攻撃、不正アクセスの標的候補をリストアップする。

探査段階: リストアップされた標的候補のシステムやネットワークに関する情報を収集し、侵入可能な標的をノミネートして、侵入する入口や手口を選定する。

侵入段階: 実際に標的に侵入し、制圧する準備としてウイルス感染やシステム特権の奪取を行う。

攻撃段階: 侵入した標的に対して、情報の奪取、改竄や棄損及びサービス妨害や停止などを行う。

各段階で用いられる攻撃と不正アクセスの手口を以下に述べ、非技術的攻撃である事前調査段階を除き、表 2.1 に示す。

事前調査段階では、一般社会で閲覧可能な Web サイトや新聞・TV などのマスメディアから得られる組織の公開情報から、組織や個人の社会的地位や保有する企業情報や個人情報の金銭的価値について調査し、評価が高いものを標的候補としてリストアップをする。

探査段階の手口として、事前調査段階でリストアップした標的候補に対して、OSI 参照モデル [72] の下位層から上位層に順に遡りながら、以下の技術的調査を行い、標的への侵入の入口や手口の選定をする。下位層の入口として、ARP スキャン [73] により、標的の MAC アドレスを調査する。次の上位層の入口として、IP アドレススキャン [74]、ポートスキャン [75] により、標的の IP アドレスとポートを調査する。最上位層の入口として、脆弱性試験（Vulnerability Scanner）[76] や URL 自動探査ツール [77] により、標的のサービスやアプリケーションを調査する。脆弱性試験では、標的のサービスやアプリケーションに関する既知の攻撃と不正アクセスの手口をリストアップし、実行することも可能である。なお、本来の脆弱性試験の利用目的は、試験対象の脆弱性の検出とその改善で、不正利用や悪用ではない。脆弱性試験には、前述の ARP スキャン、IP アドレススキャン、ポートスキャンの機能を含んでおり、ツール例として、攻撃対象の特徴を識別するフィンガープリンティング [78] を行う Nmap[79] や Xprobe[80]、システム機器構成評価を行う Nessus[81]、InsightVM[82] や Penetrator[83]、Web サイトへの既知の攻撃手口による疑似攻撃を行う OWASP ZAP（Zed Attack Proxy）[84]、Paros[85]、Ratproxy[86]、Burp Suite[87] や Nikto[88] などがある。

侵入段階の手口として、探査段階で選定した入口から侵入する。標的を制圧する手口として、主になりすましとトロイの木馬型ウイルス感染 [89] がある。なりすましは、標的のユーザ認証に対してログインパスワードクラック [90] としてブルートフォース攻撃（brute force attack）[91]、辞書攻撃（dictionary attack）[92]、レインボーテーブル攻撃 [93] を行って、正規のユーザとなることである。トロイの木馬型ウイルス感染では、攻撃対象のユーザに悪意のある電子メールを送り付けたり悪意の Web サイトを閲覧させたりするなどにより、ユーザをだまし標的に侵入する。トロイの木馬型ウイルスは、攻撃対象の内側から通信ポート（バックドア [94]）を

開設し、攻撃段階で使用するプログラムやツール群（ツールキット [95]）を事前に標的に送り込んだり、攻撃対象をボット化して攻撃の踏み台に仕立てあげたりする。ただし、サイバー攻撃として、この段階では、まだ深刻な被害に至っていない。

攻撃段階の手口として、実際に攻撃対象に被害を与える段階で、主に各種マリシャスコード [96] と各種サービス妨害による攻撃がある。

マリシャスコードによる攻撃には、OSI 参照モデルの上位 3 層のアプリケーション関連に向けた攻撃と、下位 4 層の通信機能関連に向けた攻撃がある。前者のアプリケーション関連としては、さらにメール、WWW、P2P やその他に向けた攻撃に分けられる。メールに向けたマリシャスコードとして、スクリプト型 [97] と添付型ワーム [98] がある。スクリプト型では、メールのメッセージに悪意のスクリプトを埋め込み、セキュリティホールを悪用しメールを読むだけで感染する BubbleBoy や KakWorm などのワームがある [99]。添付型では、添付ファイルとしての Bugbear、Beagle や Netsky などのワームがある [100]。WWW に向けたワームのマリシャスコードとして、Web エンジンや Web 系言語の脆弱性を突くマリシャスコード、Web アプリケーションの脆弱性を突く CodeRed[101] や Slammer[102] などがある。WWW に向けたワーム以外のマリシャスコードとして、SQL インジェクション [103] や、動的 Web ページの脆弱性を突くクロスサイトスクリプティング [104] などがある。P2P のマリシャスコードとして、ファイル共有ソフトを利用するための P2P ネットワークの脆弱性を突く Gile や Antinny などがある [105]。その他の通信機能関連に向けたマリシャスコードとして、TCP/IP、FTP、SMB/CIFS の脆弱性を突く Sircam、Blaster や Sasser がある [106]。後者の下位 4 層の通信機能関連に向けた攻撃では、侵入段階で開設したバックドアから侵入し、予め攻撃対象に送り込んでおいたツールキットを遠隔操作して、標的のユーザ認証の情報やユーザのオンラインバンキングやクレジット決済など、信用情報の ID パスワードや機密または重要ファイルなどを漏洩させる。また、遠隔操作により、踏み台のユーザになりすまし第三者の攻撃対象に攻撃を行い、踏み台を犯罪者に仕立てる。

サービス妨害には、直接的と間接的サービス妨害がある。直接的なサービス妨害では、攻撃側から直接、特定の攻撃対象を攻撃（スピア攻撃 [107]）する。OSI 参照モデルの上位 3 層のアプリケーション関連に向けた攻撃として、大量の問合デー

タを送り付け、それを何度も反復させるリフレクション攻撃、大容量のファイルを添付した大量のメールを転送する攻撃がある。下位 4 層の通信機能関連に向けた攻撃として、攻撃大量の通信パケットを断続的に送り付けるバッファオーバフロー攻撃、膨大な数の通信ノードへの接続確認を行う PoD（Ping of Death）[108]、攻撃対象へ攻撃対象自身の IP アドレスを送信元及び送信先アドレスを指定してアドレス解決要求パケット（ARP）を送信することでフリーズやクラッシュさせる重複 ARP（Conflicted ARP）[109] があり、これらを総じて DoS 攻撃と呼ぶ。間接的なサービス妨害では、攻撃側の C&C サーバ（Command and Control server）[110] が、侵入段階でバックドア開設した膨大な数のボットを遠隔操作して、間接的にサービス妨害を行う攻撃で、これらを総じて DDoS 攻撃と呼ぶ。1.1 節で述べた現在最も深刻なサイバー攻撃である DoS/DDoS 攻撃の具体的な手口は、次節で述べる。

表 2.1　OSI 参照モデルにおける攻撃と不正アクセスの段階と手口

<table>
<tr><td colspan="2">OSI
参照モデル</td><td>探査段階</td><td colspan="2">侵入段階</td><td colspan="2">攻撃段階</td></tr>
<tr><td>層</td><td>名 称</td><td>侵入口の探査</td><td>なりすまし</td><td>マリシャスコード</td><td>マリシャスコード</td><td>サービス妨害</td></tr>
<tr><td>7</td><td>アプリケーションン層</td><td rowspan="3">脆弱性試験、URL 自動探査ツール</td><td rowspan="3">パスワードクラック、フィッシング、ディレクトリトラバーサル</td><td rowspan="3">各種ウイルス（スパイウェア、アドウェア、ハッキングツール、ルートキット、ドロッパー）</td><td rowspan="3">各種ワーム（メール型、SQL インジェクション、クロスサイトスクリプティング）</td><td rowspan="3">L7（7 層）DoS/DDoS 攻撃（リフレクション攻撃)、標的型大容量ファイル添付メール転送、バッファオーバフロー、スパム</td></tr>
<tr><td>6</td><td>プレゼンテーション層</td></tr>
<tr><td>5</td><td>セッション層</td></tr>
<tr><td>4</td><td>トランスポート層</td><td rowspan="2">IP アドレススキャン、ポートスキャン</td><td rowspan="2">IP スプーフィング（IP アドレス偽装）</td><td rowspan="2">トロイの木馬型ウイルス（バックドア）</td><td rowspan="2">ネットワーク型ワーム（遠隔操作）</td><td rowspan="2">DoS/DDoS 攻撃（フラッド型、脆弱性型）</td></tr>
<tr><td>3</td><td>ネットワーク層</td></tr>
<tr><td>2</td><td>データリンク層</td><td rowspan="2">ARP スキャン</td><td rowspan="2">ARP スプーフィング（MAC アドレス偽装）</td><td rowspan="2">—</td><td rowspan="2">—</td><td rowspan="2">バッファオーバフロー攻撃、PoD（Ping of Death)、重複 ARP（Conflicted ARP）</td></tr>
<tr><td>1</td><td>物理層</td></tr>
</table>

2.3.2 DoS 攻撃/DDoS 攻撃の種類と手口

DoS 攻撃/DDoS 攻撃は、特に近年、社会的に深刻な被害を発生させているサイバー攻撃であり、大別するとフラッド型と脆弱性型がある。

フラッド型は、攻撃対象のサービスを遅延や停止させる種類の攻撃で、RFC 技術仕様に準じた正常なパケットを大量に送り付け、攻撃対象の通信帯域幅や通信処理能力の容量を浪費し過負荷にさせる手口がある。フラッド型の手口で送り付けられるパケットの発信元 IP アドレスは正当であるため、1 つのパケットのヘッダやデータだけでは、正常か異常か判定することが困難である。フラッド型の攻撃対象は、OSI 参照モデルの階層の違いから、トランスポートポート層に対する TCP/UDP フラッド攻撃 [111] とアプリケーション層に対する WWW/DNS/NTP フラッド攻撃 [112] があり、それぞれ、表 2.2 と表 2.3 に示す。TCP/UDP フラッド攻撃には、TCP のフラグやデータのオフセットを悪用して、TCP ポートへのコネクション要求、ハーフコネクション要求、データのフラグメントを大量発生させ、UDP ポートへの最大長データの大量送付などがある。WWW/DNS/NTP フラッド攻撃には、HTTP GET/POST Flood Attack、Slow HTTP DoS Attack、DNS Flood Attack、DNS reflection Attack（DNS amplification Attack）、NTP reflection Attack がある。

脆弱性型は、攻撃対象に異常処理をさせ、クラッシュやフリーズなどを発生させる種類の攻撃である。脆弱性型は、攻撃対象となる OSI 参照モデルの階層の違いから、ネットワーク層に対する ICMP/IP フラグメント攻撃とアプリケーション層に対する UDP フラグメント攻撃があり、それぞれ、表 2.4 と表 2.5 に示す。ICMP/IP フラグメント攻撃には、PoD（Ping of Death）、Teardrop、Fragmented IGMP があり、UDP フラグメント攻撃には、Too Big Fragment（過大フラグメント）、Tiny Fragment Attack（タイニーフラグメント攻撃）がある。これら脆弱性型は、RFC 技術仕様を逸脱したフラグ、長さ、通信プロトコルでパケットを送り付ける手口が用いられ、この手口で送り付けられるパケットの発信元 IP アドレスは詐称されているため、攻撃者を特定することが困難である。

表 2.2　DoS/DDoS 攻撃の種類と手口

種類	手口
SYN Flood Attack、FIN Flood Attack[113]	攻撃対象の TCP ポートへ、同期（SYN）、切断要求（RST）または終了要求（FIN）パケットを大量に送り付け、ACK 応答待ち状態（数十秒）の接続の限界数を超過させる攻撃である。
ACK Flood Attack[114]	攻撃対象の TCP ポートへ、アクティブな TCP 接続がないにもかかわらず応答（ACK）パケットを連続して送り付け、攻撃対象の通信ログを溢れさせて他攻撃をカムフラージュするか、攻撃対象に不合理な ACK パケットを廃棄させ大量の接続拒否（RST）の返信処理をさせることで過負荷にする攻撃である。
Connection Exhaustion Attack[115]	攻撃対象の TCP ポートと大量のコネクションを確立するだけで、データ転送は行わず、攻撃対象の接続状態テーブルを枯渇させることで、新たな TCP コネクション確立不能にさせ、全てのサービスを提供できなくする攻撃である。この攻撃は、ファイアウォールなどのセキュリティ装置のポリシーベースのアクセス制御において違反にならないため、それらの装置で遮断することは困難である。
Overlapping Fragment Attack[116]	攻撃対象の TCP ポートへ、TCP のオフセットフィールド情報を改竄し、TCP フラグメントのヘッダ情報を上書きしたデータを転送することで、ファイアウォールなどのフィルタリングで、以降の不正コードを伴うフラグメントの通過を許可させ、攻撃対象をクラッシュさせ、サービスを停止させる攻撃である。
LAND Attack[117]	攻撃対象の TCP ポートへ、攻撃対象に送信元と送信先を同じアドレスに指定した同期（SYN）パケットを送り付け、攻撃対象が同期応答（SYN/ACK）を自身に送信させる攻撃である。
ランダム・ポート・フラッド攻撃（UDP flood attack）[118]	攻撃対象のランダムな UDP ポートへ、最大長の UDP データグラムを大量に送り付け、攻撃対象の閉鎖ポートから大量の ICMP 応答（Destination Unreachable）処理をさせ過負荷にする攻撃である。

表 2.3　WWW/DNS/NTP フラッド攻撃

種類	手口
HTTP GET/POST Flood Attack [119]	攻撃対象の Web サーバへ、HTTP GET リクエストの URL にランダムなパスなどを含めたパケットを大量に送信して、攻撃対象においてリクエストが正常か攻撃かの判断処理を過負荷にすることで、リプライの処理を滞留させて、攻撃対象の Web サービスを遅延や停止させる攻撃である。
Slow HTTP DoS Attack[120]	攻撃対象の Web サーバへ、長大な HTTP GET リクエストヘッダを含めたパケットを、待機時間を挟みながら、断続的に送信することで、攻撃対象の TCP セッションの数やメモリ空間を占有し、他のリクエスト処理を不能にする攻撃である。
DNS Flood Attack[121]	攻撃対象の DNS サーバへ、名前解決処理能力を超えた大量の無効な名前解決リクエストを含めたパケットを送信することにより、攻撃対象の名前解決キャッシュを枯渇させると共に、攻撃対象のトラフィックを過負荷にすることで DNS サービスを不能にする攻撃である。
DNS reflection Attack（DNS amplification Attack）[122]	攻撃対象の DNS サーバへ、送信元アドレスを攻撃対象自身の IP アドレスに偽装した DNS リクエストのパケットを送信することで、攻撃対象内でメッセージが折り返され、トラフィックが次々増幅され、攻撃対象のネットワークが過負荷になり、DNS サービスを不能にする攻撃である。この攻撃には、一般的に、膨大な数のボットが踏み台として用いられる。
NTP reflection Attack[123]	この攻撃には、一般的に、膨大な数のボットが踏み台として用いられる。攻撃対象の NTP サーバへ、断続的に送信元 IP アドレスを偽装した問合せを送り付け、NTP サーバに大きなサイズの応答を送信させて、攻撃対象のネットワークを過負荷にさせて NTP サービスを不能にする攻撃であるこれは。これは、NTP が通常 UDP を用いて通信を行うことから、容易に送信元 IP アドレスを詐称できるためである。

表 2.4　ICMP/IP フラグメント攻撃

種類	手口
PoD（Ping of Death)[108]	攻撃対象の IP アドレスへ、通常はパケットサイズが 64 バイトの ping（ICMP/ICMPv6）を、IP パケットのフラグメント化を悪用して 65,536 バイト以上のパケットにして送り付け、バッファオーバフローにさせたり、フリーズさせたり再起動させたりする攻撃である。
Teardrop[124]	攻撃対象の IP アドレスへ、IP パケットのオフセットフィールド情報を改竄し、IP フラグメントのオフセット値が重複するパケットを送り付け、攻撃対象でパケットを復元する処理で、クラッシュさせサービスを停止させる攻撃である。
Fragmented IGMP[125]	攻撃対象の IP アドレスへ、動画や音楽のストリーミングなど、同時に多数のホストへ同一のデータを送信する攻撃。マルチキャストのプロトコル（L3）である IGMP は、パケットのフラグメント化に対応できないことを悪用して、攻撃対象にフラグメント化した IGMP パケットを送り付け、攻撃対象をフリーズさせたり遅延させたりする攻撃である。

表 2.5　UDP フラグメント攻撃

種類	手口
Too Big Fragment（過大フラグメント）[126]	攻撃対象の UDP ポートへ、攻撃対象へ最大長 1500 バイトまたは超最大長の UDP データグラムを大量送信することでバッファをオーバフローさせる攻撃である。
Tiny Fragment Attack（タイニーフラグメント攻撃）[127]	攻撃対象の UDP ポートへ、攻撃対象へパケットの断片サイズを極小化して送信する攻撃で、攻撃対象や経路上の通信処理装置に大きく二種類ある。1 つは、ノードが完全なパケットに復元するために断片を長時間連続してメモリに大量滞留させられ資源を浪費させ、通信遅延や通信不能に陥らせる攻撃。他は、トラフィックのフィルタリングにおいて、パケットに不正なデータが含まれていてもフラグメントにより、個々の断片を見過させ通過させる攻撃である。

2.3.3　サイバー攻撃に対する従来の検知手法

サイバー攻撃に対する従来の主な検知手法には、サイバー攻撃の段階毎に、下記の手法がある。

探査段階における侵入口の探査の検知手法では、主にアプリケーションのログ解析やパケットキャプチャを含めた通信データ解析がある。侵入段階におけるなりすましの検知手法としては、主に検疫ネットワークがある。侵入段階と攻撃段階におけるマリシャスコードの検知手法としては、主にウイルス検知、侵入検知、異常検知がある。攻撃段階におけるサービス妨害の検知手法としては、主に各種ファイアウォールがある。これら検知手法を、OSI 参照モデルの階層で分類したものを表 2.6 に示し、以下の詳細を述べる。

- アプリケーションのログ解析とパケットキャプチャを含めた通信データ解析
  アプリケーションのログ解析は、サーバーやファイアウオールなどのログ収集とその解析によって、インシデントの発生を検知する手法である。ログを解析するためには、あらかじめシステムがログを出力するように設定しなければならない。アプリケーションの種類により、ログの出力形式、項目や種類が異なるため、ログの解析の担当者には、利用者および管理者としてのアプリケーションの知識、操作、運用、管理について一定以上の経験とセキュリティ対策の力量が必要である。また、複数アプリケーションにまたがるシステム全体のログ解析の担当者も必要である。パケットキャプチャを含めた通信データ解析は、ネットワーク通信のデータを収集、記録し、人的操作により検索したり解析したり、通信プロトコルの手順やフラグを追跡したり、ペイロードも表示したりすることが可能だが、不正データを検知することはできない。パケットキャプチャによっては、マクロプログラムにより攻撃や不正アクセスを検知することも可能であるが一般的ではない。
- 検疫ネットワーク
  なりすましとは、ネットワーク上のノードが認証やセキュリティ対策済みを偽って、システムやネットワークに接続しようとする行為のことである。検疫ネットワークでは、このなりすましに対して以下の検査を行い、その結果により不正を検知し隔離する。検査には、セキュリティポリシーによる認証

表 2.6　従来の検知手法の種類と OSI 参照モデルの各階層

層	探査段階	侵入段階		攻撃段階
	侵入口の探査の検知手法	なりすましの検知手法	マリシャコードの検知手法	サービス妨害の検知手法
7、6、5	アプリケーションのログ解析	検疫ネットワーク（認証スイッチ方式、認証ゲートウェイ方式)	ウイルス検知（コンペア法、チェックサム法/インテグリティチェック法、インテグリティチェック法、ヒューリスティック法)	アプリケーションゲートウェイ型ファイアウォール、ステートフルインスペクション型ファイアウォール、パーソナルファイアウォール
4、3	パケットキャプチャを含めた通信データ解析		侵入検知（IDS)、侵入防止（IPS)	サーキットレベルゲートウェイ型ファイアウォール
				パケットフィルタ型ファイアウォール
2、1		検疫ネットワーク（認証 DHCP 方式）	異常検知（システム監視、トラフィック監視、SNMP)	MAC アドレスフィルタ

及び検査、認証違反やポリシー違反の検査、ウイルス感染の有無の検査、セキュリティパッチ更新済検査、セキュリティ対策実施済検査がある。検疫ネットワークは、アクセス制御により認証スイッチ方式、認証ゲートウェイ方式、認証 DHCP 方式などがある。

認証スイッチ方式: 有線 LAN または無線 LAN 上のノードが、レイヤ 2 スイッチや無線 LAN アクセスポイントに接続する際に、IEEE 802.1X によるユーザ認証や Web 認証を行う方式で、違反を検知し

接続を拒否する。

認証ゲートウェイ方式: ネットワーク上の中間ノードであるルータやレイヤ 3 スイッチ、ファイアウォール、VPN ゲートウェイなどをトラフィックが通過する際に検査を行う方式である。前記の認証スイッチ方式と比較して、対応機器が少なく済むため、導入のコストを低く抑えることが可能である。

認証 DHCP 方式: DHCP サーバからノードへ IP アドレスを払い出す仕組みを利用した認証方式であり、一旦一時的なアドレスを割り当て、検査に合格した場合にのみ、通常に通信可能なアドレスを割り当てる。このノードからのネットワーク接続は、限られた範囲でのみルーティングを許可する。

- ウイルス検知

  ウイルス検知は、検知対象のデータや通信処理などのふるまいからウイルスを検出して、人に通知するものである。ウイルス検知は、一般的に利用者が直接操作する PC やスマートフォンなどのノード、メールサーバやゲートウェイサーバなどにおいて検出され、その利用者や管理に通知されるが、その他にハニーポットなどのモニタリングシステムや検疫ネットワークで検出して専門家に通知し、未知のウイルスの発見に利用される場合ある。ウイルス検知には以下の検知手法がある。

  コンペア法: ウイルス検査対象のファイルと真正な原本ファイルをバイナリで比較し、差異があればウィルスに感染していると検出する手法である。

  チェックサム法/インテグリティチェック法: 真正な原本ファイルの容量やハッシュ値を保管しておき、ウイルス検査対象のファイルの容量とハッシュ値を比較して、差異があれば感染していると検出する手法である。原本を保管する領域を確保する必要がないが、偽装は可能である。

  インテグリティチェック法: ディジタル署名を含めてウイルス検査対象のファイルの真正性を確認する手法である。

  パターンマッチング法: ウイルスの特徴的なコードのパターンについ

て、ウイルス検査対象と登録情報（パターン）であるパターンデータ、ウイルスパターン、パターンファイルやウイルス定義ファイルなどと比較して検出する手法である。特徴的なコードのパターンとして、ファイル名、メール送受信者の名前やアドレス、送信経路情報、メールメッセージの表題、テキスト文章、バイナリ情報などが含まれる。

ヒューリスティック法: ウイルスとして予測される動作を事前に登録しておき、ウイルス検査対象コードに含まれる一連の動作と比較して検出する手法である。プログラムの挙動をルールベースで静的に解析する。

ビヘイビア法: ヒューリスティック法の 1 つとして、ウイルスの実際の感染・発病動作を、ルールとして登録しておく。その上で、ウイルス検査対象プログラムを実際に実行させて動作をルールベースで監視し、危険な挙動を検知する手法である。対象プログラムの実行環境として、実マシン環境と仮想環境がある。検知する挙動は、感染・発病動作そのものと感染・発病動作によって起こる環境の様々な変化の 2 種類がある。前者には、書込み動作、複製動作、破壊動作の危険な動作、後者には、例外ポート通信・不完パケット・通信量の異常増加・エラー量の異常増加、送信時データと受信時データの量的変化・質的変化などがある。

- 侵入検知（IDS）/侵入防止（IPS）

  ネットワークやホストへの侵入や攻撃を検知するシステムとして、パターン認識技術を利用した侵入検知システム（Intrusion Detection System）があり、検知後に侵入や攻撃を防止する侵入防御システム（Intrusion Prevention System）がある。パターン認識技術は、「観測」「前処理」「特徴抽出」「特徴変換」「識別」の処理過程から成り立っている [128]。IDS の参照モデルとして CIDF （The Common Intrusion Detection Framework）[129] が提案されており、前述のパターン認識技術の処理過程に対応する機能を以下に挙げる。

  観測: データ収集機能（Data Collectors）

前処理: データ作成機能（Data Generators）

特徴抽出: データ解析機能（Data Analyzers）

特徴変換: アクション機能（Response Units）

識別: 記録機能 (Event Database)、ログ解析機能 (Log Analyzers)

検知の型として、以下がある。

不正を定義するシグネチャ型検知では、あらかじめアクセスを不正とするポート番号、マリシャスコードなどを含む不正な文字列や異常行動パターンを定義しておき、この定義と検知対象が一致する場合を異常と検知する。例として、FTP（TCP のポート番号 21 番）通信で、Guest というユーザー・アカウントによるアクセスを不正と判断するシグネチャは、「SELECT Protocol[TCP] AND Port[21] WHERE Contains["Guest"]」と定義する。他の例として、HTTP（TCP のポート番号 80 番）通信で、ディレクトリトラバーサルを不正アクセスと判断するシグネチャは、「SELECT Application[HTTP] WHERE Contains[../..]」と定義する [130]。一般的に、このようなシグネチャが数百から数千種類程度定義され、データベースに登録される。

正常を定義するアノマリ型検知では、あらかじめ正常な値の範囲や行動パターンをルールとして定義しておき、この定義以外の検知対象をすべて異常として検知する。例として、パスワードの入力フィールドの正常データは、キーボードから入力できる ASCII 文字のみとするルールを定義すると、ASCII 文字以外の文字の入力はすべて異常だと検知する。

検知システムの導入場所として、ネットワーク上で通過する全てのトラフィックをキャプチャして監視する NIDS（Network-Based IDS、ネットワーク型 IDS）と、ホストに常駐して自ホストへの侵入のみを検知する HIDS（Host-Based IDS、ホスト型 IDS）がある。ネットワーク型 IDS はホスト型 IDS に比べて導入が簡単であり、リアルタイムで検知できる可能性が高いことがメリットであるが、トラフィックが増大してキャプチャが過負荷になると、トラフィックのボトルネックとなる危険性が高くなることがデメリットである。

- 異常検知

異常検知には、システム監視やトラフィック監視などがある。

システム監視は、コンピュータや通信処理装置のシステムリソースである CPU 使用率、メモリ使用率、ディスク使用率、デバイスの使用率、ネットワークインタフェースの通信データ量などの異常を検知する。

トラフィック監視は、トラフィック容量、トラフィック速度、パケット数、トップトーカー（通信対象）、帯域使用率などを観測し、ネットワークの性能低下、応答遅延や停止などの異常を監視する。例えば、通信処理装置におけるトラフィック監視では、装置自身の各ポート別に送受信されたパケット数、エラーパケット数、ポートの状態（up/down）、及び CPU 使用率、メモリ使用率などの異常を監視するが、装置に接続するノードの特定プロセスの死活の異常も監視する場合もある。

ただし、システム監視やトラフィック監視では、時刻、曜日、期、季節などの時系列で傾向分析も含めて異常を監視する必要がある。例えば、自他組織間のネットワーク接続形態変更、業務上の繁忙期/閑散期、システム（サービス）変更に伴うノード毎の種類や規模の変更、組織変更に伴うネットワークの再構築や機器の入替作業による変動の傾向や異常は、攻撃や不正アクセスによる異常と区別する必要がある。

実際のシステムやネットワークの異常検知のツールとしては、主に sar コマンドや SNMP（Simple Network Management Protocol）[131] を用いたモジュールがある。前者の sar は、Linux 系 OS において汎用的なコマンドで、監視情報をログファイルに記録するだけで軽量である。後者の SNMP は、監視情報をセンシングする SNMP エージェントと監視情報の解析を行う SNMP マネージャ間の通信プロトコルで、その監視情報は管理情報ベース（Management Information Base）に蔵置される。監視情報はポーリングとトラップにより収集される。ポーリングは、定期的に SNMP マネージャが定期的に SNMP エージェントから管理情報を収集する機能で、トラップは、エージェント側で何かの異常が発生した場合に SNMP マネージャにその情報を通知する機能である。異常検知によるイベント検知については、2.3.4 節で述べる。

- ファイアウォール

ファイアウォールは、信頼性の異なるネットワーク毎に求められる安全性を維持することを目的とした技術概念またはフィルタリング技術であり、攻撃や不正アクセスを検出し制御する。ファイアウォールは、構内ネットワークのように機密性の高いネットワークを、インターネットのようなオープンなネットワークから分離する。また、ファイアウォールは、インターネットからの攻撃防止だけでなく、構内ネットワーク内部からの情報の漏洩防止にも用いられる。ファイアウォールを実現するには、ハードウェア（専用ファイアウォール）やソフトウェア（ルータ、Proxy やサーバのファイアウォール機能）を利用する。ソフトウェアには、専用のファイアウォール・プロダクト、OS（ファイアウォール機能）やアプリケーション・プログラムのモジュール（構成部分）がある。パケットの通過を許可するか拒否するかの判断は、静的または動的に定められたフィルタリング・テーブルに定義されるフィルタリング・ルールに基づく。ファイアウォールの主機能として、アクセス制限（権限管理）、アドレス変換（内部アドレス情報の隠蔽）、ユーザ認証（利用者制限）、ログ収集／解析（インシデントの発見と原因分析）、コンテンツフィルタリング（データの内容検査）やルーティング（データの通信経路管理）などがある。また、フィルタリングする通信方向として、内部から外部方向へのパケット（メッセージ）と外部から内部方向へのパケット（メッセージ）の別に定義する。ファイアウォールの種類として、OSI 参照モデルのレイヤ（階層）毎に以下のものがある。

レイヤ 7〜5: アプリケーションゲートウェイ型ファイアウォール （プロキシ型型ファイアウォール）内部ネットワークと外部ネットワークを直接通信させず中継するプロキシサーバの仕組みを持つファイアウォール。これにより、通信データの中に既知の不正コードやウイルスが含まれていれば遮断する。

レイヤ 7〜5: ステートフルインスペクション型ファイアウォール アプリケーション毎の通信フローのステート（セッションの状態）も認識するダイナミックパケットフィルタリング（パケットフィルタリング型ファイアウォールを参照）を行うファイアウォールである。

レイヤ 7〜5: パーソナルファイアウォール　主にパーソナルコンピュータにおいて、外部ネットワークから内部への侵入、および内部からの外部ネットワークへの通信を許可または遮断する。

レイヤ 4〜3: サーキットレベルゲートウェイ型ファイアウォール　（トランスポートゲートウェイ型ファイアウォール）ファイアウォールが内部ネットワークから外部ネットワークへの接続要求を受け取りセッションログとして保管しておき、外部ネットワークに対してファイアウォールがコネクションを確立し、内部ネットワークと外部ネットワークを結ぶ仮想的な通信路を確立する。TCP、UDP、IP などの下位の通信プロトコルの規約に従ったパケットは許可し、矛盾や違反したパケットおよびなりすましのパケットは遮断する。アプリケーションゲートウェイ型と似ているが通信データは確認しない。

レイヤ 3: パケットフィルタリング型ファイアウォール　2 つのフィルタリングを行う。1 つは、外部から内部へ不要、不正なパケットを流入させず、内部から外部へセキュリティルール違反のパケットを送出させないスタティックパケットフィルタリングである。もう 1 つは、内部から外部への要求パケットをスタティックパケットフィルタで許可し送出した後、履歴をフィルタリング・ルールに追記しておき、外部から内部への応答パケットと照合して許可または遮断するダイナミックパケットフィルタリングである。

レイヤ 2: MAC アドレスフィルタ　無線アクセスポインタや有線スイッチなどのネットワーク機器に、予め登録しておいた MAC アドレスによりネットワークへのアクセス制限を行う。

2.3.4　異常検知手法によるイベント検知

2.2 節で述べた IoT セキュリティガイドラインでは、機器等がどのような状態かを把握し、記録する機能を設けることを求めている。よって、論文による IoT デバイスの検知手法の提案においても、機器等がどのような状態かを把握して、攻撃や不正アクセスのイベント検知を検討する。そのような手法としては異常検知が最も

近いため、本節では、異常検知に関して、検知対象となる時系列データ、異常検知の統計手法、異常検知のデータ解析に用いられる予測モデル、自己回帰モデルに用いられる統計手法と変化点検知の手順、異常検知で汎用的に用いられるシステム監視について述べる。

異常検知は、異常な値や変化を検出する技術 [132, 133, 134] である。検知対象データとして、数値データ、音声、静止画、動画、文字やコンテキストなどがあり、時系列データと時系列ではないデータである場合がある。時系列データは、各時刻の観測順並べられたデータであり、時系列ではないデータは、時間の経過に影響を受けない独立なデータである。時系列データにおいて、異常（Anomaly）の種類には、Error（誤り）、Outlier（外れ値）、Deviation（変動）、Noise（ノイズ）、Fraud（不正）、Intrusion（侵入）、Fault（過失）、Defect（故障）、Novelty（新規）、Rare（稀有）、Exception（例外）などがある。時系列データの変動要因として、傾向変動（トレンド）、循環変動（サイクル）、季節変動（シーズナル）及び不規則変動の 4 つが挙げられる。時系列データにおいて、異常を発生させる要因は、天災、天候不順や電磁嵐などの環境的脅威、人的錯誤、不正や犯罪行為などの人的脅威、故障、障害や品質劣化などの物理的脅威などがある。さらに、時系列データの異常検知に用いられる検知技術として、データ収集のためのセンシング技術、統計解析、パターン認識、人工知能（AI: Artificial Intelligence）などのデータ解析の技術、ビックデータを利用した未知の規則性や傾向を採掘するデータマイニングなどがある。時系列データにおける異常検知の検知単位として、外れ値、異常部位、変化点がある。外れ値は、正常時には発生しないデータ点が検出単位である。異常部位は、異常が発生している部分時系列が検出単位である。変化点は、時系列データのパターンが急激に変化する時点が検出単位である。時系列データの異常検知として、外れ値だけでは、前記の異常のいずれに当たるか判断できないので、インシデントに繋がるかは判断できない、また、外れ値が連続する異常部位だけは、ユーザ処理によるものか侵入・攻撃によるものか判断できない。時系列データの異常検知において、異常は、外れ値と異常部位の検知に加えて、変化点の検知により判定される。

異常検知の統計手法として、検知対象の多次元データ分布のデータから外れ値を検出する検知（Outlier detection）[135] がある。この外れ値は、距離（最近傍法、k 最近傍法や局所部分空間法 [136],134 など）、密度（LOF（Local Outlier Factor）

[137]、Random Forest[138] など)、角度(ABOD(Angle-Based Outlier Detection)アルゴリズム [139] など)、統計解析(統計的検定 [140]、マハラノビス＝タグチ法(MT 法)[141])やその他(1 クラス SVM(Support Vector Machine)[141] や情報量 [142])などから判定する。

セキュリティ・サイバー攻撃における異常検知の手法として、静的検知手法と動的検知手法がある。

静的検知手法には、外れ値による検知として不正アクセス、侵入や障害発生の検知など、変化点による検知としてネットワーク攻撃、ワーム感染や攻撃予兆の検知など、ビヘイビア法 [143] による異常行動やなりすましの検知などがある。

動的検知手法には、検知対象の多次元時系列データで異常が発生した変化点を検知する方法(Change point detection)[144] と、検知対象の部位時系列から異常動作や異常行動の状態を検知する方法(Anomaly detection)[145] として、前述の自己回帰モデルがある。異常検知のデータ解析として、異常部位の前後の変化点を用い、ホテリング理論 [146] を応用した“予測モデル [147]”による技術がある。

予測モデルとして、主に、AR モデル(Autoregressive Model: 自己回帰モデル)と MA モデル(Moving Average Model: 移動平均モデル)の 2 つのモデルがある [148]。これらモデルを組み合わせて応用したモデルやマルコフ連鎖を利用したモデルとして、ARMA モデル(Autoregressive Moving Average Model: 自己回帰移動平均モデル)[149]、ARIMA モデル(Autoregressive Integrated Moving Average Model: 自己回帰和分移動平均モデル)[150]、EC モデル(Error Correction Model: 誤差修正自己回帰モデル)、ARCH モデル(Autoregressive Conditional Heteroscedasticity model: 分散自己回帰モデル)、GARCH モデル(Generalized Autoregressive Conditional Heteroscedasticity Model: 一般化分散自己回帰モデル)、SV モデル(Stochastic Volatility Model: 確率的ボラティリティモデル)、MSM モデル(Markov Switching Model: マルコフスイッチングモデル)、MSM モデル(Markov Switching Multifractal Model: マルコフスイッチングマルチフラクタル)がある。

AR/MA/ARMA/ARIMA モデルより複雑な時系列モデルを構成するために、SS モデル(State Space Model: 状態空間モデル)がある。このモデルの基本的な手法としてカルマンフィルタ(Kalman filter)があり、これは逐次ベイズフィルタ

の一種である。状態空間モデルは、観測できない隠れた状態モデルと、観測した結果である観測モデルから構成される。

時系列データの予測モデルである、自己回帰モデルと状態空間モデルを以下に比較する。自己回帰モデルである AR/MA/ARMA/ARIMA モデルの利点は、周期性があるデータの異常検知に向いており、異常検知以外にも時系列予測、変化点検知に使用可能であることである。欠点は、次数が固定されるため状態が一定なデータでないと異常検知として利用することが難しいことである。状態空間モデルの利点は、状態が変化しても対応可能かつノイズに強く、自己回帰モデルと同様に時系列予測、変化点検知に使用可能であることである。また、観測できない状態を構造に組み込むことで、時系列予測をさまざまな要因分解の結果として行なえるため、時変パラメータの解釈や可視化が容易である。欠点は、周期性があり安定しているデータには対応しづらいことである。自己回帰モデルの汎用的な統計手法として、移動平均法、指数平滑化法がある。移動平均法は、MA モデルを利用した ARMA モデル（自己回帰移動平均モデル）で、季節変動や不規則変動の変動要因の影響を取り除くために、時系列データを平滑化して、傾向を明らかにする手法である。指数平滑化法は、移動平均法と同様に時系列データを平滑化するが、時間軸で現在に近いデータほど重要視し、過去にさかのぼるほど重要度を落としていく加重平均法を行いる。

異常検知によるイベント検知として、検知したイベントがインシデントに紐づくか判定するためには、アーカイブデータには、検知対象が平時の場合のデータだけでなく、検知対象が異常時の場合のデータも含まれる必要である。例えば、各システムリソースに対して基準値や閾値を設定する場合、CPU 使用率の基準値を 50 %として、その前後 20% 内であれば正常とし、CPU 使用率の閾値を上限 70 %として、それを超えると異常とする。ただし、これら基準値や閾値は、開発や導入時に一義的に固定されるのではなく、正常時と異常時利用法の場合でシステム監視を行い、見直し調整する必要がある。実際に、システム監視を利用してイベント検知を行う場合は、検知対象とアーカイブデータのシステムリソースのデータや統計データを比較して判定する。

2.3.5 本論文で検知対象とする攻撃

本論文では、検知対象とする攻撃として、以下の 3 種類を想定する。

- DDoS 攻撃

  攻撃対象のシステムやネットワークを過負荷にして、それらを機能不全にする [151]。そのために、ボットネットなどから大量のパケットを転送する。

- 不正アクセス

  攻撃対象システム内に侵入して、情報を改竄、漏洩したり、不正利用したりする [152]。不正アクセスは、一般的に 2 段階で行われる。前段では、攻撃対象候補のネットワーク内の全 IoT デバイスに対してポートスキャンを行い、TCP/IP、OS やアプリケーションの脆弱性を探査する [153]。このために、脆弱性試験ツールが悪用されることが多い。この時、大量の問い合わせとその応答の通信データが発生する。後段では、前段で発見された脆弱性を利用して攻撃を実行する。

- 踏み台攻撃

  DDoS 攻撃と同様に膨大な数のボットから攻撃対象を攻撃するが、攻撃元の追跡を攪乱させ、被害者側での対応や回復を阻害したり不能にしたりする。例としては、DNS サービスを悪用した DNS リフレクター攻撃 [154] や、Web サービスや Mail サービスへの踏み台攻撃 [155] などがある。上記の各攻撃において、イベントとインシデントの具体例は以下のようになる。DDoS 攻撃の場合、悪意のある大量のデータが攻撃対象に送信されることで、攻撃対象の受信処理や応答のための送信処理、システム処理の負荷が高くなることがイベントであり、サービス妨害が発生し、サービスが遅延し、使用できなくなる事件がインシデントに相当する。不正アクセス攻撃の場合、脆弱性を探査する要求が送信されることで、受信した要求に応じて送受信処理やシステム処理が発生し、それらの負荷が高くなることがイベントであり、それにより引き起こされる情報漏えい事件がインシデントに相当する。踏み台攻撃の場合、検知対象側では、大量のデータの送信処理や応答のための受信処理、システム処理の負荷が高くなることがイベントであり、それらは攻撃対象側でのインシデントの原因となる。ここで、イベントは、

ユーザやシステムの処理によっても発生するものであり、全てのイベントがインシデントに至るとは限らないことに注意する必要がある。

2.4 従来の検知手法の問題点と分類

2.4.1 従来の検知手法の問題点

- アプリケーションのログ解析やパケットキャプチャを含めた通信データ解析の問題点

  本検知手法が不足しており、専門家が常時ログを監視し、解析することは現実的でないこと。

- 検疫ネットワークの問題点

  検疫ネットワークには、方式毎に、下記の問題点がある。認証スイッチ方式や認証ゲートウェイ方針では、認証を行うスイッチやゲートウェイを通過しない通信は、認証を受けなくてもアクセスが可能となってしまうという問題点がある。パーソナルファイアウォール方式では、パーソナルファイアウォールがインストールされていないクライアントは、検査を受けなくてもアクセスが可能となってしまうという問題点がある。認証 DHCP 方式では、DHCP を利用していない環境では導入できなく、静的にアドレスを設定されてしまった場合には効果がないという問題点がある。また、IoT デバイスに公開鍵暗号基盤（PKI）や SSL/TLS 電子証明書によるユーザ認証機能を対応させることは可能だが、現状の製品には普及していない。

- ウイルス検知の問題点

  ウイルス検知には、手法毎に、下記の問題点がある [156]。コンペア法とチェックサム法/インテグリティチェック法では、検査対象が未感染の状態から感染状態に変化したことを検出するため、原本、チェックサム、ディジタル署名など未感染の状態を事前に保存していない検知対象に対してウイルス検知はできず、また、検知できたとしても、検知対象のプログラムが感染したウイルスが、検知対象内に既に侵入している可能性がある。パターンマッチング法では、検知対象と既知のウイルスの情報と照合してウイルスを検知するため、新種や亜種などの未知ウイルス、OS の機能の一部になすまし自身を隠蔽するウイルス、感染のたびにコードが変化するウイルスは検知

できない。ヒューリスティック法では、検知対象がウイルスらしい行動を起こすかプログラムコードを静的な解析により判断してウイルスを検知するため、暗号化されたウイルス、OS の機能の一部になすまし自身を隠蔽するウイルス、感染のたびにコードが変化するウイルスは検知できない。ビヘイビア法では、ウイルスの振る舞いを監視するため検知対象プログラムを実際に実行する必要があるため、被害を発生させてしまう危険性があること、ウイルスが検知環境を識別してウイルスの機能を停止させて検知させない可能性がある。

- 侵入検知（IDS）/侵入防止（IPS）の問題点

  IDS/IPS には、型を問わず、下記の問題点がある。CPU、メモリ、ハードディスクなどのシステムリソースを大量に消費する。一般的に、暗号化通信に対応できず、なりすまし攻撃にも対応できない。主にネットワーク層を対象に不正アクセスを検知・遮断するため、Web などのアプリケーション層のに対する攻撃には対応することができない。シグネチャ型では、パターンファイルに定義されていない異常は検出できなく、新たな攻撃手法が出現した場合に対応するシグニチャがリリースされるまでには数日が掛かりその間は検出できない。また、シグネチャに関する誤報が多いため監視運用を混乱させ、広帯域のネットワークでは大量のパケットに対して処理不能になる場合がある。アノマリ型では、実運用中のネットワークにおいて、常に正常なトラフィックのみを学習させることは、現実的に困難であることなどから、十分な検出精度を実現することが困難である。

- 異常検知の問題点

  2.3.4 節で述べたように、システム監視、ネットワーク監視、SNMP を利用した異常検知は、専門家がログデータを統計解析や波形解析を前提とするため、自動的にイベントを検知して利用者に通知する機能はなく、被害に至ってしまう危険性が高い。また、異常検知の結果が、インシデントに繋がるイベントであるか判定するためには、過去の正常時と異常時のログデータと比較検討する必要がある。正常時のデータは、日時、月、四半期、年単位で保存する必要があり、また異常時のデータは、不正アクセスや攻撃を受けた際に保存する必要がある。よって、ログデータの保存領域を大量に消費する問

題がある。

一般的に、SNMP と MIB を組み合わせることにより、遠隔から監視対象機器のシステム資源や通信データに関する設定や状態の情報を収集してデータベースに蓄積することは可能になるが、実際に機器の監視し、異常を解析したり検知する機能は提供されない。そのため、SNMP と MIB をサイバーセキュリティ対策の異常検知に利用するためには、別途に監視用ツールや解析プログラムを導入して、専門家が解析する必要がある。SNMP によるモニタリングでは、SNMP エージェントで異常が発生しなくても、デフォルトとして 5 秒間隔で SNMP マネージャとの間で情報の要求と応答の通信データが発生するため、SNMP マネージャは SNMP エージェント数の増加に伴いシステム負荷は重くなり、トラフィック量も増大する問題点がある。また、SNMP は下位層で UDP を用いるため、通信パケットの送信元アドレスの偽装やデータの改竄などの危険性が高いなどの問題点もある。

IoT デバイスを SNMP エージェントとして、その情報を SNMP マネージャに通知することは可能である。しかし、IoT デバイス自らが SNMP マネージャとなり異常検知を行うことは、システム負荷の問題や、表示機能の制約から監視用ツールを利用できないなどの問題から、現実的ではない。さらに、IoT デバイスが、攻撃や不正アクセスを受けた場合、システムが過負荷になり、通信データ量が急増するため、SNMP マネージャへの通知が困難または不能になる。仮に、SNMP マネージャは監視対象の IoT から情報を収集できたとしても、事件事故の予兆をリアルタイムにイベントとして検知する機能はなく、事件事故の発生後のインシデントの原因解析の機能に限定される。よって、IoT において SNMP と MIB をサイバーセキュリティ対策のイベント検知として利用することは、やはり現実的でないと言える。

- ファイアウォールの問題点

  ファイアウォールには、型を問わず、トラフィックの増大やサービス妨害が発生した場合に、ログファイルや異常通知メールが増大しディスク容量がオーバフローしたり、ファイアウォール自身が過負荷になったり、ファイアウォール自身が保護すべきネットワークにとってボトルネックになったりする問題点がある。Web サイトやメールサーバなどの公開サーバが存在す

る非武装地帯のネットワークのファイアウォールでは、外部から公開サーバへのアクセス制限は緩やにする必要があるため、非武装地帯のネットワークを経由して保護すべきネットワークに侵入される危険性が高くなる。パケット・フィルタリング型ファイアウォールでは、通信パケットのヘッダ情報のみでアクセス許可・拒否を判断するため、ヘッダ情報のアクセス許可の条件さえ満たせば、データ領域に不正なプログラムコードやデータが隠蔽されていても拒否しないので、ファイアウォールとして保護すべきネットワーク内に侵入されてしまう。アプリケーションゲートウェイ型ファイアウォールでは、そのハードウェアやソフトウェアに内在するバグやセキュリティホールなどの脆弱性を、攻撃や不正アクセスされると、ファイアウォール機能が無効化され、保護すべきネットワークが攻撃や不正アクセスされてしまう。

2.4.2 従来の検知手法の分類

従来の検知手法と IoT のイベント検知手法を、システムリソースの消費量と静的/動的解析の観点で分類し、図 2.2 に示す。以下にそれらの詳細を述べる。

検疫ネットワーク: 検疫ネットワークの本来の目的は、ユーザ認証やネットワーク認証であるため、検知として独立しており、システムリソースはあまり消費しない。検知は認証時のみであるため、静的である。

ウイルス検知: ウイルス検知の各方式では、パターンファイルの蔵置のディスク容量を、特にヒューリスティック法とビヘイビア法では、CPU パワーとメモリ領域を消費する。また、パターンマッチング法はコードを受け取ったときに検知するので静的であり、ヒューリスティック法とビヘイビア法は、検知対象の動作を継続に検知するので動的である。

IDS/IPS: 検知プログラムを稼動させるため CPU パワーとメモリ領域を大量に消費し、特にネットワーク型は接続するネットワーク上の全てのデータが検知対象となりネットワークインタフェースの送受信データが大量になる。ホスト型は自ノードの検知対象データの送受時に検知するので静的であり、ネットワーク型はネットワーク上の全ての検知対象データを通信プロトコルに従って時系列的に検知するので動的である。

ファイアウォール（FW）: 各型の違いにより CPU パワーとメモリ領域

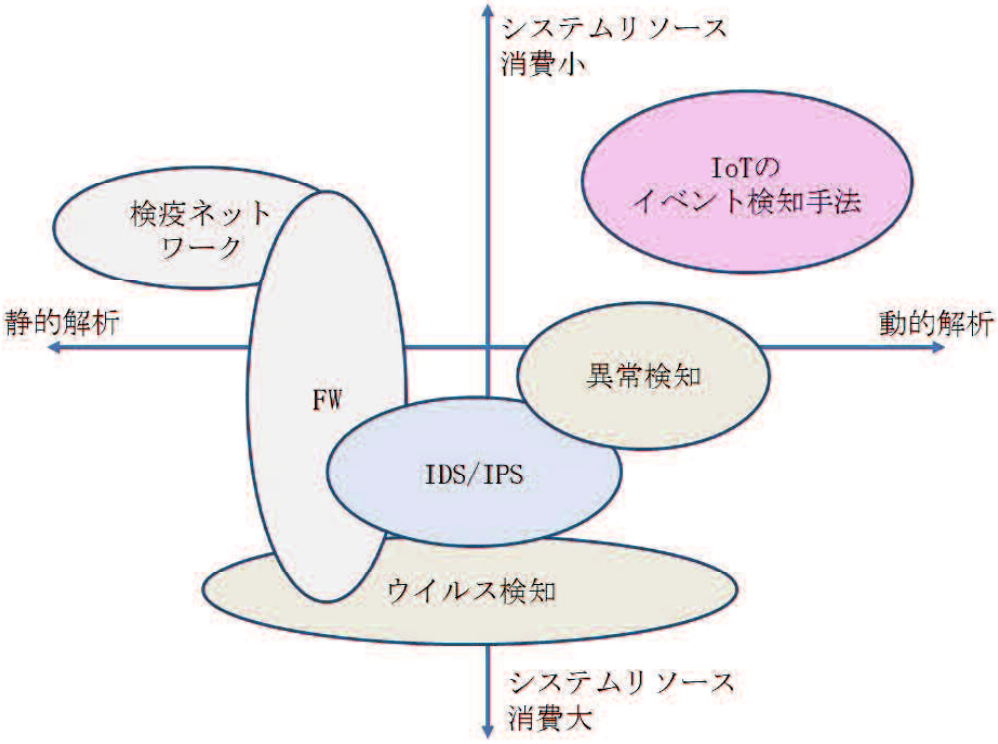

図 2.2　従来検知手法と IoT のイベント検知手法の位置づけ

を消費する量は異なる。パケット・フィルタリングは自ノードの全通信データが検知対象となり解析のために CPU パワーとメモリ領域を大量に消費し、サーキットレベルゲートウェイ型は中継する全ての通信データが検知対象となりネットワークインタフェースの送受信データが大量になる。同検知は、パケット単位またはメッセージ単位で送受時に解析し検知するため、静的である。

異常検知:　いずれの手法も共通して、ログファイルを蔵置するためのディスク容量を消費する。検知対象データを時系列的に解析するため、動的である。

IoT デバイスのイベント検知手法:　IoT デバイスでは、システムリソースをできるだけ消費しない必要がある。また、リアルタイムに検知する必要があるため、動的である。

以上のように、従来の検知手法と IoT のイベント検知手法の間には大きな乖離がある。

2.5　IoT のイベント検知に対する要件

IoT のイベント検知に対する要件について、これまで述べてきた従来の検知手法の種類、内容及び問題点を踏まえて、以下にまとめる。

- 検知精度が高いこと
- 検知手法は、イベントのインシデント判定は短時間で行えること
- 検知手法は、暗号化通信にも対応できること
- IoT デバイスを検知対象とするため、検知モジュールの容量や稼働時のシステム負荷（CPU、メモリ、ディスク、ネットワーク）が、軽量であること
- 定期的な更新や保守作業を前提としないこと

2.6　IoT におけるシステム監視による異常検知

IoT のイベント検知の要件を満たし得る手法として、2.3.4 節で述べた、システム監視による異常検知がある。ここでは、IoT デバイスにおいて、平時と疑似攻撃を与えた場合に、システム監視を行い、その監視ログからイベントを検出する予備実験を行い、有効性や問題点を検証する。なお、予備実験の詳細については、付録 A で示し、本節では実験の概要と、結果および考察のポイントみを述べる。

予備実験では、IoT デバイス上で、システム監視には sar コマンドを用い、システムリソースの使用率やデータ量を収集し、移動平均法を用いて外れ値、異常部位、変化点からイベントを検出する。システムリソースの要素は、システムとユーザの CPU 使用率、物理メモリの使用率、ネットワークインタフェースの 1 秒あたりの受送信データサイズ、ディスク入出力の 1 秒あたりの読み聞き込み入出リクエスト数の 6 種類とする。平時の場合として、明示的なユーザ処理が無い場合と、Web カメラ監視の場合の 2 通りを考える。疑似攻撃を与えた場合として、検知対象の IoT デバイスのシステムに接続して不正にリモートアクセスを行う場合、ネットワーク過負荷の場合、システム過負荷の場合の 3 通りを考える。

予備実験の結果を、図 2.3〜図 2.6 に示す。システムが利用している CPU 使用率は、平時の場合として、Web カメラ監視の場合は約 15%〜約 70% で、その観測データと線形近似を図 2.3 に示す。検知対象の IoT デバイスに疑似攻撃を加えた場

合として、システム過負荷の場合は数 %～約 60% で、その観測データと線形近似を図 2.5 に示す。考察として、システム CPU 使用率の外れ値、異常部位、変化点について、観測データの移動平均値と線形近似として、Web カメラ監視の場合は図 2.4 に示し、システム過負荷の場合は図 2.6 に示し、イベントの発生について、観測データと観測データの移動平均値の基本統計量（表 付録 A.3、A.4、A.5、A.6 を参照）における変動と分散の値からも示す。以上のことから、CPU 使用率の値は、Web カメラ監視の場合の方がシステム過負荷の場合よりが高く、外れ値、異常部位、変化点については両場合は近似しており、攻撃や不正アクセスではない Web カメラ監視の場合を、インシデントに繋がるイベントと誤検知する可能性が高い。

また、実験結果としてのログファイルの記録によるシステムリソースの消費量について、以下に考察する。本予備実験の 1 回の実験は 20 分間で、その際に記録されたログファイルの平均容量は約 10MiB であった。これは、単純計算で、1 日 24 時間で約 720MiB、1 週間で約 5GiB となる。また、イベントの判定として、検知対象のログファイルと正常時や異常時のアーカイブファイルと比較する必要があるため、これらのファイル容量も確保する必要がある。しかしながら、本予備実験で使用した Raspberry Pi 3 model B では、一般的に補助記憶デバイスとしてマイクロメモリを使用するが容量は 4GByte から 32GByte であり、長時間のシステム監視を運用することは困難がある。この代替策として、IoT デバイスからログファイルを Syslog サーバ [157] に転送する方法があるが、その際、Syslog 用の通信パケットが大量に発生して逆にネットワークを過負荷にする可能性があり、またログファイルのデータを盗聴される可能性もあり、採用することは困難であると言える。

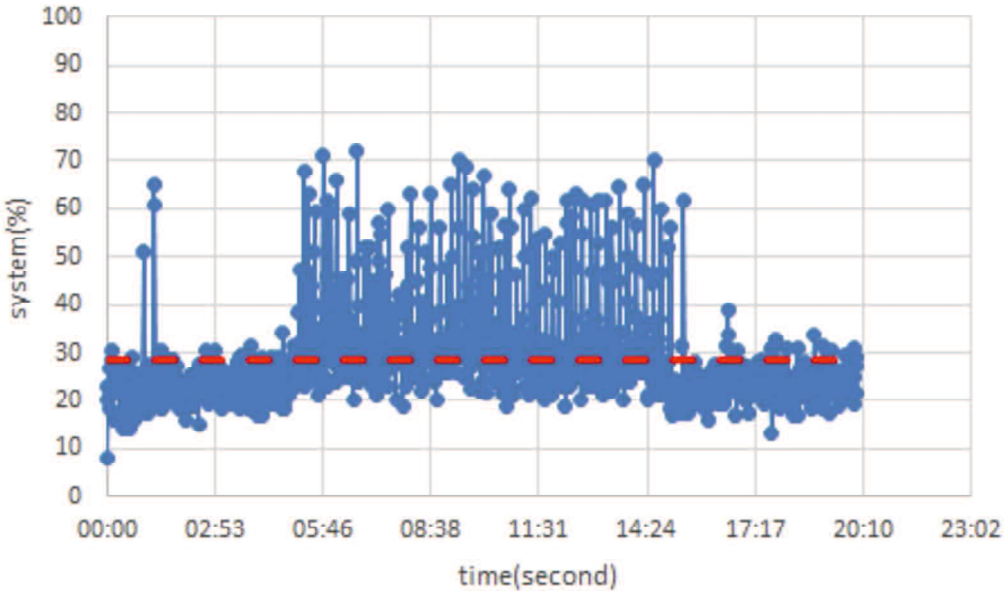

図 2.3　Web カメラ監視におけるシステム CPU 使用率

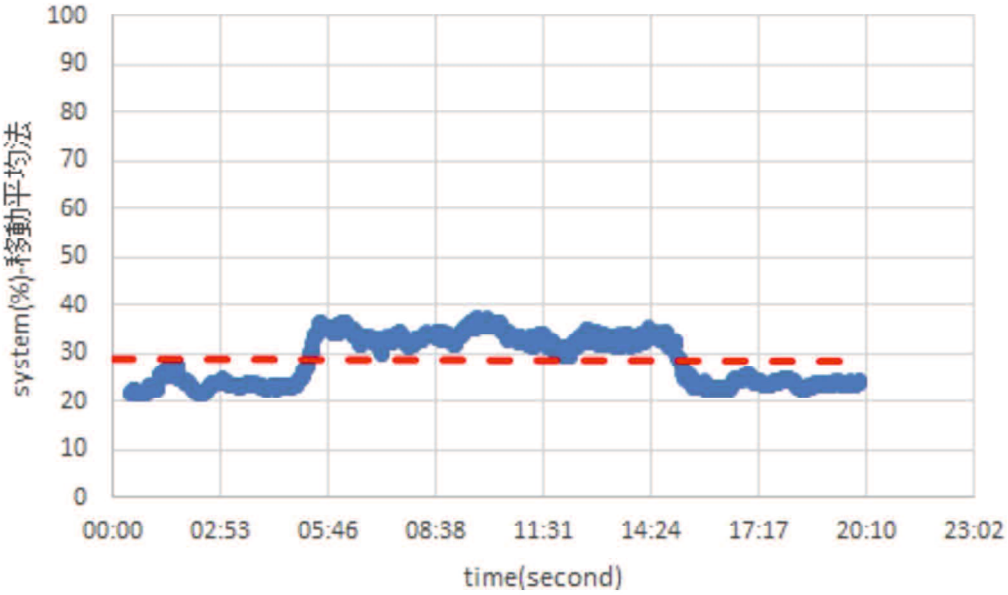

図 2.4　Web カメラ監視における移動平均法によるシステム CPU 使用率

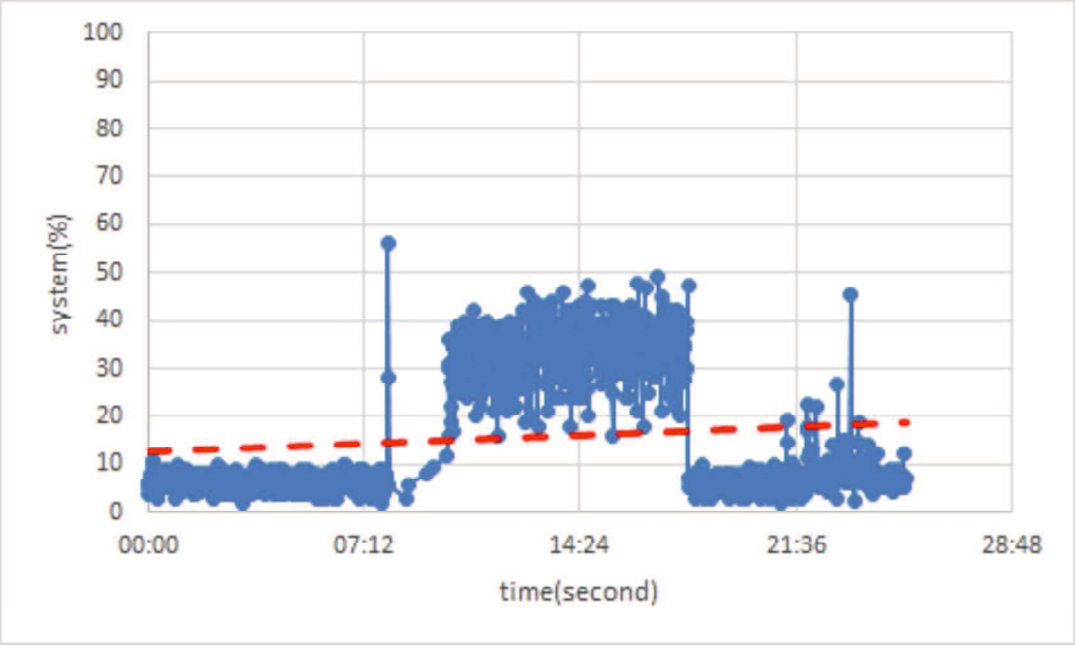

図 2.5　システム過負荷時のシステム CPU 使用率

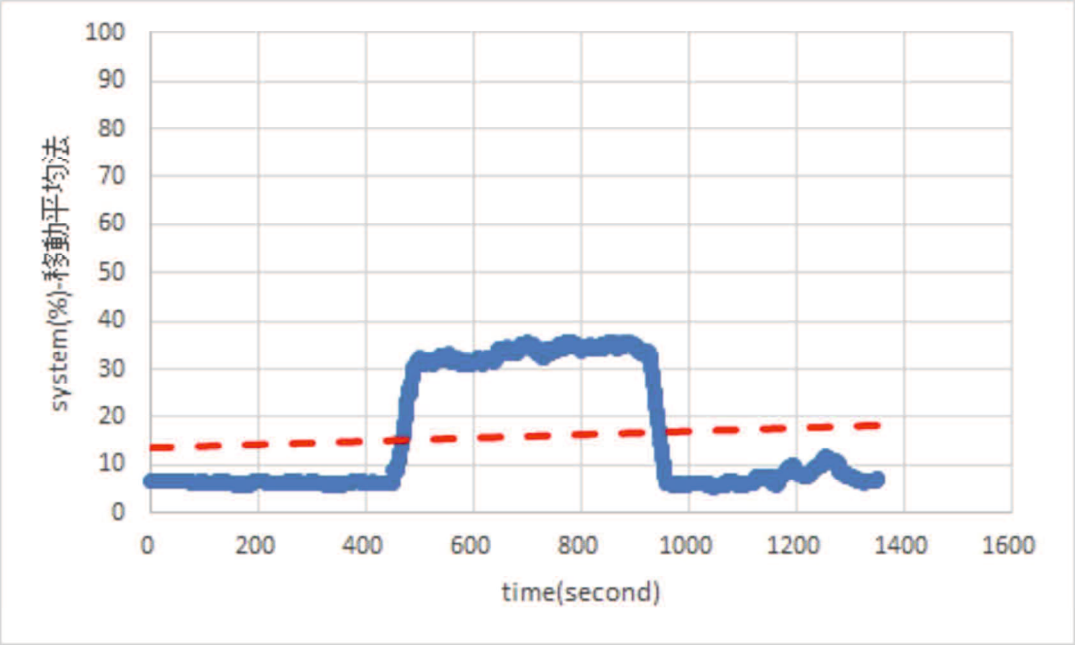

図 2.6　システム過負荷時の移動平均法によるシステム CPU 使用率

2.7　まとめ

本章では、IoT に関連するセキュリティ対策の標準規格、攻撃手口及び従来の検知手法とその問題点について述べた。セキュリティに関係する標準規格と政府機関のガイドラインでは、被害拡大を遮断するために、インシデントの発生を検知するためのインシデント検知だけでなく、インシデントの予兆であるイベントを検知することを求めている。さらに、IoT セキュリティガイドラインの要点では、機器等の状態や他の機器との通信状況を把握して記録することを求めている。サイバー攻撃の種類には、なりすまし、マリシャスコードやサービス妨害があり、不正アクセスの種類には、侵入口の探査には脆弱性試験やネットワーク識別子の各種スキャンがある。その中でも、DDoS 攻撃による社会基盤サービス妨害の被害が深刻である。この攻撃の仕組みとしてボットネットがあり、ボット化された IoT デバイスが攻撃の踏み台として加担している。サイバー攻撃や不正アクセスに対する従来の検知手法には、ウイルス検知、不正侵入検知（IDS）/不正侵入防止（IPS）及びシステム監視、トラフィック監視があり、そのほかに各ファイアウォールと検疫ネットワークがある。これらの従来の検知手法では、多種多様なマリシャスコードやサービス妨害の異なる手口をバターン化または機械学習する必要があるので、パターンファイルが肥大化し、検知モジュールの更新が不可欠で、緊急的や定期的な専門家によるアップデートとメンテナンスが検知の前提となっている。また、従来の汎用的なシステム監視が、IoT デバイスにおいてサイバー攻撃や不正アクセスの異常検知として有効であるか評価するために、予備実験を行った。この結果、従来のシステム監視では、疑似攻撃を与えた場合と平時の場合のイベントを判別することが困難なため、イベント検知として誤検知する可能性が高いことがわかった。本章では、従来の検知手法はインシデント検知が主流で、IoT デバイスを攻撃対象または踏み台とするサイバー攻撃や不正アクセスに対するイベント検知としては課題があることを明らかにした。

第 3 章

ネットワーク時刻同期技術を用いたホスト型イベント検知手法

3.1 はじめに

本章の目的は、時刻同期技術を用いたホスト型のイベント検知手法を提案することである。本論文で着目する時刻同期技術である chrony の時刻補正の仕組みを述べた上で、提案するホスト型のイベント検知手法の詳細を述べる。本手法は、NTP パケットから得られる時刻補正量、時刻同期要求・応答の通信遅延の揺らぎを計測することにより、イベントの検知を行う。本手法に基づく検知モジュールを開発して IoT デバイスに組込み、擬似的にサイバー攻撃を発生させた場合のイベント検知の実証実験を行う。本実験により、検知対象の IoT デバイスが、DDoS 攻撃の被害者または加害者、不正アクセスの被害者、さらには、非攻撃時のアイドル状態、許可されたユーザ処理としてファイルの送受信を行う場合のいずれにおいても、見逃しと誤検知の少ない精度のよいイベント検知が可能であることを確認し、本手法の有効性を示す。

3.2 ネットワーク時刻同期技術

3.2.1 NTP の階層構造とデータフォーマット

IoT デバイスを含めたコンピュータや通信処理装置などのノードは、各々のシステムクロックにより個別に時刻を管理している。1 基のノードのシステムクロックでは、正しい時刻に対して遅延や先行のずれが発生し、個々のノードの間ではシステムクロックの時刻のばらつきが発生する。時刻のずれは、各ノードはそのシステ

ム容量や性能を超えたシステム処理や大容量の通信データの送受信処理などの発生により、システムやデバイスの割り込み処理が多発して、システムクロックの時刻更新処理が正しく行われず発生する。時刻のばらつきは、各ノードのハードウェアやソフトウェアの特性や環境の違いにより発生し、ばらつきは時間経過とともに大きくなっていく。各ノードのシステムクロックの時刻のずれやばらつきは、異なるノード間でのシステム処理の同期や、情報セキュリティマネジメントシステム（ISMS）が求めるログ証拠の信頼性の確保にとって、大きな問題となる。この問題解決のため、一般的に、各ノードはネットワーク時刻同期技術を利用して、自身のローカル時計を標準時刻に同期させて整合を図っている。

ネットワーク時刻同期技術の機能と仕組みは、以下の通りである。ネットワーク時刻同期技術の機能は、時刻配信技術と時刻同期技術によって実現される。これら機能を提供する技術は、インターネット時刻供給技術またはネットワーク時刻同期技術と呼ばれている。ネットワーク時刻同期技術の仕組みは、インターネット上にStratum と呼ばれる階層構造から成り、これを図 3.1 に示す。この階層構造の最上位に、1 次参照源として原子時計に基づく標準時刻の NTP サーバが設置されている。1 次参照源に接続する下位層の NTP サーバが、電気通信事業者、インターネット接続事業者、企業や組織などに設置されている。同じ階層構造の NTP サーバ間はピアによりネットワーク時刻同期技術を冗長化している。

ネットワーク時刻同期技術を利用する IoT デバイスやコンピュータが NTP クライアントとして、各階層の NTP サーバに接続する（図 3.2 を参照)。これら NTP クライアントと NTP サーバとの間は NTP で接続される。NTP の通信手順（図 3.3、3.4 を参照）とデータフォーマット（表 3.1 を参照）は、IETF の RFC5905 で規定されている。例として、図 3.2 に示す NTP クライアントと NTP サーバとの時刻のずれは、以下の通信手順で認識する。

(1) NTP クライアントは、その時刻 T_1 に、NTP サーバに NTP リクエストを送信して、NTP サーバの時刻を問い合わせる。
(2) NTP サーバは、その時刻 T_2 に、NTP クライアントからの NTP リクエストを受信する。
(3) NTP サーバは、その時刻 T_3 に、NTP クライアントへ NTP レスポンスを返信して、NTP サーバの時刻を応答する。この T_3 は、NTP サーバが NTP リ

クエストを受信した時刻 T_2 から、NTP サーバの内部処理の時間が加算された時刻である。

(4) NTP クライアントは、その時刻 T_4 に、NTP サーバからのレスポンスを受信して、NTP サーバの時刻を得る。

(5) NTP クライアントは NTP リクエストの送信からレスポンスの受信までに経過した時間（T_4 - T_1）から、NTP サーバの処理時間（T_3 -T_2）を除いた、半分の時間を NTP の通信の片道時間とする。

(6) NTP クライアントは T_1 から T_2 までの時間と、上記 (5) の NTP の通信の片道時間を比較することで、NTP サーバと NTP クライアントの時刻のずれを認識する。

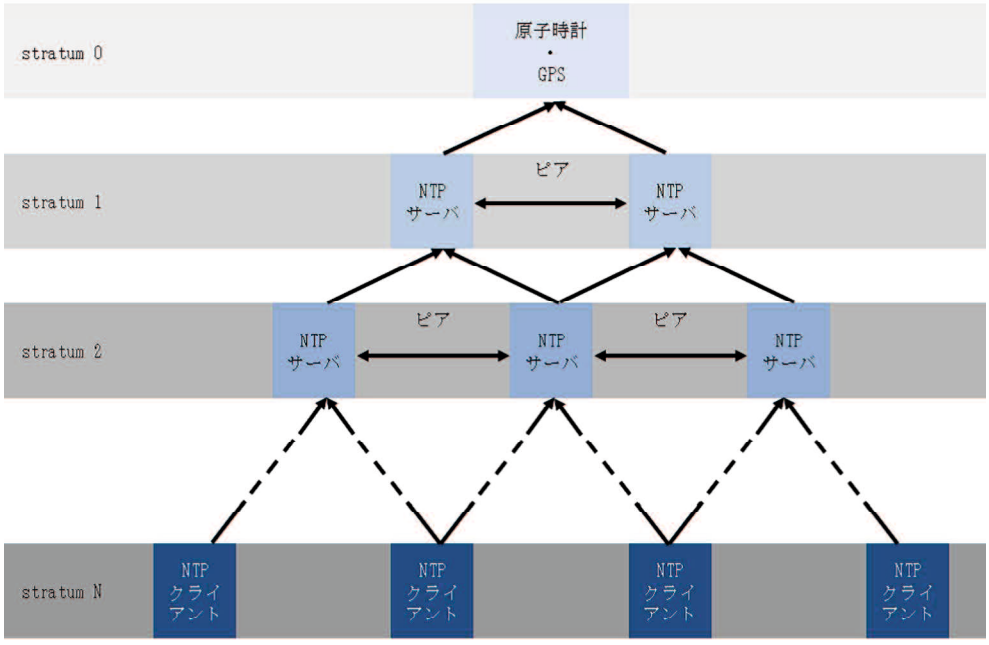

図 3.1　NTP の階層構造

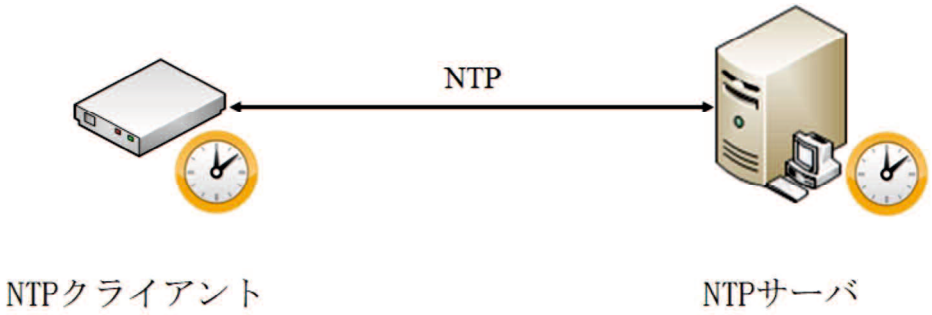

図 3.2　NTP クライアントと NTP サーバ

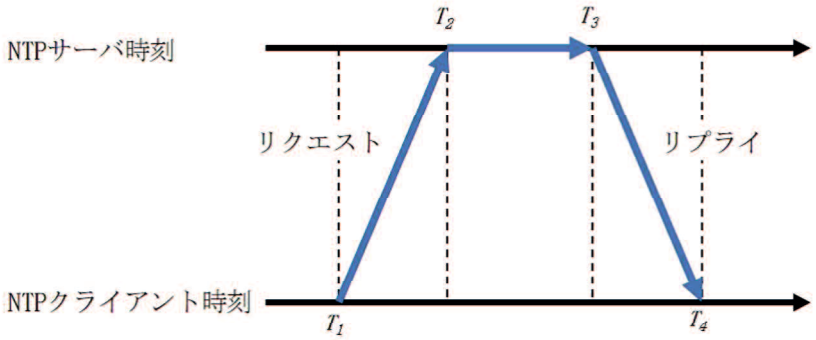

図 3.3　NTP データのリクエストリプライ

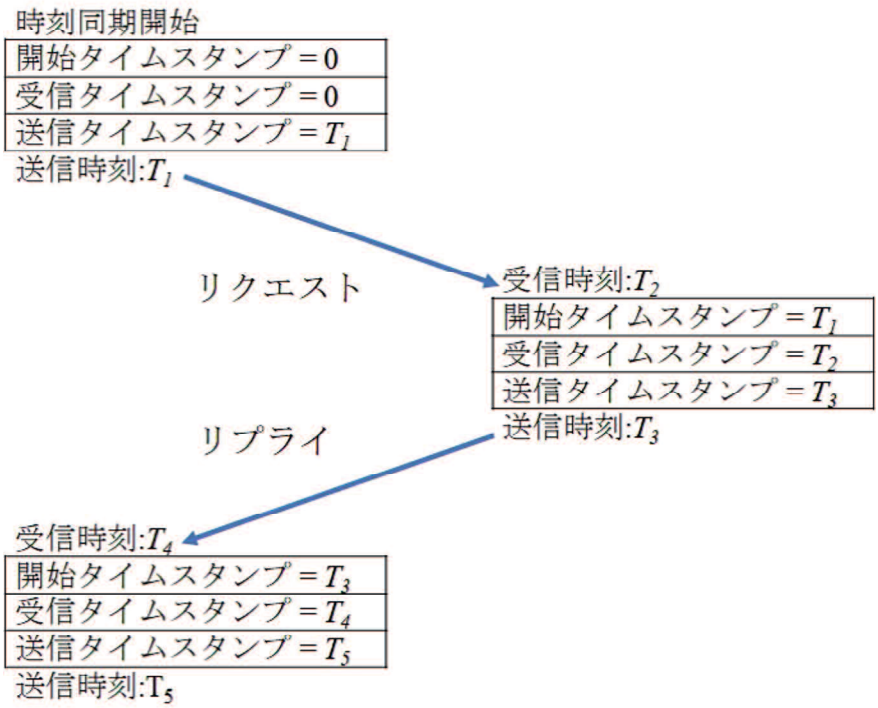

図 3.4　NTP データのシーケンス

表 3.1　NTP データフォーマット

閏秒指示子 (2bit)	バージョン番号 (3bit)	モード (3bit)	階層 (8bit)	ポーリング間隔 (8bit)	精度 (8bit)
ルート遅延（32bit、符号付固定小数点)					
ルート分散（32bit、符号付固定小数点)					
参照識別子（IPv4 アドレス)（32bit、文字列)					
参照タイムスタンプ（64bit、タイムスタンプフォーマット)					
開始タイムスタンプ（64bit、タイムスタンプフォーマット)					
受信タイムスタンプ（64bit、タイムスタンプフォーマット)					
送信タイムスタンプ（64bit、タイムスタンプフォーマット)					
鍵識別子（32bit)					
メッセージダイジェスト（128bit)					

- 閏秒指示子（Leap Indicator）

  うるう秒の事前予告を通知する。当該日の最後の 1 分が 1 秒追加または 1 秒削除されるかを指定する。本論文の実験ではすべて 0: 予告なし（通常状態）である。

- バージョン番号（Version Number）

  NTP バージョンを通知する。本実験では、すべて 3 とする。

- モード（Mode）

  動作モードを通知する。本実験では、検知対象の IoT デバイスは 3（Client)、NTP サーバは全て 4（Server）とする。

- 階層（Stratum）

  NTP サーバの階層を通知する。0 は原子時計や GPS 時刻であるが実際の NTP データとしては存在しない。1 は Stratum 0 に直結した NTP サーバ。2 以降は、直上位の Stratum の NTP サーバ、16 は時刻同期無し、17 以降は予約値である。

- ポーリング間隔（Poll Interval）

  連続する NTP データの送出間隔の最大時間間隔（秒）を通知する。本実験では、Stratum2 でデフォルトの 64 秒、Stratum3 の IoT デバイスで 2 秒とする。

- 精度（Precision）

NTP を扱う機器のシステムクロックが扱える時計の精度である。本実験では、Stratum0 はピコ秒単位、Stratum1 は 10 ナノ秒以内、Stratum2 は Stratum1 までのインターネット及び WAN 接続回線の距離とアクセス負荷の重さに影響を受け数百マイクロ秒程度、Stratum3 は Stratum2 の精度に構内ネットワークのアクセス負荷の重さに影響を受け数ミリ秒から数十ミリ秒程度とする。

- ルート遅延（Root Delay）

  1 次参照源（Stratum 1 の NTP サーバ）までの往復遅延（ラウンドトリップ）の遅延時間である。この値が大きいほど、ネットワークの距離が遠く、精度が低くなることを意味する。NTP サーバは直上位の NTP サーバからこの値を受け取ると、自サーバの RTT を加え、下位 NTP クライアントへ通知する。

- ルート分散（Root Dispersion）

  1 次参照源までの往復遅延の相対的な誤差時間である揺らぎ（Jitter）である。この値が大きいほど、ネットワーク遅延の予測が荒く、精度が低くなることを意味する。NTP サーバは直上位 NTP サーバからこの値を受け取ると、自サーバの揺らぎを加え、下位 NTP クライアントへ通知する。

- Reference Clock（相手先）

  参照している NTP サーバである。Stratum2～14 の NTP サーバの場合は、直上位の NTP サーバの IP アドレス。Stratum1 の NTP サーバは、参照先を暗示する任意の 1～4 文字の ASCII 文字列である。

- 鍵識別子（Key Identifier）

  本実験では、使用しない。

- メッセージダイジェスト（Message Digest）

  本実験では、使用しない。

3.2.2 NTP による時刻補正の仕組み

NTP クライアントは、NTP サーバに対して、以下の NTP リクエストと NTP レスポンスに含まれる 2 つのカテゴリの情報により、システムクロックを補正する。カテゴリ 1 は、NTP クライアントと 1 次参照源との間の情報であり、1 次参

照源の標準時計からのずれ（ルート分散: RD_t）と、通信の往復遅延時間（ルート遅延: RD_c）である 。カテゴリ 2 は、NTP クライアントと直上位の NTP サーバとの間の情報であり、NTP リクエストの発信時刻（T_1）と着信時刻（T_2）、および、NTP レスポンスの発信時刻（T_3）と着信時刻（T_4）である。NTP リクエストの遅延時間（δ_u）と NTP レスポンスの遅延時間（δ_d）は、それぞれ、式 (3.1) と式 (3.2) で表される。

$$\delta_u = T_2 - T_1, \tag{3.1}$$

$$\delta_d = T_4 - T_3. \tag{3.2}$$

直上位の NTP サーバとの相対的なローカル時計の差 θ は、式 (3.1) で表される。

$$\theta = \frac{(T_2 - T_1) + (T_4 - T_3)}{2}. \tag{3.3}$$

ネットワーク時刻同期技術では、Slew モード（段階的補正）と Step モード（即時的補正）を組み合わせて、RD_t と θ から補正値（r_o）を算出し、ノードの時刻を補正する。これにより理論的には、同じ 1 次参照源にアクセスするノードは、全て同じ標準時刻に補正され、同期することになる。

3.2.3 時刻同期モジュールによるローカルクロックの補正

IoT デバイスは、一般的なコンピュータと同様にマザーボード上のハードウェアクロックとして、バッテリ駆動の RTC（Real Time Clock）を搭載する。だが、家電などでは RTC を搭載しない IoT デバイスも多く存在する。IoT デバイスは、システム起動時にハードウェアクロックを時刻の初期値として参照して時刻を管理する。システムクロックの標準時刻は、UTC の Universal Time と JST などの Local Time が使用される。Linux 系 OS で Time Protocol を利用した時刻同期モジュールによるローカルクロックの補正としては、従来は ntpd が標準的に実装され利用されてきたが、最近は chronyd が標準的に実装され始めている。chronyd は、Linux 系 OS 上のシステムクロックのレートを幅広い範囲で調整可能である。また、RTC を搭載しない IoT デバイスでもインターネット経由で現在の時刻を参照しシステムクロックの管理を可能にする。特に、IoT デバイスで利用される可能性の高い仮想マシン下のゲスト OS 上のシステムクロックが異常動作を発生させたり動作が不安定になったりしても時刻補正が可能である。chronyd は ntpd と比較

して以下のメリットある。システムクロックの同期が高速で正確性もより高いこと。外部の時間参照アクセスが断続的する場合や長時間ネットワークの通信処理が遅延する場合でも機能すること。水晶振動子の温度変化などによってシステムクロックのレートが突然変動しても短時間に安定すること。デフォルト設定として他の実行中のプログラム処理に影響を与えないために、システム起動時にシステムクロックが同期された直後は、短いインターバルで時刻を更新しないこと、などがある。一方、ntpd には、以下のデメリットがある。時刻同期には規則的な時間参照のポーリングが必要であること。温度変化による変動が収束するまでに長時間かかる場合があること。細やかな時刻同期には複雑な設定手続きが必要であること、などがある。chronyd は、NTP のカテゴリ 1 とカテゴリ 2 の揺らぎとシステムクロックの時刻補正の値を、以下のログに記録する。measurements.log には NTP の測定結果と関連情報の値を、statistics.log には回帰処理結果の統計情報を、tracking.log にはシステムクロックの先行または遅延の予測レート、変更値及びその追跡結果を、rtc.log にはシステムのリアルタイムクロックについての情報を、refclocks.log には生およびフィルター処理された参照クロックの測定結果を、tempcomp.log には温度測定結果とシステムレートの補正値を、記録する。

3.3 ホスト型イベント検知手法

3.3.1 原理

攻撃対象の IoT デバイスでは、サイバー攻撃により多数のノードから大量の通信データを受信すると、受信ノード内では、ネットワークカード（Network Interface Card）から通信ドライバ、アプリケーションへとデータが転送される際に、ハードウェア及びソフトウェア割り込み処理が連鎖的に大量に発生する。また、攻撃対象の IoT デバイスは、受信に対するリプライを返信するために、逆方向にも割り込み処理が発生する。この結果、攻撃対象の IoT デバイスでは、システム処理やネットワーク処理が過負荷になり、時刻同期処理が遅延するため、時刻のずれや通信遅延の揺らぎが発生する。さらに、攻撃対象の IoT デバイスにおいて、それら処理の過負荷が、重篤な場合は、バッファオーバフローにより NTP の通信データが破棄され、時刻同期処理がスキップされるため、時刻のずれは大きくなる。時刻同期技術を利用したホスト型イベント検知手法の原理は、攻撃対象の IoT デバイスにおい

て、NTP データから得られる通信遅延の揺らぎと時刻のずれを用いてイベント検知を行うことである。これらの指標は、1 次参照源の時刻を基準とし、攻撃を受けた際の割り込み処理量により変動するため、過負荷な状態が攻撃によるものか否かを判断しやすい。

3.3.2 検知手法

IoT デバイス向けに、時刻同期技術を利用したホスト型イベント検知手法を提案する。本提案手法では、検知対象 IoT デバイスにおいて、時刻の補正値 r_o と、式 (3.1)、(3.2) に示す遅延時間 δ_u、δ_d を用いてイベントを検知し、それがインシデントに至る可能性の高い（インシデントの予兆を示す）イベントであるかどうかを判定する。直上位の NTP サーバにリクエストを送信し、各データを取得する間隔を F_{req} とする。提案検知手法では、F_{req} 毎に、以下に示す手順を繰り返す。この手順を 3.5 に示す。

1. データの記録

   時刻 t において採取したデータを $r_o(t)$、$\delta_u(t)$、$\delta_d(t)$ として記録する。ノード内に、直近の N 個のデータを保持するものとする。

2. 基本統計情報の算定

   記録した各データについて、平均 μ_{r_o}、μ_{δ_u}、μ_{δ_d} および標準偏差 σ_{r_o}、σ_{δ_u}、σ_{δ_d} を算定する。

3. ノイズの平準化

   $r_o(t)$、$\delta_u(t)$、$\delta_d(t)$ のうち、σ_{r_o}、σ_{δ_u}、σ_{δ_d} の 2σ（約 96%）の信頼区間を外れた値はノイズとし、それぞれ μ_{r_o}、μ_{δ_u}、μ_{δ_d} に置換する。

4. 移動平均法による統計情報の算定

   1 スパンのデータ数を n とし、スパン内の移動平均 $SMA(r_o(t))$、$SMA(\delta_u(t))$、$SMA(\delta_d(t))$ を算定する。ただし、移動平均 $SMA(P(t))$ は次式で定義される。

$$SMA(P(t)) = \sum_{i=0}^{n-1} \frac{P(t-i)}{n}. \tag{3.4}$$

   直近の N 個の移動平均 $SMA(P(t'))$ から、標準偏差 $\sigma_{SMA(r_o(t))}$、

$\sigma_{SMA(\delta_u(t))}$、$\sigma_{SMA(\delta_d(t))}$、および、変動係数 $CV(r_o(t))$、$CV(\delta_u(t))$、$CV(\delta_d(t))$ を算定する。ただし、$t' = t, t-1, \cdots, t-(N-1)$ であり、変動係数 $CV(P(t))$ は次式で定義される。

$$CV(P(t)) = \frac{\sigma_{SMA(P(t))}}{SMA(P(t))}. \tag{3.5}$$

5. イベントの判定

   $r_o(t)$、$\delta_u(t)$、$\delta_d(t)$ の 1 スパン内で、それぞれ、$\sigma_{SMA(r_o(t))}$、$\sigma_{SMA(\delta_u(t))}$、$\sigma_{SMA(\delta_d(t))}$ の σ（約 68%）の信頼区間を外れた値が連続した場合、そのスパンをイベントと判定する。それぞれにおいて、イベントか否かを表すフラグを $F_{r_o(t)}$、$F_{\delta_u(t)}$、$F_{\delta_d(t)}$ とし、イベントであれば 1、そうでなければ 0 を記録する。

6. イベントのインシデント判定

   各イベントのフラグと変動係数から、以下の条件 (3-A) または条件 (3-B) に当てはまる場合に、検知したイベントをインシデントの予兆であると判定する。

   条件（3-A）　式 (3.6) が成立

$$\begin{aligned}&(F_{r_o(t)} = 1 \wedge 0.9 \leq CV(r_o(t))) \wedge \\ &(F_{\delta_u(t)} = 1 \wedge 0.5 \leq CV(\delta_u(t))) \wedge \\ &(F_{\delta_d(t)} = 1 \wedge 0.5 \leq CV(\delta_d(t)))\end{aligned} \tag{3.6}$$

   条件（3-B）　式 (3.7) が成立

$$\begin{aligned}&(F_{r_o(t)} = 1 \wedge 0.5 \leq CV(r_o(t))) \wedge \\ &\{(F_{\delta_u(t)} = 1 \wedge 0.9 \leq CV(\delta_u(t))) \vee \\ &(F_{\delta_d(t)} = 1 \wedge 0.9 \leq CV(\delta_d(t)))\}\end{aligned} \tag{3.7}$$

上記のインシデント判定の式 (3.6) と (3.7) は、2.3.5 に示したサイバー攻撃の特性を考慮したものである。条件（3.6）は、DDoS 攻撃の被害側の場合及び踏み台攻撃の加害者側の場合の条件で、攻撃により、直接的または間接的に CPU やメモリのシステムリソースが消費され、システムが過負荷になり、ネットワーク負荷も増加する可能性が高いと想定する。一方、条件（3.7）は、不正アクセスの標的側の場合の条件で、以下の可能性が高いと想定する。脆弱性試験では、その負荷が検知されないように探査が行われる。

ポートスキャンでは、探査とその応答で上り下りともにネットワークは過負荷になるが、システムは過負荷にはならない。Web への疑似攻撃では、その応答処理でシステム負荷が増加し、応答のために下りのネットワークのみ過負荷になる。

本提案手法の利点は、以下のとおりである。本提案手法では、汎用的な時刻同期技術で取得できる情報を活用するため、検知のために特別な装置や技術を必要とせず、ノードやネットワークに検知のための負荷をほとんど発生させない。また、検知に用いる NTP は、フィルタリング装置や他の検知装置を通過させるのが一般的であるため、ネットワークトポロジや他のノードの配置などの要因による妨害は受けにくい。本提案手法を用いた 3-4 で述べる検知用モジュールのバイナリファイルの合計容量は約 160KiB、検知用モジュールの物理メモリサイズ消費量（Resident Set Size）は約 7MiB、仮想メモリ領域（Virtual Set Size）は約 8MiB と小さく、本提案手法はメモリ容量に乏しい IoT デバイス向きである。一方、本手法は、$r_o(t)$、$\delta_u(t)$、$\delta_d(t)$ の値があまり変化しないような攻撃には不向きである。そのため、本提案手法は、より巧妙な攻撃に対応するために、他の手法との併用することが考えられる。

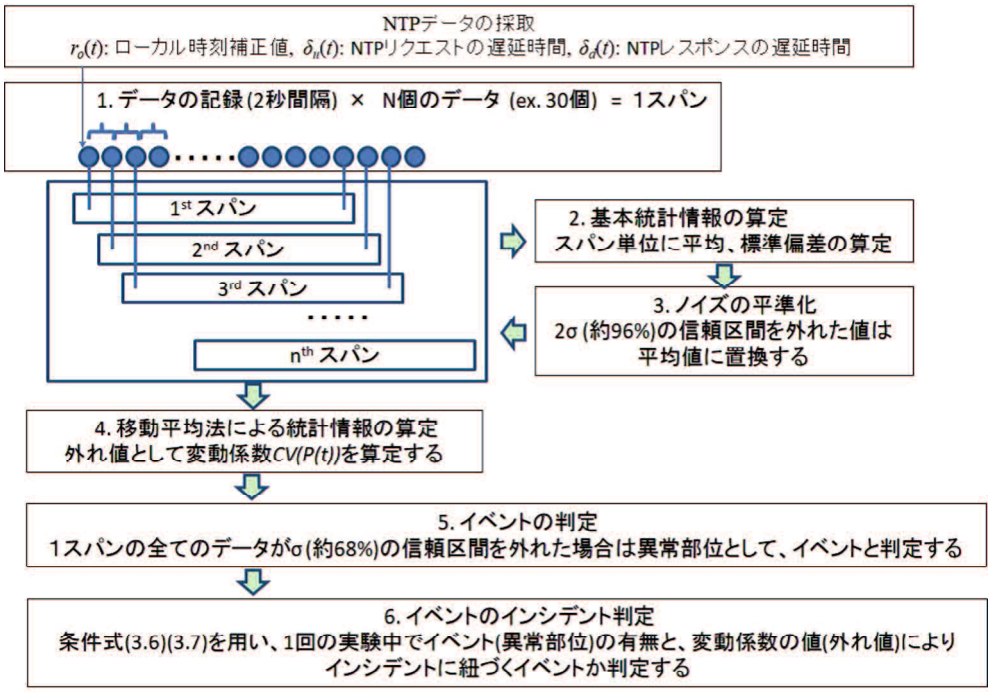

図 3.5　ホスト型イベント検知手法の手順

3.3.3 実験目的と方法

本実験では、ホスト型イベント検知手法により、疑似的にサイバー攻撃を行った場合に、インシデントの予兆としてのイベントを正しく検知できるか、また、攻撃のない平時において誤検知があるかを確認し、検知手法としての有効性を検証することを目的とする。本実験のサイバー攻撃は、表 2.4 に示したデータリンク層への攻撃として PoD（Ping of Death)[108] による DDoS 攻撃と、アプリケーション層への探査としての不正な脆弱性試験、アプリケーション層への攻撃として標的型大容量ファイル添付メール転送とした。実験の狙いは、上記の順に、大量パケット受信によるネットワークの過負荷、各種アプリケーションへの不正アクセスによる CPU 負荷の増大、大容量ファイルの受信による通信処理の増大とディスク書き込みによる CPU の割り込み処理増大である。

本提案手法が、検知対象とする IoT デバイスのソフトウェア環境に求める要件を、以下に述べる。OS は、Linux 系 OS または Windows 系 OS であること。ただし、制御系 OS へのポーティングも可能。ネットワーク時刻同期モジュール（chronyc）が既に稼働しているか、追加導入が可能なこと。パケットスニファの API が提供されること。例として、Linux 系 OS の場合は libpcap 、Windows 系 OS の場合は WinPcap が挙げられる。実験環境を図 3.6 に示す。図 3.6 の時刻同期の階層構造は、Stratum1 の日本標準時刻サーバを時刻の一次時刻源として、構内ネットワーク内に Stratum2 のローカル時刻同期サーバを置き Stratum1 に接続する、検知対象の IoT デバイスを Stratum3 の時刻同期クライアントとして Stratum2 に接続する。図 3.6 の環境では、外部の攻撃者 N_W が標的 IoT デバイス T_{L1-IoT} を直接攻撃する、または、内部のマルウェア M_L が T_{L2-IoT} を踏み台として間接的に攻撃する 2 通りの攻撃ルートを想定する。実験環境のノードを表 3.2 に、ネットワーク接続を表 3.3 に示す。T_{L1-IoT} 及び T_{L2-IoT} には、図 3.7 に示すボードコンピュータ Raspberry Pi を用い、開発した検知モジュールを稼働させる。

この検知モジュールは、主に C 言語を用いて開発した。本実験の Web カメラ監視では、カメラ映像をストリーミング配信する WWW サービス（Apache httpd)、映像ファイルの保存のためにファイル共有サービス（samba）及び OS やプログラム更新のために FTP サービス（vsftpd）を稼働させる。ストリーミング配信は、1 フレーム 320x249 ピクセル、サンプリングフォーマット YUV、映像圧縮 JPEG、

Quality80 で、1 秒間に平均 10 フレーム転送され、この受信は T_S1、T_S2 の 2 ノードで行う。システム時刻補正モジュールには、chronyd を採用する。この設定は、ローカルの ND へのリクエスト頻度 Freq を 2 秒、NW へのリクエスト頻度を 1024 秒とし、その他はデフォルト値とする。また、各実験でノード内に保持する取得情報数 N は 600、1 スパンのデータ数 n は 30 とする。理由は、サイバー攻撃の事例で、従来の検知手法で発見しづらい 1 回あたりの攻撃時間が最大でも 3 分間程度であるためである。N_W から T_{L1-IoT} へ不正アクセスは、脆弱性検知スキャナである Nessus[158] を使用する。N_W から T_{L1-IoT} への DDoS 攻撃は、ネットワークテストコマンドである ab（ApacheBench、Version2.3）[159] を使用する。M_L は踏み台として N_W からの遠隔操作により T_{L1-IoT} へ、ab で DDoS 攻撃を行う。実験は、以下のとおり、サイバー攻撃がある場合（実験 3-1、3-2）と平時の場合（実験 3-3）について行った。

実験 3-1: サイバー攻撃がある場合

(3-1-1) 検知対象が PoD による DDoS 攻撃の被害者側の場合

N_W から T_{L1-IoT} に疑似的に DDoS 攻撃を実行。実験開始 10 分後に、以下のコマンドを 1 回実行

ab -n 100000 -c 100 TL1-IoT の IP アドレス

(3-1-2) 検知対象が不正アクセスの被害者側の場合

N_W から T_{L1-IoT} に脆弱性試験ツールを利用して疑似的に不正アクセスを実行。実験開始 10 分後に、以下のツールを 1 回実行

Nessus（Web Application Scanning)、T_{L1-IoT} の IP アドレス

(3-1-3) 検知対象が PoD による DDoS 攻撃の踏み台（加害者）側の場合

M_L から TL に疑似的に DDoS 攻撃を実行。実験開始 10 分後に、以下のコマンドを 1 回実行

ab -n 100000 -c 100 TL の IP アドレス

実験 3-2: 検知対象が大容量ファイル転送の受信処理を行う場合（セキュリティ違反として）

(3-2-1) 検知対象にて中程度のリスクが予測される場合

TL から T_{L1-IoT} へ ftp で 100MiB のファイル転送を、実験開始 10 分

後に、1 回実行

(3-2-2) 検知対象にて高いリスクが予測される場合

TL から T_{L1-IoT} へ ftp で 1GiB のファイル転送を、実験開始 10 分後に、1 回実行。

実験 3-3: サイバー攻撃がなく平時処理を行う場合（セキュリティ許可として）

(3-3-1) 平時処理のみの場合

(3-3-2) 検知対象がファイル受信処理を行う場合

TL から T_{L1-IoT} へ ftp で 10MiB のファイル転送を、実験開始 10 分後に、1 回実行。

(3-3-3) 検知対象がファイル送信処理を行う場合

T_{L1-IoT} から TL へ ftp で 10MiB のファイル転送を、実験開始 10 分後に、1 回実行。

1 回の実験時間は 20 分間とした。試行回数は、攻撃・不正アクセスが有の場合の実験として、実験 3-1 は各 20 回で合計 60 回、実験 3-2 は各 20 回で合計 40 回の合計 100 回とする。平時の場合の実験として、実験 3-3 は (3-3-1) で 40 回、(3-3-2) および (3-3-3) で 30 回の合計 100 回とする。本実験では、1 回の実験の終了時点で計測されたデータに対して、3.3.2 で示した検知手法の手順 5. と 6. によってイベントのインシデント判定を行うものとする。ただし、実験の終了時点で N 個の移動平均が揃っていない場合は、その分の値は無視して計算するものとする。

3.3.4 実験結果

実験結果を表 3.4〜表 3.6 に示す。表 3.4 は、実験 3-1 と実験 3-2 において、インシデントとしてイベントを検知した回数を TP に、検知できなかった（見逃し）回数を FN に集計し、実験 3-3 において、誤ってイベントを検知した（誤検知）回数を FP に、検知しなかった回数を TN に集計したもので、表 3.5 は、各実験で成立した条件式の内訳を示している。表 3.4 の結果より、攻撃・不正アクセス有と平時で、それぞれ 100 回の実験のうち、見逃しは 1 回、誤検知は 2 回であり、ほとんどの場合において、イベントを正しく検知してており、イベントの誤検知も少ない。

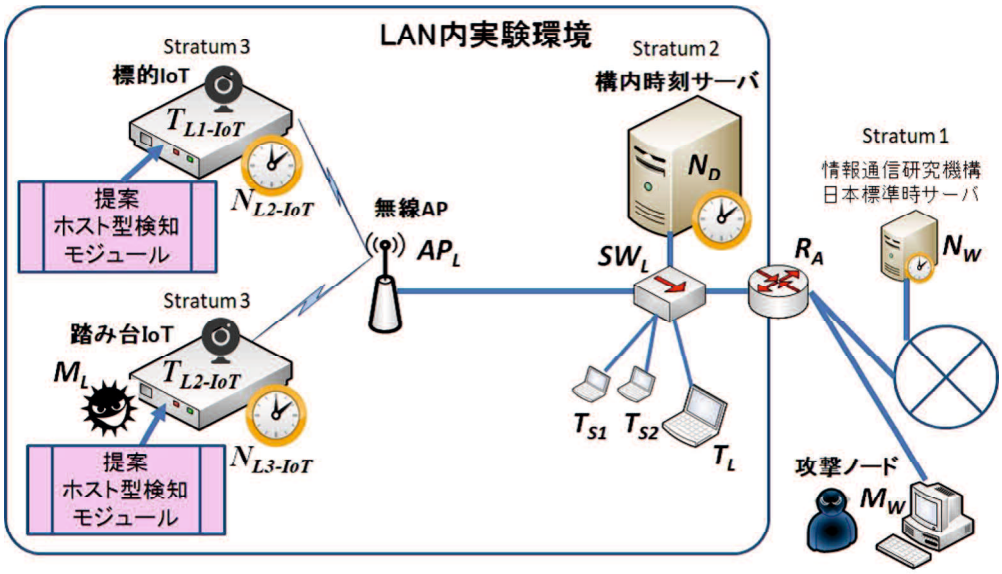

図 3.6　ホスト型検知手法に向けた実験環境

表 3.2　ホスト型イベント検知手法に向けたノードの仕様

ノード	CPU（Clock)	メモリ	OS（Kernel)
T_{L1-IoT}、T_{L2-IoT}	ARM1176JZF-S ARMv6l（700MHz)	512MB	Raspbian 7.8 (4.1.13+)
T_L、T_{S1}、T_{S2}	AMD Athlon X2 5000B（2.6GHz)	8GB	Fedora 21 (4.1.13)
M_W	AMD Athlon II X2 220（2.8GHz)	10GB	Fedora 21 (4.1.13)

表 3.3　ホスト型イベント検知手法に向けたネットワーク接続

区間	リンク	規格	最大速度
T_{L1-IoT}、$T_{L2-IoT} \rightleftharpoons AP_L$	Wiress LAN	IEEE802.11n	300Mbps
AP_L、N_D、T_L $M_W \rightleftharpoons R_A$	Wired LAN	IEEE802.3	1Gbps
$R_A \rightleftharpoons$ WAN	FTTH	IEEE802.3u	100Mbps

3.3.5　考察

実験結果について、表 3.5 と表 3.10 から考察する。攻撃・不正アクセス有の場合、次の実験の全回で、DDoS の踏み台攻撃は、式 (3-4)、大容量ファイル転送は。

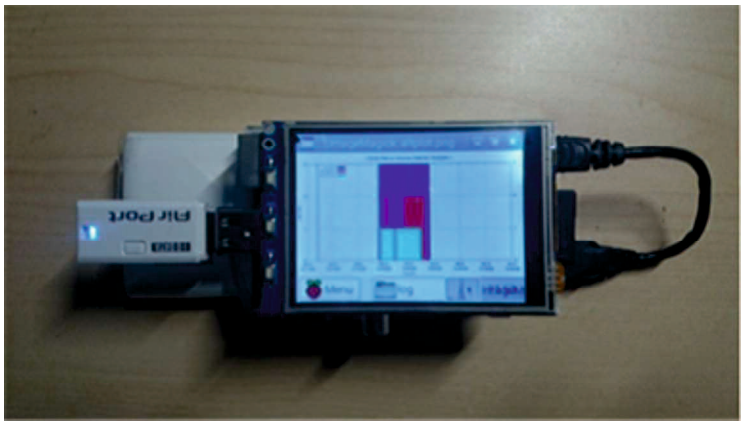

図 3.7　検知手法を実装したボードコンピュータ

式 (3-3) の条件を満たしていることから、前者はネットワーク負荷が、後者はシステム負荷の影響が極めて高い。DDoS 攻撃とシステム負荷とネットワーク負荷の影響が両方とも極めて高い。脆弱性試験では、システム負荷よりネットワーク負荷の影響が極めて高い。見逃しは DDoS の踏み台攻撃の 2 回で $CV(\delta_d(t))$ の値が 0.5 以下であった。平時では、100 回のうち 16 回 $CV(r_o(t))$ が 0.9 を超し、10Mib のファイルの送信で 2 回の誤検知となった。表 3.11 の結果より、見逃しは DDoS の踏み台攻撃の 2 回のみであった。理由は、表 3.10 で示された $CV(\delta_d(t))$ の値と表 3.11 で示された $F(\delta_u(t))$ と $F(\delta_d(t))$ の判定結果による。その他のすべての攻撃・不正アクセスの $F_{r_o(t)}$、$\mathrm{F}(\delta_u(t))$ と $\mathrm{F}(\delta_d(t))$ は正しく判定している。平時では、システム負荷として 100 回中 60 回で $F_{r_o(t)}$ の値が 1 であった。理由としては、システム負荷には、OS や起動中のサービス、その他のデーモンの処理がバックグラウンド処理として行われるが、それらが偶然重なり、各フラグや変動係数に影響を及ぼしたためだと考えられる。ただし、最終のインシデント判定としては、ネットワーク負荷の判定条件により誤検知は抑制されている。表 3.4 の実験結果を定量的に評価するため、パターン認識で取り扱われている陽性尤度比（LR+）、陰性尤度比（LR-）、オッズ比（OR）及び F 値を算出して、表 3.6 に各指標の定義式と値を表す。一般的に、LR+ が 10.0 以上であれば正しく判定できており、LR-が 0.1 以下であれば誤った判定を排除できていると判断される [160, 161]。表 3.6 より、LR+、LR-とも基準値以上または以下の値を示している。OR の値が 4950 であることは、正しく検知する割合が誤検知をする割合の 4950 倍であることを示している [162]。また、適合率、再現率及び F 値から、正確性と網羅性のバランスもよい。

イベント検知結果を可視化したディスプレイ表示例として、実験 (3.8)、(3.9)、(3.10) 及び (3.11) の結果を、それぞれ、図 3.8、図 3.9、図 3.10、図 3.11 に示す。縦軸は、1 がフラグ $F_{r_o(t)}$、2 が $F(\delta_u(t))$、3 が $F(\delta_d(t))$ に対応している。各値の大きさには意味はなく、同じスパンでのフラグの重なりを見やすくするためのものである。横軸は、実験時間を表している。図 3.8 では、09:08:52〜09:28:52 の 20 分間で監視を行い、09:12:50〜09:25:20 の間に DDoS 攻撃を行った結果を示している。09:13:38〜09:25:37 の間で $F_{r_o(t)}$、$F(\delta_u(t))$ 及び $F(\delta_d(t))$ のいずれかでイベントを検知し、攻撃を行った時間帯とほぼ一致している。そのうちの 09:15:14〜09:25:03 の間で、3 つともイベントが重なっており、式 (3.6) が成立し、インシデントの予兆を示すイベントであると判定している。図 3.9 では、14:55:03〜15:15:03 の 20 分間で監視を行い、15:00:19〜15:10:06 の間に Nessus による脆弱性試験を行った結果を示している。14:54:53〜14:56:11 の間にイベントを検知しているが、脆弱性試験の開始以前であり他の処理によるものである。15:01:16〜15:02:13 に $F(\delta_d(t))$ で Nessus からのポートスキャンと疑似攻撃のイベントを検知している。15:02:55〜15:06:49 の間に $F(\delta_u(t))$ または $F_{r_o(t)}$ でイベントを検知し、そのうちの 15:03:36〜15:06:09 の間で 2 つのイベントが重なっており、式 (3.7) が成立している。図 3.10 では、09:02:00〜09:22:00 の 20 分間行い、09:09:03〜09:14:51 の間に踏み台として DDoS 攻撃を行った。この結果、$F_{r_o(t)}$ は 09:10:32〜09:14:53、$F(\delta_u(t))$ 09:10:21〜09:15:16、$F(\delta_d(t))$ は 09:10:45〜09:15:24 の間でイベントを検知している。これら 3 つのイベント検知の結果を、それぞれ図 3.12、図 3.13、図 3.14 で、それぞれ取得したデータ、スパン内の移動平均、イベントか否かを表すフラグを示す。これら 3 図のイベント検知の結果を図 3.10 で重ねると、攻撃を行った時間帯とほぼ一致している。また、$CV(r_o(t))$ は 2.21、$CV(\delta_u(t))$ は 1.78、$CV(\delta_d(t))$ は 1.79、で式 (3.6) 及び式 (3.7) ともに成立している。図 3.11 では、20:30:03〜20:50:13 の 20 分間で監視を行い、20:40:00〜20:40:02 の間にファイル受信を行った結果を示している。ファイル受信後の 20:41:18〜20:42:04 の間、$F_{r_o(t)}$ でイベントを検知しているが、同時間帯において他の 2 つではイベントを検知しておらず、ファイル受信をインシデントの予兆を示すと誤検知する判定はなかった。これらの表示はあくまで一例であり、検知対象の IoT デバイスによっては、LED の点滅やメールの送信などにより通知する方法も考えられる。その他、攻撃時と平時のいず

れの実験においても $r_o(t)$、$\delta_d(u)$、$\delta_d(t)$ にイベントが発生するが、その発生時刻には時系列的な差異があることが読み取れる。本検知手法は、これらの値の特徴をインシデント判定の条件に考慮することで、誤検知の少ない判定を可能にしている。

観測データの正規性と有効性についての考察

本小節では、実験で観測した標本データ $CV(P(t))$ の正規性と有効性について下記の 3 つの評価方法により統計的に確認する。なお、攻撃・不正アクセスが有の場合、平時の平均に対して、システムクロックの時刻やネットワークの通信速度の変動は基本的に遅延するのみのため、$CV(P(t))$ の確立分布は右片側のみとなる。

1 つめの評価方法として、まず、標本データ $CV(P(t))$ の正規性について、ヒストグラムから確認する。$CV(r_o(t))$、$CV(\delta_u(t))$、$CV(\delta_d(t))$ を、実験 3-1、実験 3-2、実験 3-3 毎に、0.1 単位でランク別に集計して、ヒストグラムにデータの分布状態とデータの範囲を図 3.15、図 3.16、図 3.17 で示す。さらに、例として実験 3-1 の標本データ $CV(r_o(t))$ を図 3.18、図 3.19、図 3.20 で示し、ヒストグラムと正規分布曲線とを重ねて正規性を確認した。

2 つめの評価方法として、標本データ $CV(P(t))$ の基本統計量の尖度と歪度から正規性を確認する。基本統計量は、表 3.7 に示す。$CV(\delta_u(t))$ の尖度は、2.6280 で正規性が高く、$CV(r_o(t))$ は 0.1906 で、$CV(\delta_d(t))$ は 7.5400 で、これらの正規分布は左にずれており、尖りが急である。$CV(r_o(t))$ の歪度は 1.2924、$CV(\delta_u(t))$ の歪度は 1.9406、$CV(\delta_d(t))$ の歪度は 2.6836 で、いずれの正規分布の形状も山がとがっているので、外れ値が存在しえる。よって、標本データには正規性があると言える。

3 つめの評価方法として、標本データ $CV(P(t))$ の有効性を、(3-1-1) 検知対象が PoD による DDoS 攻撃の被害者側の場合と (3-3-1) 平時の身の場合の 2 つの標本データ $CV(r_o(t))$ の有意差を検定することで確認する。平均の有意差を t 検定で、分散の有意差を F 検定で行う。有意差は、仮説として 2 つの標本データの平均と分散には差異が無いとして、帰無仮説を棄却により確認する。まず、t 検定の結果を表 3.8 に示し、平均の有意差を下記 2 つの仮説の棄却によって確認する。1 つめの仮説は、$P(T <= t)$ が設定した棄却域の確率より小さければ、帰無仮説を棄却する。結果として、$P(T <= t)$ は 1.38E-10 で、棄却域の確率の 0.05 より小さい

ので、帰無仮説を棄却できる。2 つめの仮説は、t 境界値が t の絶対値より小さければ、帰無仮説を棄却する。結果として、t 境界値は 2.00 で、t の絶対値の 7.61 より小さいので、帰無仮説を棄却できる。よって 2 つの標本データの平均には有意差がある。一方、F 検定の結果を表 3.9 に示し、分散の有意差を下記 2 つの仮説の棄却によって確認する。1 つめの仮説は、F 境界値が分散比（F 値）より小さければ、帰無仮説を棄却する。結果として、F 境界値は 1.8599 で、分散比（F 値）の 25.47 より小さいので、帰無仮説を棄却できる。2 つめの仮説は、$P(T <= t)$ が設定した棄却域の確率より小さければ、帰無仮説を棄却する。結果として、$P(T <= t)$ は 1.38E-10 で、棄却域の確率の 0.05 より小さいので、帰無仮説を棄却できる。よって 2 つの標本データの分散に有意差がある。以上のことから、標本データには有効性があると言える。

イベントのインシデント判定の条件式の考察

本小節では、イベントのインシデント判定の条件式における変動係数の閾値について考察する。本提案手法において、インシデントに紐づくイベントと判定する条件は、記録したデータに、外れ値が存在することと、異常部位が存在することである。外れ値の存在は変動係数の値に、変動または大きな変動があることであり、異常部位の存在はフラグで判定する。一般的に、変動係数の値が 0.5 以上であれば変動があり、1.0 以上であれば変動が大きいと判断される [163, 164, 165]。本手法では変動係数は、3.3.2 節における、検知手順の 4. で算定する。結果は、変動係数の範囲に分類して、実験件数を集計して表 3.10 の $CV(r_o(t))$、$CV(\delta_u(t))$、$CV(\delta_d(t))$ に示す。実証実験における変動係数の結果は、$CV(r_o(t))$ は、攻撃時は 0.9 以上、不正アクセス時は 0.5 以上であった。$CV(\delta_u(t))$ と $CV(\delta_d(t))$ は、両方とも攻撃時 05 以上で、不正アクセス時はどちらかが 0.9 以上であった。本手法では異常部位の存在としてフラグは、3.3.2 節における、検知手順の 5. で判定し、1 スパンの 30 個のデータが全て σ の信頼区間を外れた場合、条件式のフラグ $F_{r_o(t)}$、$F_{\delta_u(t)}$、$F_{\delta_d(t)}$ の値は 1 となる、実証実験における、各実験のフラグの値を表 3.11 で示す。結果として、攻撃時および不正アクセス時の実験のほぼ全てで、フラグの値は 1 であり、異常部位の存在を示している。

ここで、本提案手法における当初の変動係数の閾値が、適切であるかを以下に検

討する。まず、変動係数の閾値を、変数 a、b、c、d として、以下の条件（3-A'）と条件（3-B'）を設定する。

条件（3-A'）　式 (3.8) が成立

$$\begin{aligned}&((F_{r_o(t)}=1)\wedge(a\leq CV(r_o(t))))\wedge\\&((F_{\delta_u(t)}=1)\wedge(b\leq CV(\delta_u(t))))\wedge\\&((F_{\delta_d(t)}=1)\wedge(b\leq CV(\delta_d(t)))).\end{aligned}\tag{3.8}$$

条件（3-B'）　式 (3.9) が成立

$$\begin{aligned}&((F_{r_o(t)}=1)\wedge(c\leq CV(r_o(t))))\wedge\\&\{((F_{\delta_u(t)}=1)\wedge(d\leq CV(\delta_u(t))))\vee\\&((F_{\delta_d(t)}=1)\wedge(d\leq CV(\delta_d(t))))\}.\end{aligned}\tag{3.9}$$

式 (3.8) と式 (3.9) の a、b、c、d の値の組み合わせにより、ケース A、B、C、D、E、F、G を設定し、検知精度を比較検討する。ケース A は、式 (3.6) と式 (3.7) として、当初に設定した閾値で、a=0.9、b=0.5、c=0.5、d=0.9 である。このケース A を基準として ±0.05 の値の組み合わせで、他のケースを設定し、これを表 3.12 のケースと閾値の項目に示す。そして、実験データを用いて、検知精度として SN（感度）と SP（特異度）（表 3.6 を参照）を算定して、表 3.12 に各実験の結果を示し、図 3.21 にプロットして、下記に考察する。ケース A、C、D、F の 4 つのケースは、SN が 0.98、SP が 0.99 で共に高くてバランスが良い。ケース B は、SN が 0.99 と最も高いが、SP が 0.98 と低い。逆に、ケース E は、SN が 0.95、SP が 0.98 と共に低い。この結果を、各実験の内容に照合すると下記のことが言える。実験 3-1 サイバー攻撃がある場合のうち、(3-1-2) 検知対象が不正アクセスの被害者側の場合では、閾値を厳しくすると見逃しが増え、閾値を緩やかにすると見逃しは無くなる。実験 3-2 検知対象が大容量ファイル転送の受信処理を行う場合（セキュリティ違反として）のうち、(3-2-1) 検知対象にて中程度のリスクが予測される場合では、閾値を厳しくすると見逃しが増える。実験 3-3 サイバー攻撃がなく平時処理を行う場合（セキュリティ許可として）のうち、(3-3-2) 検知対象がファイル（10MiB）受信処理を行う場合では、閾値を緩やかにすると誤検知が発生する。(3-3-3) 検知対象がファイル（10MiB）送信処理を行う場合では、閾値を厳しくすると誤検知が減り、閾値を緩やかにすると誤検知が増える。以上のことから、閾値

を厳しくすると不正アクセスでは誤検知が減るが攻撃での見逃しが増えてしまうこと、閾値を緩くすると攻撃では見逃しが減るが不正アクセスでは誤検知が増えてしまうこと、が明らかになった。よって、3.3.2 節で定義した条件 (3-A) と条件 (3-B) における、インシデント判定の式 (3.6) と (3.7) の閾値として、当初の $a = 0.9$、$b = 0.5$、$c = 0.5$、$d = 0.9$ が適切であると言える。

3.3.6 まとめ

実証実験の結果、本提案のホスト型検知手法は、総実験回数 200 回中、見逃しが 1 回、誤検知が 2 回と非常に少なく、検知精度と再現率のバランスを示す F 値が良好な値を示していた。よって、本提案手法が、サイバー攻撃や不正アクセスに対するイベント検知に非常に有用であることが示された。

表 3.4　ホスト型イベント検知手法によるイベント検知結果

		攻撃・不正アクセス有		平時	
		記号	値	記号	値
検知結果	陽性	TP	99	FP	2
	陰性	FN	1	TN	98

凡例: T = true, F = false, P = positive, N =negative

表 3.5　ホスト型イベント検知手法による実験結果のイベントのインシデント判定の条件式の内訳

	実験	式 (3.6) =TRUE	式 (3.7) =TRUE	式 (3.6) ∨ 式 (3.7) =TRUE	合計
攻撃・不正アクセス有	(3-1-1)	19	19	20	99
	(3-1-2)	1	19	19	
	(3-1-3)	16	20	20	
	(3-2-1)	20	18	20	
	(3-2-2)	20	20	20	
平時	(3-3-1)	0	0	0	2
	(3-3-2)	0	0	0	
	(3-3-3)	2	1	2	

表 3.6　ホスト型イベント検知手法の評価指標の定義と実験結果

指標	定義式	値
感度（SN: sensitivity)	$TP/(TP+FN)$	0.99
特異度（SP: specificity)	$TN/(FP+TN)$	0.98
陽性尤度比（LR+: Likelihood Ratio Positive)	$SN/(1-SP)$	49.5
陰性尤度比（LR-: Likelihood Ratio Negative)	$(1-SN)/SP$	0.01
オッズ比（OR: Odds Ratio)	$LR+/LR-$	4950
適合率（P: precision)	$TP/(TP+FP)$	0.98
再現率（R: recall	$TP/(TP+FN)$	0.99
F 値（F-measure)	$(2P\times R)/(P+R)$	0.98

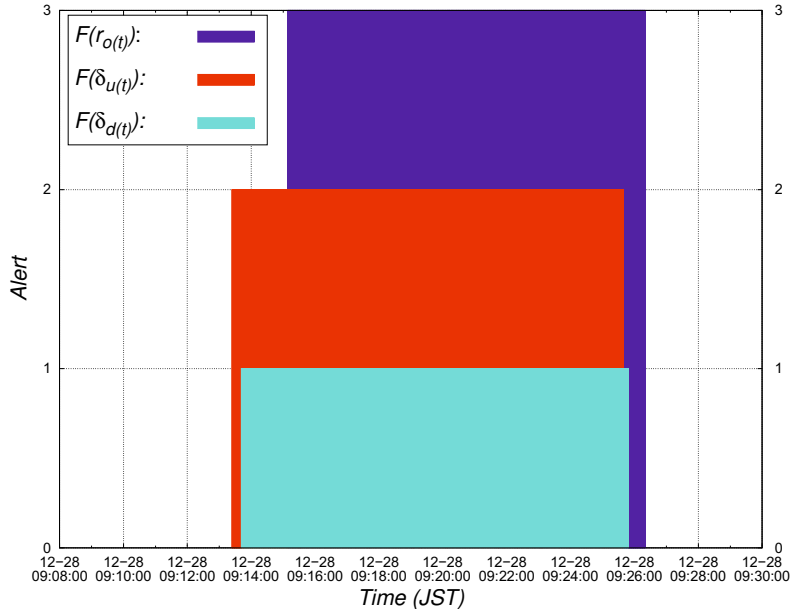

図 3.8　実験 (3-1-1) ホスト型イベント検知手法によるイベント検知結果例

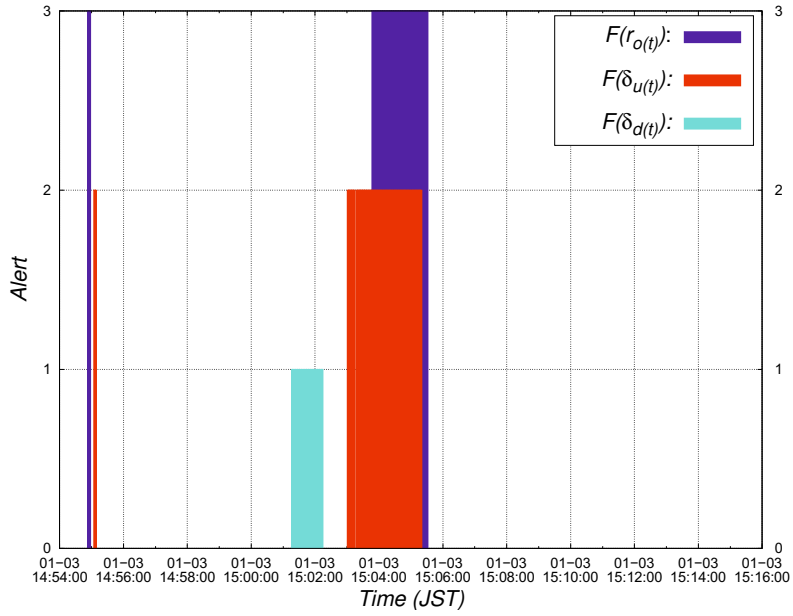

図 3.9　実験 (3-1-2) ホスト型イベント検知手法によるイベント検知結果例

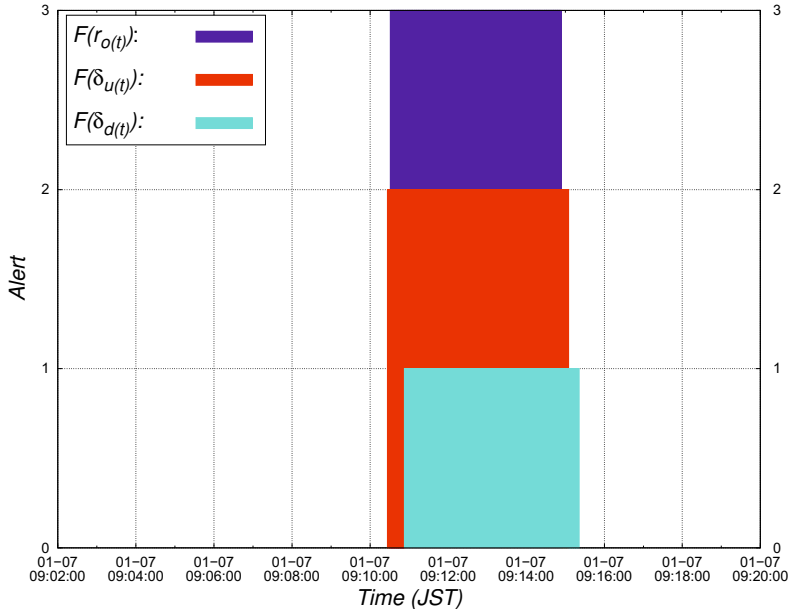

図 3.10　実験 (3-1-3) ホスト型イベント検知手法によるイベント検知結果例

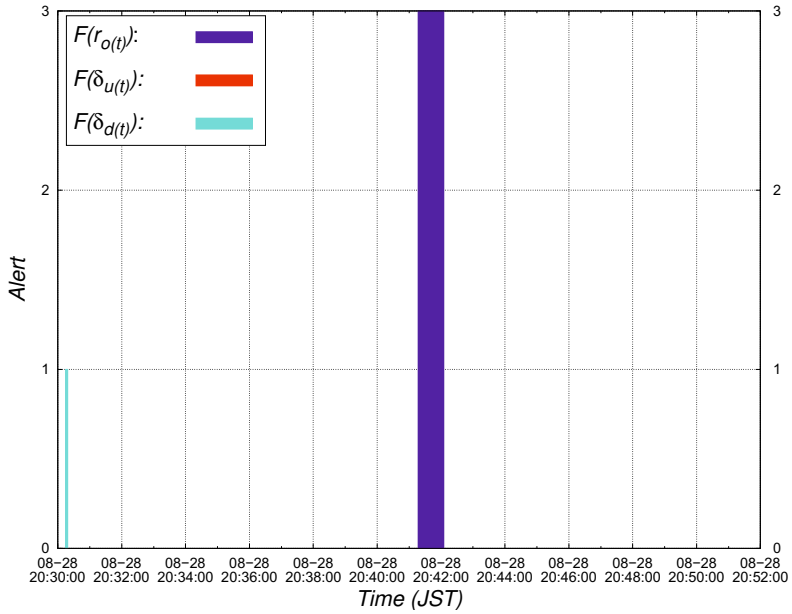

図 3.11　実験 (3-3-2) ホスト型イベント検知手法によるイベント検知結果例

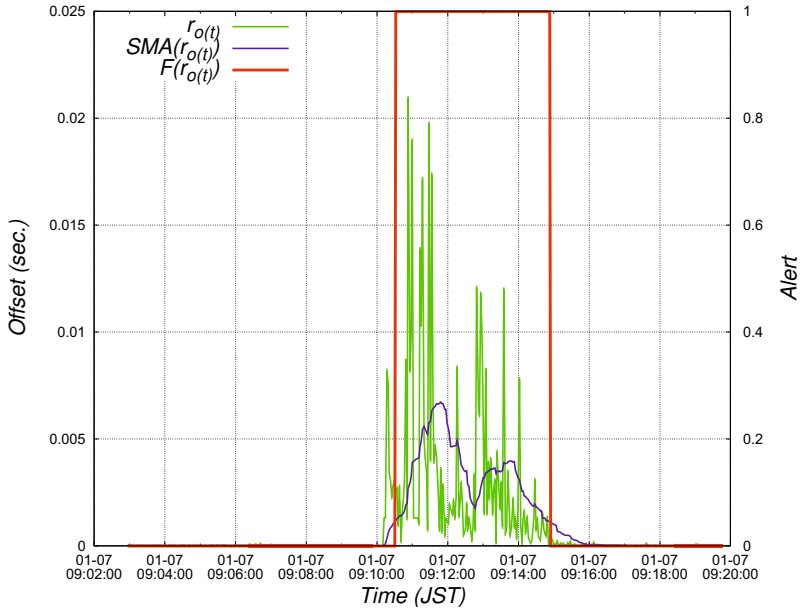

図 3.12　実験 (3-1-3) ホスト型イベント検知手法による $F(r_o(t))$ 結果例

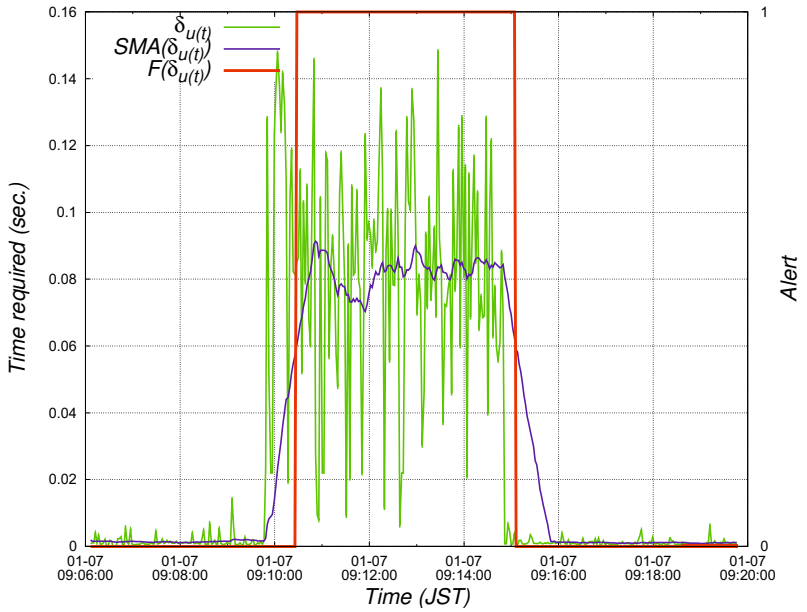

図 3.13　実験 (3-3-2) ホスト型イベント検知手法による $F(\delta_u(t))$ 結果例

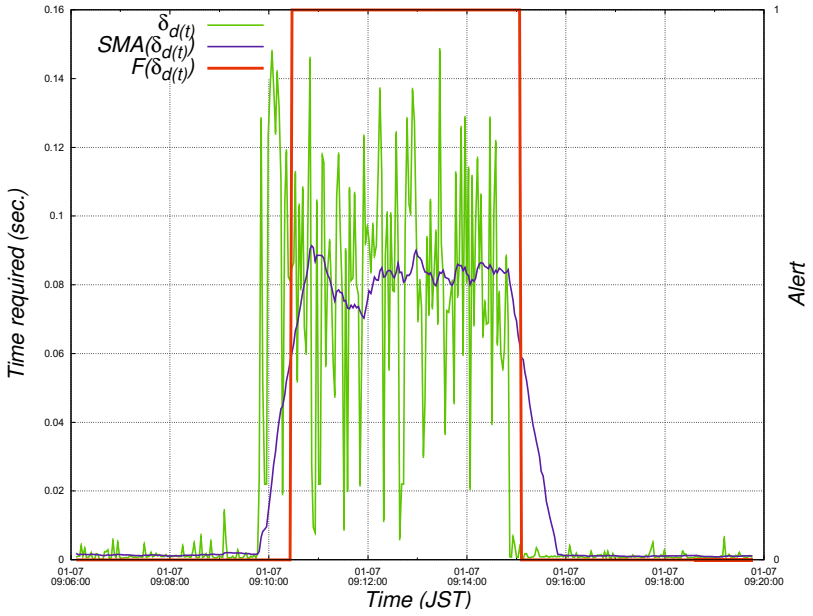

図 3.14　実験 (3-3-2) ホスト型イベント検知手法による $F(\delta_d(t))$ 結果例

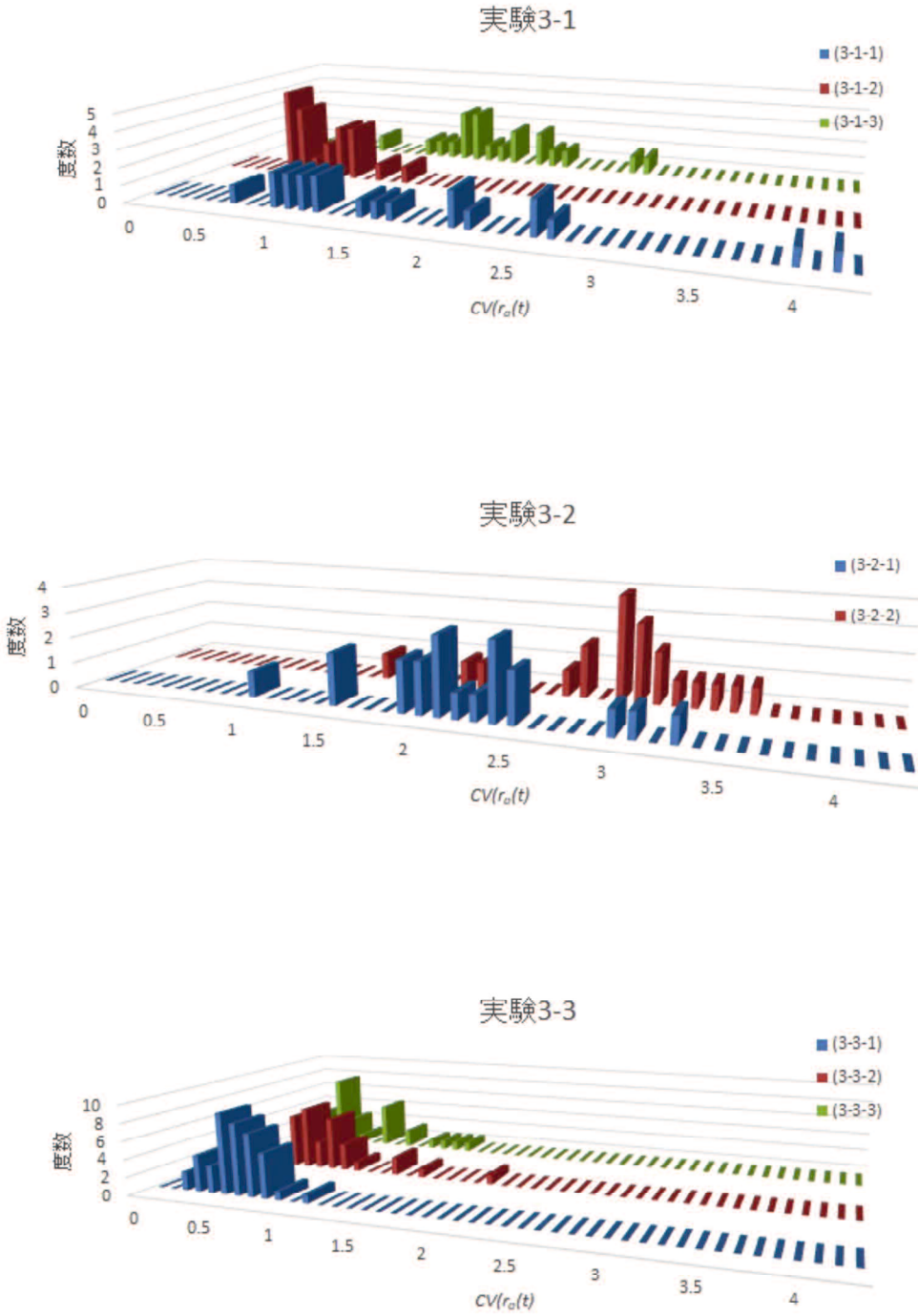

図 3.15　$CV(r_o(t))$ のヒストグラム

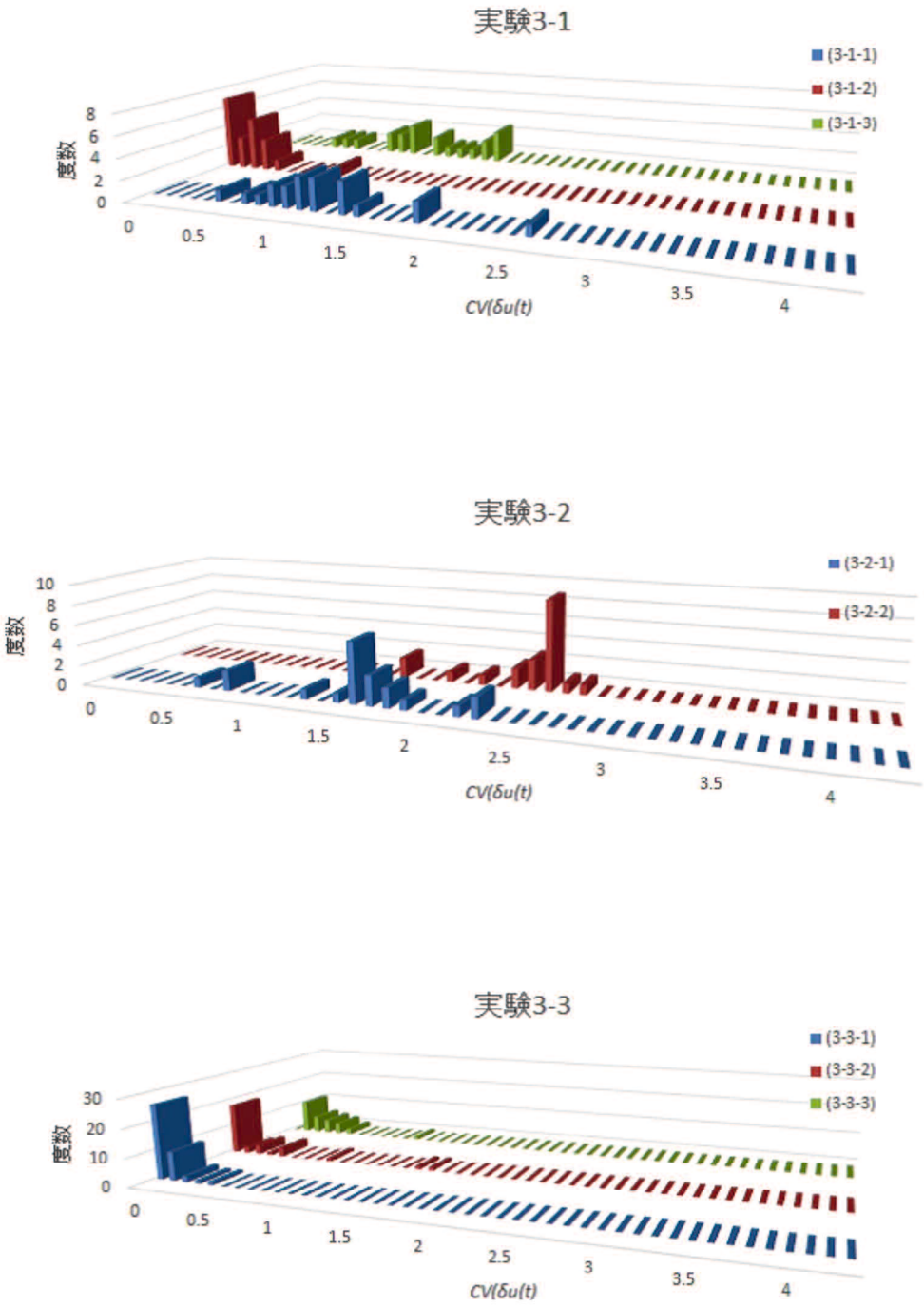

図 3.16　$CV(\delta_u(t))$ のヒストグラム

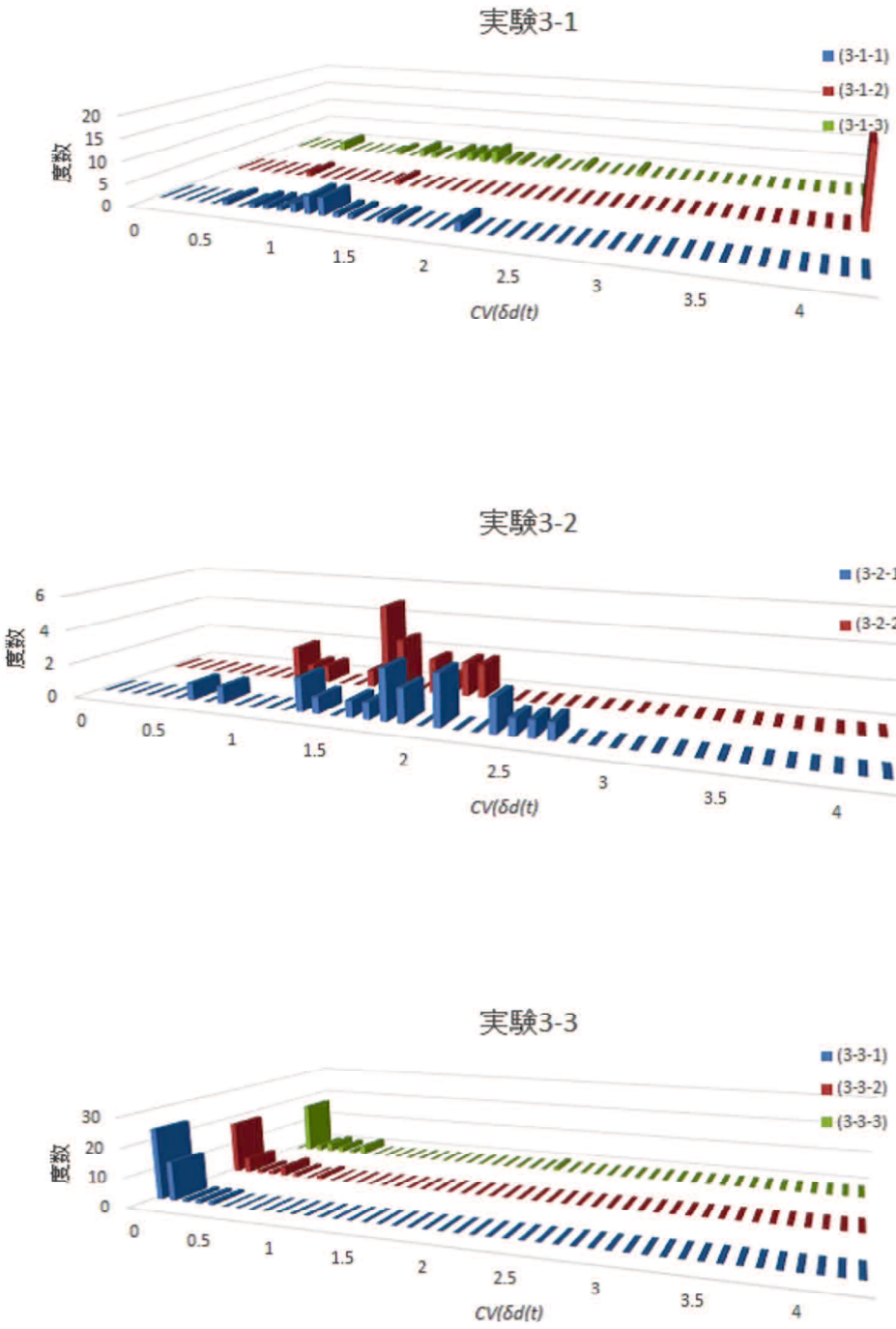

図 3.17　$CV(\delta_d(t))$ のヒストグラム

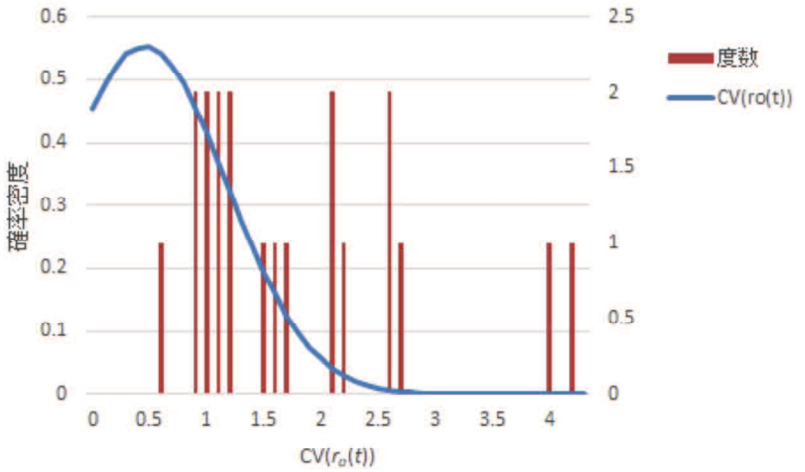

図 3.18　実験 (3-1-1) の $CV(r_o(t))$ のヒストグラムと正規曲線

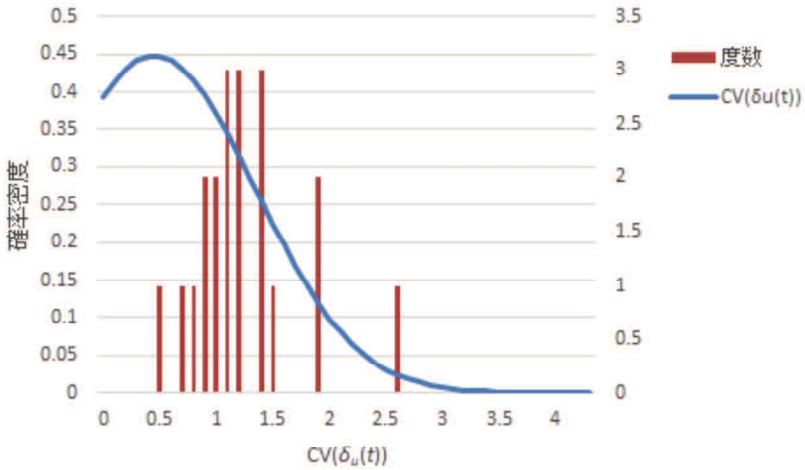

図 3.19　実験 (3-1-1) の $CV(\delta_u(t))$ のヒストグラムと正規曲線

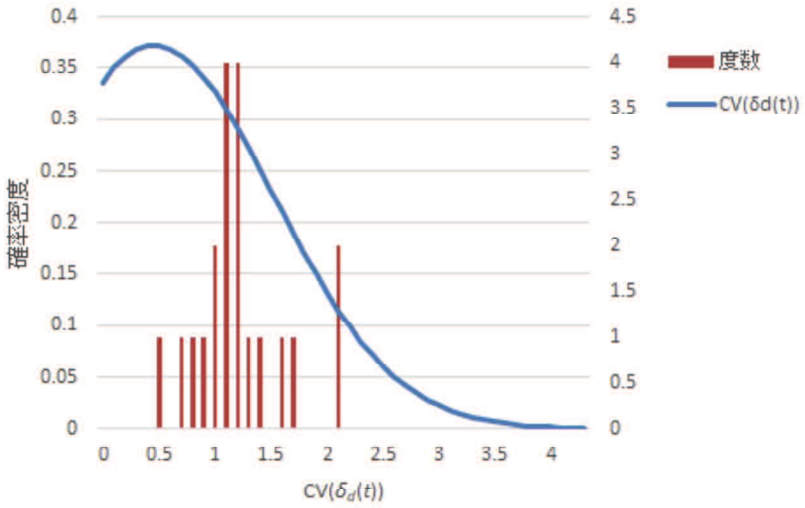

図 3.20　実験 (3-1-1) の $CV(\delta_d(t))$ のヒストグラムと正規曲線

表 3.7　実験 (3-1-1) の基本統計量

	$CV(\delta_u(t))$	$CV(\delta_u(t))$	$CV(\delta_d(t))$
平均	0.4545	0.4545	0.4545
標準誤差	0.1100	0.1358	0.1434
標準偏差	0.7299	0.9010	0.9512
分散	0.5328	0.8118	0.9049
尖度	0.1906	2.6580	7.5400
歪度	1.2924	1.9406	2.6836

表 3.8　実験 (3-1-1) と実験 (3-3-1) の 2 標本の t 検定

有意水準=0.05	変数 1: 実験 (3-1-1)	変数 2: 実験 (3-3-1)
平均	1.8632	0.6144
分散	1.0143	0.0398
観測数	20	40
プールされた分散	0.3590	
仮説平均との差異	0	
自由度	58	
t	7.6101	
$P(T <= t)$ 片側	1.38E-10	
t 境界値 片側	1.6716	
$P(T <= t)$ 両側	2.76E-10	
t 境界値 両側	2.0017	

表 3.9　実験 (3-1-1) と実験 (3-3-1) の 2 標本の F 検定

有意水準=0.05	変数 1: 実験 (3-1-1)	変数 2: 実験 (3-3-1)
平均	1.8632	0.6144
分散	1.0143	0.0398
観測数	20	40
自由度	19	39
観測された分散比	25.4705	
$P(F <= f)$ 片側	2.63E-16	
F 境界値 片側	1.8599	

表 3.10　ホスト型イベント検知手法による実験結果 $CV(P(t))$ のデータ範囲

	実験	データ範囲	$CV(r_o(t))$	$CV(\delta_u(t))$	$CV(\delta_d(t))$
攻撃・不正アクセス有	(3-1-1)	<0.5	0	0	0
		[0.5,0.9)	1	3	3
		>=0.9	19	17	17
	(3-1-2)	<0.5	0	19	0
		[0.5,0.9)	12	0	1
		>=0.9	8	1	19
	(3-1-3)	<0.5	0	1	2
		[0.5,0.9)	1	2	0
		>=0.9	19	17	18
	(3-2-1)	<0.5	0	0	0
		[0.5,0.9)	0	3	2
		>=0.9	20	17	18
	(3-2-2)	<0.5	0	0	0
		[0.5,0.9)	0	0	0
		>=0.9	20	20	20
平時	(3-3-1)	<0.5	9	40	40
		[0.5,0.9)	29	0	0
		>=0.9	2	0	0
	(3-3-2)	<0.5	0	27	28
		[0.5,0.9)	22	1	2
		>=0.9	8	2	0
	(3-3-3)	<0.5	15	27	25
		[0.5,0.9)	9	2	4
		>=0.9	6	1	1

表 3.11　ホスト型イベント検知手法による実験結果 $F(P(t))$ のフラグ

	実験	フラグ	$F(r_o(t))$	$F(\delta_u(t))$	$F(\delta_d(t))$
攻撃・不正アクセス有	(3-1-1)	1	20	20	20
		0	0	0	0
	(3-1-2)	1	20	1	20
		0	0	19	0
	(3-1-3)	1	20	19	18
		0	0	1	2
	(3-2-1)	1	20	20	20
		0	0	0	0
	(3-2-2)	1	20	20	20
		0	0	0	0
	(3-2-3)	1	20	20	20
		0	0	0	0
平時	(3-3-1)	1	22	0	0
		0	18	40	40
	(3-3-2)	1	25	1	2
		0	5	29	28
	(3-3-3)	1	13	3	5
		0	17	27	25

表 3.12　ホスト型イベント検知手法における閾値条件による検知精度の比較

ケース	閾値		実験 3-1			実験 3-2		実験 3-3			実験合計	TP	FP
			1	2	3	1	2	1	2	3			
		回数	20	20	20	40	30	20	20	30	200		
A	a:0.9	検知	20	19	20	20	20	0	0	0	99		
	b:0.5	見逃し	0	1	0	0	0	0	0	0	1		
	c:0.5	誤検知	0	0	0	0	0	0	0	2	2		
	d:0.9											0.99	0.98
B	a:0.9	検知	19	19	20	20	20	0	0	0	98		
	b:0.55	見逃し	1	1	0	0	0	0	0	0	2		
	c:0.5	誤検知	0	0	0	0	0	0	0	1	1		
	d:0.95											0.98	0.99
C	a:0.9	検知	20	19	20	20	20	0	0	0	99		
	b:0.45	見逃し	0	1	0	0	0	0	0	0	1		
	c:0.5	誤検知	0	0	0	0	0	0	0	2	2		
	d:0.85											0.99	0.98
D	a:0.85	検知	20	19	20	20	20	0	0	0	99		
	b:0.5	見逃し	0	1	0	0	0	0	0	0	1		
	c:0.45	誤検知	0	0	0	0	0	0	0	2	2		
	d:0.9											0.99	0.98
E	a:0.95	検知	20	15	20	20	20	0	0	0	95		
	b:0.5	見逃し	0	5	0	0	0	0	0	0	5		
	c:0.55	誤検知	0	0	0	0	0	0	0	2	2		
	d:0.9											0.95	0.98
F	a:0.85	検知	20	19	20	20	20	0	0	0	99		
	b:0.45	見逃し	0	1	0	0	0	0	0	0	1		
	c:0.45	誤検知	0	0	0	0	0	0	0	2	2		
	d:0.85											0.99	0.98
G	a:0.95	検知	19	15	20	20	20	0	0	0	94		
	b:0.55	見逃し	1	5	0	0	0	0	0	0	6		
	c:0.55	誤検知	0	0	0	0	0	0	0	1	1		
	d:0.95											0.94	0.99

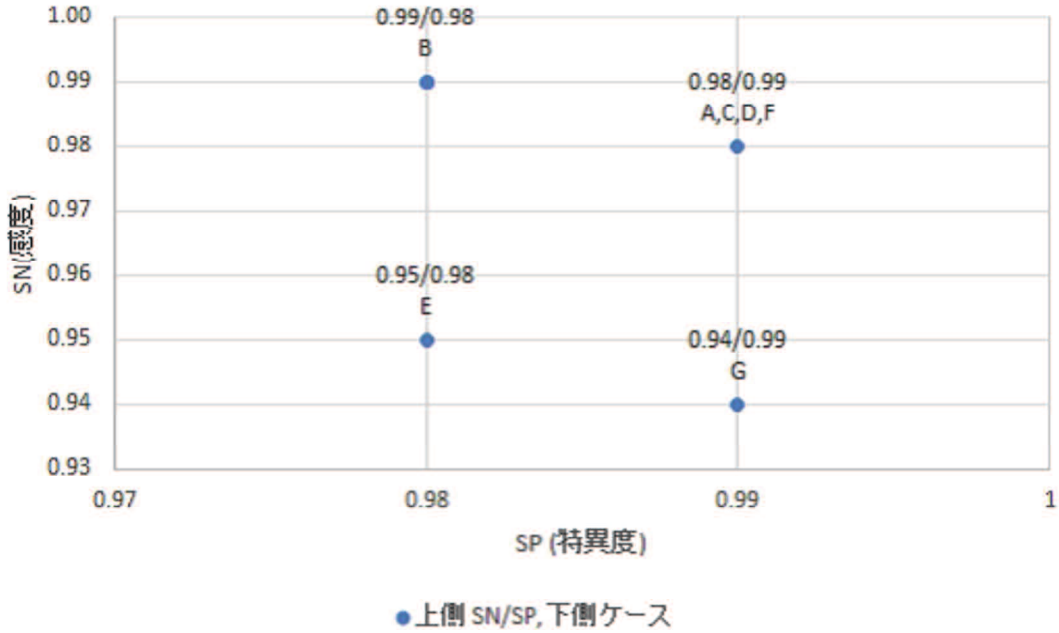

図 3.21　実験 (3-1-1) ホスト型イベント検知手法における閾値条件による検知精度

第 4 章

ネットワーク時刻同期技術を用いたネットワーク型イベント検知手法

4.1　はじめに

本章の目的は、時刻同期技術を用いたネットワーク型のイベント検知手法を提案することである。ネットワーク型の場合、外部からの攻撃を検知するために、一次時刻同期サーバとの往復通信の遅延時間の揺らぎを計測し、内部からの攻撃を検知するために、直上位の時刻同期サーバとの往復通信の遅延時間の揺らぎを計測することにより、イベント検知を行う。本手法に基づく検知モジュールを IoT ルータに組込み、第 3 章と同様の実証実験を行い、本手法の有効性を示す。

4.2　ネットワーク型イベント検知手法

4.2.1　原理

この節では、まずネットワーク型イベント検知手法の目的と検知場所、検知に利用する NTP データの採取方法を述べ、ネットワーク型イベント検知手法の原理について述べる。ネットワーク型イベント検知手法の目的は、検知場所として検知対象の IoT デバイス以外のノード、検知対象の IoT への不正アクセスや攻撃によるイベントを間接的に検知することである。この手法は、検知対象の IoT デバイスに第 4 章で提案したホスト型イベント検知手法を組み込めない場合や、組織において情報セキュリティ担当者が組織内の IoT を監視したい場合の手法である。本手

法の検知場所は、検知対象の IoT デバイスが直接リンクし、本提案手法に基づく検知モジュールを組み込み可能な無線アクセスポインタ、無線ルータ、有線スイッチ、有線ルータの通信処理装置を想定している。本提案では、無線アセスポンインタを用いる。無線アクセスポイントは、無線アクセスポイントに接続している全ノードから送受信される通信データを LAN や WAN に中継する。無線アクセスポイントが中継する通信データには、NTP のリクエストとリプライの NTP データが含まれている。無線アクセスポイントでは、NTP データのヘッダに含まれる送信元及び送信先の MAC アドレスと IP アドレスを識別することで、特定の IoT デバイスが送受信する NTP データを採取できる。ネットワーク型イベント検知手法の原理は、検知対象の IoT デバイスと NTP サーバ間の NTP データを傍受し、解析して、検知対象の IoT デバイスへの WAN 側と LAN 側からの不正アクセスや攻撃のイベントを検知することである。この原理は、第 4 章で述べたように、攻撃対象の IoT デバイスが不正アクセスや攻撃により、NTP データの通信遅延の揺らぎと時刻のずれが発生することを利用している。WAN 側から検知対象の IoT デバイスへの攻撃は、検知対象の IoT デバイスと一次参照元間の NTP データの往復通信時間の遅延時間と（ルート遅延時間、4-2-1 を参照）の揺らぎから検知する。LAN 側から検知対象の IoT デバイスへの攻撃は、検知対象の IoT デバイスとその直上位 NTP サーバ間での NTP データのリクエストとリプライの遅延時間の揺らぎから検知する。

4.2.2 検知手法

IoT デバイス向けに、時刻同期技術を利用したネットワーク型イベント検知手法を提案する。本提案手法では、検知対象 IoT デバイスが無線接続する無線アクセスポイントにおいて、4-2-2 に示した測定したルート遅延時間（$\delta_{Ru}(t)$) と、リクエストの遅延時間（$\delta_u(t)$）とレスポンスの遅延時間（$\delta_d(t)$）を用いてイベントを検知し、それがインシデントに至る可能性の高い（インシデントの予兆を示す）イベントであるかどうかを判定する。観測するデータを図 4.2 に示す。直上位の NTP サーバにリクエストを送信し、各データを取得する間隔を F_{req} とする。提案手法では、F_{req} 毎に、以下に示す手順を繰り返す。

1. データの記録

時刻 t において取得したデータを $\delta_{Ru}(t)$、$\delta_u(t)$、$\delta_d(t)$ として記録する。ノード内に、直近の N 個のデータを保持するものとする。

2. 基本統計情報の算定

   記録した各データについて、平均 $\mu_{\delta_{Ru}}$、μ_{δ_u}、μ_{δ_d} および標準偏差 $\sigma_{\delta_{Ru}}$、σ_{δ_u}、σ_{δ_d} を算定する。

3. ノイズの平準化

   $\delta_{Ru}(t)$、$\delta_u(t)$、$\delta_d(t)$ のうち、$\sigma_{\delta_{Ru}}$、σ_{δ_u}、σ_{δ_d} の 2σ（約 96%）の信頼区間を外れた値はノイズとし、それぞれ $\mu_{\delta_{Ru}}$、μ_{δ_u}、μ_{δ_d} に置換する。

4. 移動平均法による統計情報の算定

   1 スパンのデータ数を n とし、スパン内の移動平均 $SMA(\delta_{Ru}(t))$、$SMA(\delta_u(t))$、$SMA(\delta_d(t))$ を算定する。ただし、移動平均 $SMA(P(t))$ は次式で定義される。

$$SMA(P(t)) = \sum_{i=0}^{n-1} \frac{P(t-i)}{n}. \tag{4.1}$$

   直近の N 個の移動平均 $SMA(P(t'))$ から、標準偏差 $\sigma_{SMA(\delta_{Ru}(t))}$、$\sigma_{SMA(\delta_u(t))}$、$\sigma_{SMA(\delta_d(t))}$、および、変動係数 $CV(\delta_{Ru}(t))$、$CV(\delta_u(t))$、$CV(\delta_d(t))$ を算定する。ただし、$t' = t, t-1, \cdots, t-(N-1)$ であり、変動係数 $CV(P(t))$ は次式で定義される。

$$CV(P(t)) = \frac{\sigma_{SMA(P(t))}}{SMA(P(t))}. \tag{4.2}$$

5. イベントの判定

   $\delta_{Ru}(t)$、$\delta_u(t)$、$\delta_d(t)$ の 1 スパン内で、それぞれ、$\sigma_{SMA(\delta_{Ru}(t))}$、$\sigma_{SMA(\delta_u(t))}$、$\sigma_{SMA(\delta_d(t))}$ の σ（約 68%）の信頼区間を外れた値が連続した場合、そのスパンをイベントと判定する。それぞれにおいて、イベントか否かを表すフラグを $F_{\delta_{Ru}(t)}$、$F_{\delta_u(t)}$、$F_{\delta_d(t)}$ とし、イベントであれば 1、そうでなければ 0 を記録する。

6. イベントのインシデント判定

   各イベントのフラグと変動係数から、以下の条件（c）または条件（d）に当てはまる場合に、検知したイベントをインシデントの予兆であると判定する。

条件（c）　式 (4.3) が成立

$$
\begin{aligned}
&(F_{\delta_{Ru}(t)} = 1 \wedge 0.5 \leq CV(\delta_{Ru}(t))) \wedge \\
&(F_{\delta_u(t)} = 1 \wedge 0.9 \leq CV(\delta_u(t))) \vee \\
&(F_{\delta_d(t)} = 1 \wedge 0.9 \leq CV(\delta_d(t))).
\end{aligned} \tag{4.3}
$$

条件（d）　式 (4.4) が成立

$$
\begin{aligned}
&(F_{\delta_{Ru}(t)} = 1 \wedge 0.05 \leq CV(\delta_{Ru}(t)) \wedge \\
&\{(F_{\delta_u(t)} = 1 \wedge 0.06 \leq CV(\delta_u(t)) \vee \\
&(F_{\delta_d(t)} = 1 \wedge 0.06 \leq CV(\delta_d(t)))\}.
\end{aligned} \tag{4.4}
$$

ネットワーク型イベント検知手法では、検知対象の IoT のイベントをルート遅延時間、NTP リクエストの遅延時間、NTP レスポンスの遅延時間から間接的に検知する。式 (4.3) は、サイバー攻撃の特性を考慮したものである。DDoS 攻撃の被害側の場合および踏み台攻撃の加害者側の場合は、ルート遅延時間の変動係数が 0.5 以上で変動があり、かつ NTP リクエストの遅延時間または NTP レスポンスの遅延時間の変動係数が 0.9 以上で変動が大きい場合、インシデントに紐づくイベントと判断する。式 (4.4) は、脆弱試験や平時のファイル転送の特性を考慮したものである。この場合、線アクセスポイントが、攻撃元ノードと検知対象の IoT デバイス間の大量データ転送のバッファとなり、ネットワーク通信負荷は緩衝され、軽減されるため、検知対象の IoT デバイスのシステム負荷は微弱なため、、ルート遅延時間の変動係数が 0.05 以上で、かつ NTP リクエストの遅延時間または NTP レスポンスの遅延時間の変動係数が 0.06 以上であれば、インシデントに紐づくイベントと判断する。本提案手法の利点は、以下のとおりである。本手法では、汎用的な時刻同期技術で取得できる情報を活用するため、本提案手法は、特別な装置や技術を必要としない。さらに、本提案手法は、検知対象のノード以外で検知するため、検知対象の IoT には全く負荷を掛けず、検知モジュールが稼働するノードにも検知のための負荷をほとんど発生させない。また、本提案手法に用いる NTP は、フィルタリング装置や他の検知装置を通過させるのが一般的であるため、ネットワークトポロジや他のノードの配置などの要因による妨害は受けにくい。本提案手法を用いた 4.2.3 で述べる検知用モジュールのバイナリファイルの合計容量は約 170KiB、検知用モジュールの物理メモリサイズ消費量（Resident Set Size）は約 7MiB、仮

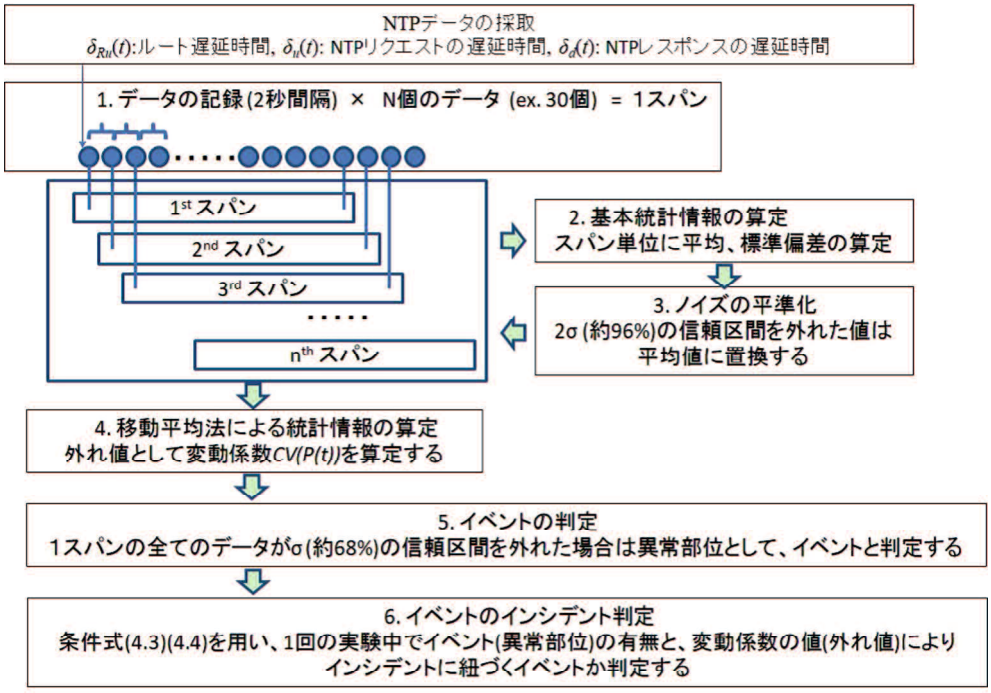

図 4.1　ネットワーク型イベント検知手法の手順

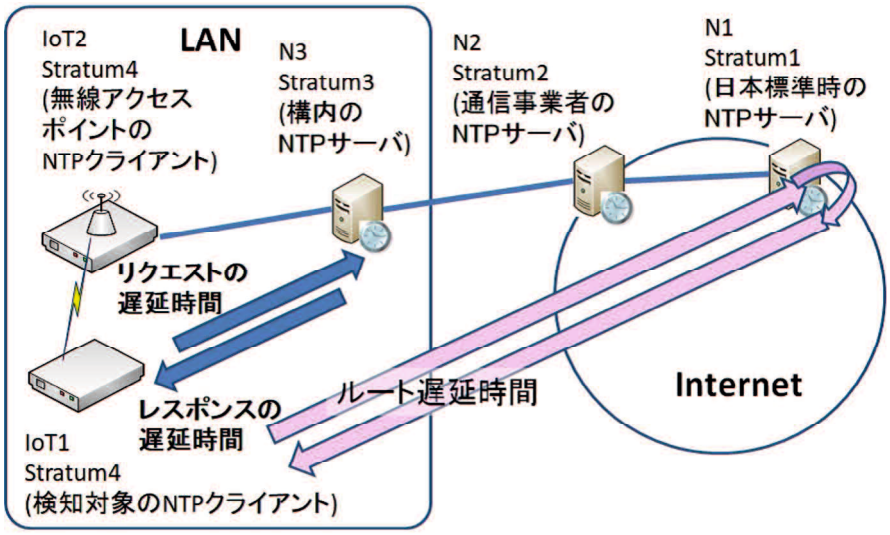

図 4.2　ネットワーク型検知手法で観測する値

想メモリ領域（Virtual Set Size）は約 8MiB と小さく、本手法はメモリ容量に乏しい IoT デバイス向きである。本提案手法によるネットワーク型検知モジュールは、ホスト型検知モジュールと同時に検知が可能なため、検知対象 IoT デバイスだけでなく無線アクセスポイントでもイベント検知を可能にしている。

4.2.3 実験目的と方法

本実験では、本提案のネットワーク型イベント検知手法により、検知対象の IoT に対して疑似的にサイバー攻撃を行った場合に、インシデントの予兆としてのイベントを正しく検知できるか、また、攻撃のない平時において誤検知があるかを確認し、検知手法としての有効性を検証することを目的とする。本実験のサイバー攻撃は、表 2.4 に示したデータリンク層への攻撃として PoD（Ping of Death）による DDoS 攻撃と、アプリケーション層への探査としての不正な脆弱性試験、アプリケーション層への攻撃として標的型大容量ファイル添付メール転送とした。実験の狙いは、上記の順に、大量パケット受信によるネットワークの過負荷、各種アプリケーションへの不正アクセスによる CPU 負荷の増大、大容量ファイルの受信による通信処理の増大とディスク書き込みによる CPU の割り込み処理増大である。

本提案手法が、検知対象とする装置の環境に求める要件を、以下に述べる。ソフトウェア環境に求められる要件は、Linux 系 OS または Windows 系 OS であること。ネットワーク時刻同期モジュール（chronyd または ntpd）が稼働していること。パケットスニファの API が提供されること。例として、Linux 系 OS の場合は libpcap 、Windows 系 OS の場合は WinPcap が挙げられる。通信環境に求められる要件は、検知対象の IoT の隣接していること。ハードウェア環境に求められる要件は、通信処理機器（無線アクセスポイント、有線スイッチなど)、汎用コンピュータまたは IoT デバイスであること。実験環境を図 4.3 に示す。図 3.1 の時刻同期の階層構造は、Stratum1 の日本標準時刻サーバを時刻の一次時刻源として、構内ネットワーク内に Stratum2 のローカル時刻同期サーバを置き Stratum1 に接続する、無線アクセスポイント及び検知対象の IoT デバイスは共に Stratum3 の時刻同期クライアントとして Stratum2 に接続する。図 4.3 の環境では、外部の攻撃者 N_W が標的 IoT デバイス T_{L1-IoT} を直接攻撃する、または、内部のマルウェア M_L が T_{L2-IoT} を踏み台として間接的に攻撃する 2 通りの攻撃ルートを想定する。実験環境のノードを表 4.1 に、ネットワーク接続を表 4.2 に示す。T_{L1-IoT}、T_{L2-IoT} 及び T_{L3-IoT} には、ボードコンピュータ Raspberry Pi を用いる。T_{L3-IoT} において、開発した検知モジュールを稼働させる。この検知モジュールは、主に C 言語を用いて開発した。N_W から T_{L1-IoT} へ不正アクセスは、脆弱性検知スキャナである Nessus[158] を使用する。N_W から T_{L1-IoT} への DDoS 攻

撃は、ネットワークテストコマンドである siege を使用する。T_{L2-IoT} 上の M_L は踏み台として N_W からの遠隔操作により TL へ、ab で DDoS 攻撃を行う。実験は、以下のとおり、サイバー攻撃がある場合（実験 4-1、4-21）と平時の場合（実験 4-3）について行った。

実験 4-1: サイバー攻撃がある場合

(4-1-1) 検知対象が PoD による DDoS 攻撃の被害者側の場合

N_W から T_{L1-IoT} に疑似的に DDoS 攻撃を実行。実験開始 10 分後に、以下のコマンドを 1 回実行 ab -n 100000 -c 100 TL1-IoT の IP アドレス。

(4-1-2) 検知対象が不正アクセスの被害者側の場合

N_W から T_{L1-IoT} に脆弱性試験ツールを利用して疑似的に不正アクセスを実行。実験開始 10 分後に、以下のツールを 1 回実行 Nessus（Web Application Scanning、T_{L1-IoT} の IP アドレス

(4-1-3) 検知対象が PoD による DDoS 攻撃の踏み台（加害者）側の場合

M_L から TL に疑似的に DDoS 攻撃を実行。実験開始 10 分後に、以下のコマンドを 1 回実行 ab -n 100000 -c 100 TL の IP アドレス。

実験 4-2: 検知対象が大容量ファイル転送の受信処理を行う場合（セキュリティ違反として）

(4-2-1) 検知対象にて中程度のリスクが予測される場合

TL から T_{L1-IoT} へ ftp で 100MiB のファイル転送を、実験開始 10 分後に、1 回実行

(4-2-2) 検知対象にて高いリスクが予測される場合

TL から T_{L1-IoT} へ ftp で 1GiB のファイル転送を、実験開始 10 分後に、1 回実行。

実験 4-3: サイバー攻撃がなく平時処理を行う場合（セキュリティ許可として）

(4-3-1) 平時処理のみの場合

(4-3-2) 検知対象がファイル受信処理を行う場合

TL から T_{L1-IoT} へ ftp で 10MiB のファイル転送を、実験開始 10 分後に、1 回実行。

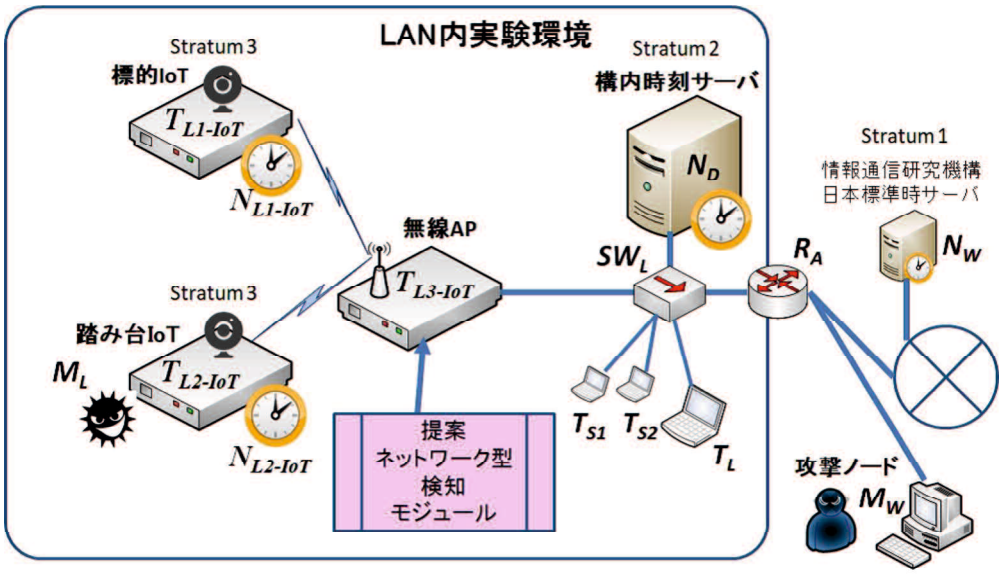

図 4.3　ネットワーク型検知手法に向けた実験環境

(4-3-3)　検知対象がファイル送信処理を行う場合

T_{L1-IoT} から TL へ ftp で 10MiB のファイル転送を、実験開始 10 分後に、1 回実行。

1 回の実験時間は 20 分間とした。試行回数は、不正アクセス・攻撃があった場合の実験として、実験 4-1 は各 20 回で合計 60 回、実験 4-2 は各 20 回で合計 40 回の合計 100 回とする。平時の場合の実験として、実験 4-3 は (4-3-1) で 40 回、(4-3-2)、(4-3-3) で 30 回の合計 100 回とする。本実験では、1 回の実験の終了時点で計測されたデータに対して、検知手順 (5) と (6) によってイベントのインシデント判定を行うものとする。ただし、実験の終了時点で N 個の移動平均が揃っていない場合は、その分の値は無視して計算するものとする。

4.2.4　実験結果

実験結果を表 4.3～表 4.5 に示す。表 4.3 は、実験 4-1 と 4-2 において、インシデントとしてイベントを検知した回数を TP に、検知できなかった（見逃し）回数を FN に集計し、実験 4-3 において、誤ってイベントを検知した（誤検知）回数を FP に、検知しなかった回数を TN に集計している。表 4.4 は、各実験で成立した条件式の内訳を示している。表 4.3 の結果より、攻撃・不正アクセス有と平時で、それぞれ 100 回の実験のうち、見逃しは 3 回、誤検知は 7 回であり、ほとんどの場合に

表 4.1　ネットワーク型イベント検知手法に向けたノードの仕様

ノード	CPU（Clock）	メモリ	OS（Kernel）
T_{L1-IoT}、T_{L2-IoT}、T_{L3-IoT}	ARM1176JZF-S ARMv6l（700MHz）	512MiB	Raspbian 7.8（4.1.13+）
T_L、T_{S1}、T_{S2}	AMD Athlon X2 5000B（2.6GHz）	8GiB	Fedora 21（4.1.13）
M_W	AMD Athlon II X2 220（2.8GHz）	10GiB	Fedora 21（4.1.13）

表 4.2　ネットワーク型イベント検知手法に向けたネットワーク接続

区間	リンク	規格	最大速度
T_{L1-IoT}、$T_{L2-IoT} \rightleftharpoons T_{L3-IoT}$	Wiress LAN	IEEE802.11n	300Mbps
T_{L1-IoT}、N_D T_L、T_{S1}、T_{S2} $M_W \rightleftharpoons R_A$	Wired LAN	IEEE802.3	1Gbps
$R_A \rightleftharpoons$ WAN	FTTH	IEEE802.3u	100Mbps

おいて、インシデントの予兆であるイベントを正しく検知しているが、平時においてイベントの誤検知が少しある。表 4.6 は、4.2.2 で示した検知手順 4. で変動係数 $CV(\delta_R u(t)))$、$CV(\delta_u(t)))$、$CV(\delta_d(t)))$ を算定した値を範囲分けして示している。表 4.7 は、4.2.2 で示した検知手順 5. により 1 回以上フラグが 1 になった場合の実験数を示している。

実験結果の詳細は、表 4.4 と表 4.6 の結果より、攻撃・不正アクセス有の場合、DDoS の踏み台攻撃は全回で、式 (4-3) と式 (4-4) を共に全て満たし、1GiB の大容量ファイルの転送で式 (4-4) を全て満たし、全回イベントを検知している。DDoS 攻撃、脆弱性試験、100MiB の大容量ファイルの転送で各 1 回を除き式 (4-4) を満たし、イベントを検知している。よって、ネットワーク型検知手法の攻撃・不正アクセスによるイベント検知精度は高い。平時の場合、平時処理のみは全回で $CV(\delta_{Ru}(t))$、$CV(\delta_u(t))$、$CV(\delta_d(t))$ は 0.05 以下で誤検知は全く無いが、10Mib のファイルの送受信において 0.05 以上で式 (4-4) を満たし 7 回が誤検知である。

理由として、1 回の実験時間である 20 分間の移動平均法の区間で統計的にインシデントと判定できるほどのネットワーク負荷の有意性はないが、極めて短時間に長いデータを受け取ることが予想されるためである。例えば、BSSID のほか電波強度や各種アプリケーション毎への GPS 位置情報のダウンロードなどが想定される。表 4.3 の実験結果を定量的に評価するため、パターン認識で取り扱われている陽性尤度比（LR+）、陰性尤度比（LR-）、オッズ比（OR）及び F 値を算出して、表 4.5 に各指標の定義式と値を表す。表 4.5 より、LR+、LR-とも基準値以上または以下の値を示している。OR の値が 430 であること は、正しく検知する割合が誤検知をする割合の 430 倍であることを示している。また、適合率、再現率及び F 値から、正確性と網羅性のバランスもよい。

イベント検知結果の表示例として、実験 (4.4)、の結果を、図 3.8 に示す。同図では、14:33:06〜14:53:01 の 20 分間で監視を行い、14:43:12〜14:48:44 の間に DDoS 攻撃を行った結果を示している。14:43:23〜14:48:32 の間で $F(\delta_{Ru}(t))$（図 4.5 参照）、$F(\delta_u(t))$（図 4.6 参照）および $F(\delta_d(t))$（図 4.7 参照）のいずれかでイベントを検知し、攻撃を行った時間帯とほぼ一致している。そのうちの 14:44:30〜14:48:30 の間で、3 つともイベントが重なっており、式 (4.4)) が成立し、インシデントの予兆を示すイベントであると判定している。

4.2.5 考察

本節では、ネットワーク型イベント検知手法を用いたモジュールによる無線アクセスポイントへのネットワーク負荷およびシステム負荷について、検知モジュールのリソースの消費量を比較して考察する。比較する検知モジュールは、ネットワーク型イベント検知手法を用いた本モジュール、システム監視として 2.3.3 で述べたシステム監視として sar コマンド、IDS として Snort（ver.2.9.4）とする。条件として監視する同時接続の検知対象数は、業務用コントローラー型無線アクセスポイントの場合は 50 基程度、民生用自立型線アクセスポイントの場合は 10 基程度とする。ただし、無線規格は IEEE802.11b/g/n および IEEE802.11ac、最大通信速度は 300Mbps〜800Mbps とする。本検知モジュールは、専用の通信データを発生させず、検知にはネットワーク時刻同期モジュールが送受信する NTP データを、ネットワークインタフェースから採取する。採取するデータ量は、NTP リクエス

トと NTP リプライ共に 90Byte（Ethernet: 14Byte、IP:20Byte、UDP: 8Byte、NTP: 48Byte 表 3.1 を参照）を、検知対象ノード単位に 2 秒間隔で、1 分間当たり 5,400Byte となり、軽微でネットワークにもシステムに影響を与えない。本検知モジュールは、前記通信データを 4.2.2 で示したメモリ空間内で記憶、処理されるので、別途ファイルとして保管する必要はないが、アーカイブデータとして記録する場宛は 1 日あたり約 8MiB を消費する。アーカイブデータの比較として、平時の 1 日あたり、本検知モジュールでは約 8MiB、sar コマンドでは約 720MiB、Snor では約 100MiB、が消費され sar コマンドの消費は非常に大きい。ただし、本検知モジュールおよび sar コマンドは通信データ量に依存しないが、Snort では通信データが記録されるため通信データ量に依存し、DDoS 攻撃を受けた場合は膨大なリソースが消費される。物理メモリサイズ消費量（Resident Set Size）の比較は、本検知モジュールが約 7MiB、sar コマンドが約 3MiB、snort が約 57MiB で、前者 2 つは同程度だが、Snort の消費は非常に大きい。仮想メモリ領域（Virtual Set Size）は、本検知モジュールが約 8MiB、sar コマンドが約 5MiB、Snort が約 319MiB で、前者 2 つは同程度だが、Snort の消費は極めて大きい。以上のことから、本検知モジュールは他の検知手法のモジュールと比較してもネットワーク負荷およびシステム負荷は軽微と言える。まとめネットワーク型イベント検知では、不正アクセスによる揺らぎの変動幅がホスト型に比べ狭いため、ホスト型よりイベント検知条件を一部厳しくした。総実験回数 200 回中、見逃しが 3 回、誤検知が 7 回と少なかったが、不正アクセスの場合、変動幅が小さいため誤検知の可能性がある。実験の評価指標では、検出精度と再現率のバランスを示す F 値が良好な値を示し、本提案手法が、リアルタイムのイベント検知に有用であることが示された。

表 4.3　ネットワーク型イベント検知手法による実験結果のイベントのインシデント判定

		攻撃・不正アクセス有		平時	
		記号	値	記号	値
検知結果	陽性	TP	97	FP	7
	陰性	FN	3	TN	93

凡例: T = true, F = false, P = positive, N =negative

表 4.4　ネットワーク型イベント検知手法による実験結果のイベントのインシデント判定の条件式の内訳

	実験	式 (4.3) =TRUE	式 (4.4) =TRUE	式 (4.3) ∨ 式 (4.4) =TRUE	合計
攻撃・不正アクセス有	(4-1-1)	19	19	20	99
	(4-1-2)	1	19	19	
	(4-1-3)	16	20	20	
	(4-2-1)	20	18	20	
	(4-2-2)	20	20	20	
平時	(4-3-1)	0	0	0	2
	(4-3-2)	0	0	0	
	(4-3-3)	2	1	2	

表 4.5　ネットワーク型イベント検知手法の評価指標の定義と実験結果

指標	定義式	値
感度（SN: sensitivity）	$TP/(TP+FN)$	0.97
特異度（SP: specificity）	$TN/(FP+TN)$	0.93
陽性尤度比（LR+: Likelihood Ratio Positive）	$SN/(1-SP)$	13.86
陰性尤度比（LR-: Likelihood Ratio Negative）	$(1-SN)/SP$	0.03
オッズ比（OR: Odds Ratio）	$LR+/LR-$	430
適合率（P: precision）	$TP/(TP+FP)$	0.93
再現率（R: recall）	$TP/(TP+FN)$	0.97
F 値（F-measure）	$(2P\times R)/(P+R)$	0.95

表 4.6　ネットワーク型イベント検知手法による実験結果 $CV(P(t))$ のデータ範囲

	実験	データ範囲	$CV(\delta_{Ru}(t))$	$CV(\delta_u(t))$	$CV(\delta_d(t))$
攻撃・不正アクセス有	(4-1-1)	<0.5	1	1	0
		[0.05,0.5)	1	0	0
		[0.5,0.9)	18	0	0
		>=0.9	0	19	20
	(4-1-2)	<0.5	20	0	0
		[0.05,0.5)	0	20	20
		[0.5,0.9)	0	0	0
		>=0.9	0	0	0
	(4-1-3)	<0.5	0	0	0
		[0.05,0.5)	0	0	0
		[0.5,0.9)	20	0	0
		>=0.9	0	20	20
	(4-2-1)	<0.5	1	0	0
		[0.05,0.5)	1	20	20
		[0.5,0.9)	18	0	0
		>=0.9	0	0	0
	(4-2-2)	<0.5	0	0	0
		[0.05,0.5)	20	20	20
		[0.5,0.9)	0	0	0
		>=0.9	0	0	0
平時	(4-3-1)	<0.5	33	29	0
		[0.05,0.5)	7	11	40
		[0.5,0.9)	0	0	0
		>=0.9	0	0	0
	(4-3-2)	<0.5	21	16	0
		[0.05,0.5)	9	14	30
		[0.5,0.9)	0	0	0
		>=0.9	0	0	0
	(4-3-3)	<0.5	20	16	0
		[0.05,0.5)	10	14	30
		[0.5,0.9)	0	0	0
		>=0.9	0	0	0

表 4.7　ネットワーク型イベント検知手法による実験結果 $F(P(t))$ のフラグ

	実験	フラグ	$F(\delta_{Ru}(t))$	$F(\delta_u(t))$	$F(\delta_d(t))$
攻撃・不正アクセス有	(4-1-1)	1	19	20	19
		0	1	0	1
	(4-1-2)	1	19	20	19
		0	1	0	1
	(4-1-3)	1	20	20	20
		0	0	0	0
	(4-2-1)	1	19	20	20
		0	1	0	0
	(4-2-2)	1	20	20	20
		0	0	0	0
平時	(4-3-1)	1	8	23	39
		0	32	17	1
	(4-3-2)	1	11	20	30
		0	19	10	0
	(4-3-3)	1	11	23	30
		0	19	7	0

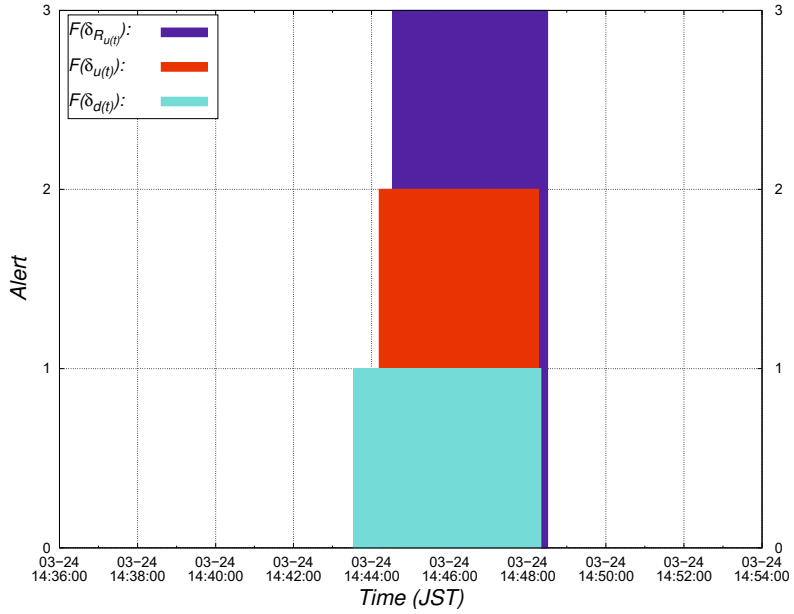

図 4.4　実験 (4-1-1) ネットワーク型イベント検知手法によるイベント検知結果例

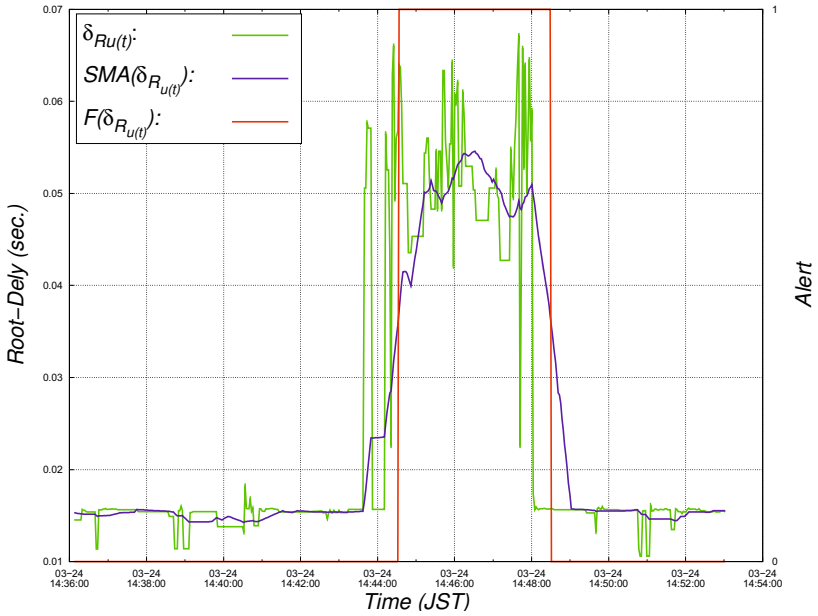

図 4.5　実験 (4-1-1) ネットワーク型イベント検知手法による $F(\delta_{Ru}(t))$ 結果例

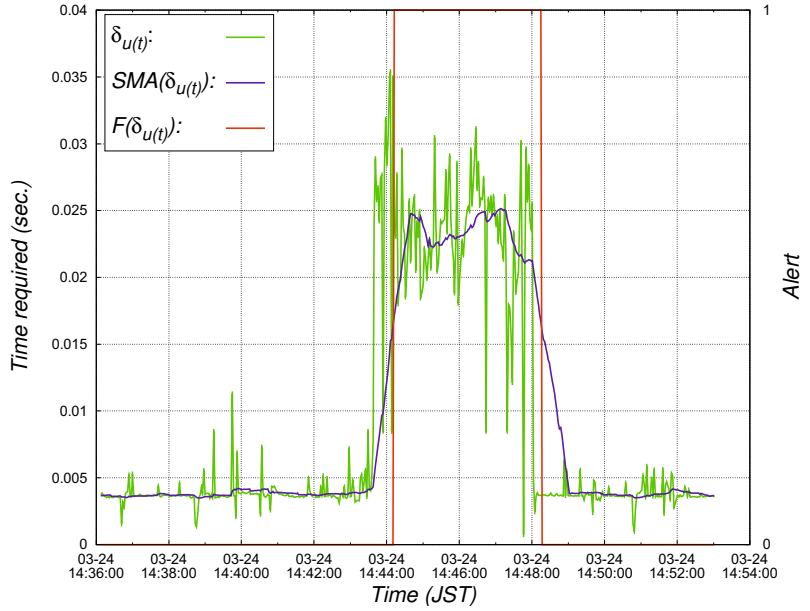

図 4.6　実験 (4-1-1) ネットワーク型イベント検知手法による $F(\delta_u(t))$ 結果例

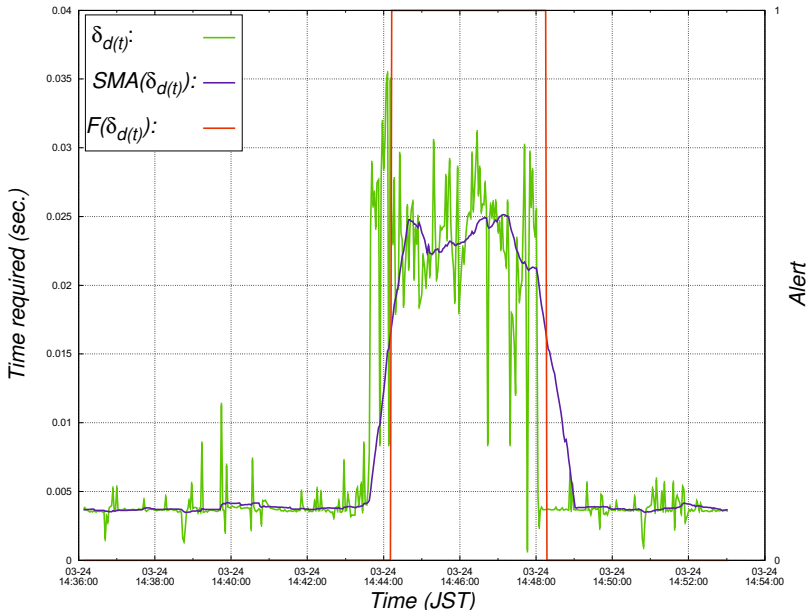

図 4.7　実験 (4-1-1) ネットワーク型イベント検知手法による $F(\delta_d(t))$ 結果例

第 5 章

結言

本論文では、IoT デバイスに向けられたサイバー攻撃を検知するために、IoT デバイスへ組み込み可能な精度のよい軽量なイベント検知手法を提案した。

第 2 章では、本論文が IoT デバイスの検知手法を提案する前提として、IoT に関連するセキュリティ対策の国際規格、攻撃手口及び従来の検知手法とその問題点、IoT デバイスにおける従来の検知手法の有用性の確認として、以下のことを示した。セキュリティに関係する国際規格や各国のガイドラインでは、セキュリティの規範として、従来は事故後を検知するインシデント検知が中心であったが、インシデント検知では対策が間に合わず、被害にまで至ってしまうため、最近はインシデントの予兆であるイベント検知を求めている。攻撃対象として、従来は対象外であった社会インフラ、産業インフラ、個人生活のシステムが、近年ではサイバー攻撃の標的になっている。特に、IoT 環境に関連した攻撃手口として、特に DDoS 攻撃が深刻な被害を発生させている。従来の検知手法は、多種多様な攻撃の手口をバターン認識などを用いるため、システムリソースを大量に消費し、新たな攻撃手口に対応するため緊急的や定期的な更新や保守が前提となっているため、IoT デバイスの検知手法としては課題がある。従来の検知手法が IoT デバイスへの攻撃に有効であるか予備実験により確認したが、イベント検知として有効ではなかった。以上のことから、現在のところ、IoT デバイスのセキュリティ対策に求められるイベント検知手法が存在せず、実装方法も明確になっていないことを示した。

第 3 章では、ホスト型イベント検知手法を提案し、実証実験により有効性を評価した。本検知手法は、ネットワーク時刻同期技術を用い、検知対象の IoT デバイスでイベントを直接的に検知する。本検知手法の特徴は、以下の通り。実証実験

の結果として、見逃しや誤検知が少なく、検知結果の正確性、網羅性、信頼性が高く、バランスも良く検知精度が高い。イベント判定は、2 秒単位に直近の 1 分間について短時間で行える。パケットデータの解析を行わないので、暗号化通信にも対応できる。検知モジュールのモジュールサイズは約 160Kib、物理メモリサイズは約 7Mib、仮想メモリサイズは約 8Mib で軽量である。定期的な更新や保守作業を必要としないこと。以上のことから、本提案手法が、サイバー攻撃や不正アクセスに対するイベント検知に有用であることを示した。

第 4 章では、ネットワーク型イベント検知手法を提案し、実証実験により有効性を評価した。本検知手法は、ネットワーク時刻同期技術を用い、検知対象の IoT デバイスがリンクする無線アクセスポイントや有線スイッチでイベントを間接的に検知する。本検知手法の特徴は、以下の通り。ホスト型イベント検知手法が、既存の IoT デバイスで新たに組み込めない場合、検知対象の IoT デバイスに制約があり組み込めない場合でも、検知対象の IoT のイベントを検知できること。実証実験の結果として、見逃しや誤検知が少なく、検知結果の正確性、網羅性、信頼性が高く、バランスも良く検知精度が高い。イベント判定は、2 秒単位に直近の 1 分間について短時間で行える。パケットデータの解析を行わないので、暗号化通信にも対応できる。検知モジュールのモジュールサイズは約 170Kib、物理メモリサイズは約 7Mib、仮想メモリサイズは約 8Mib で軽量である。定期的な更新や保守作業を必要としないこと。以上のことから、本提案手法が、サイバー攻撃や不正アクセスに対するイベント検知に有用であることを示した。

付録 A では、汎用的なシステム監視が、IoT デバイスにおけるイベント検知手法として有効であるかを予備実験により、以下のことを示した。システム監視により、IoT デバイスのシステムリソースの変化や変動から異常を検出することは可能であるが、見逃しや誤検知も発生しイベント検知手法として精度がよくないことが確認された。また、異常の原因判断するためには、検知対象のログファイルと過去の正常時と異常時のアーカイブファイルとを比較して判別する必要があるが IoT デバイスにはその記憶容量を確保することは困難であった。以上のことから、従来の検知手法を IoT デバイスのイベント検知として適用することには問題があることを示した。

まとめとして、本論文の要件に対する結果を以下に示す。検知精度は、検知率 97

～99%、見逃し率 1～ 3%、誤検知率 2 ～ 7%、と正確性、網羅性、信頼性が高く、バランスも良い。検知モジュールは、バイナリファイルのサイズ約 160KiB、物理メモリサイズ消費量約 7MiB、仮想メモリ領域約 8MiB、と軽量である。イベント判定は、2 秒単位に直近の 1 分間の判定が可能である。暗号化通信にも対応できる。定期的な更新や保守作業を前提としない。以上のことから、本論文では、IoT デバイスに対するサイバー攻撃のイベント検知において、検知場所に応じた検知手法の提案して、軽量な検知モジュールの開発を行い、実証実験より一定の検知精度を確認できたことで、実用化に向けた一歩が踏み出せたと考える。今後は、検知対象を、家電、制御機器、スマートデバイスなどの多種多様な IoT デバイスに広げ、提案検知手法の検証と改良により、汎用性と信頼性の向上を図りたい。

最後に、今後の課題と将来に向けた研究の発展性について述べる。

今後の課題として、ネットワーク型イベント検知手法における不正アクセスの検知手法の改良と不正アクセスに対する検知精度の向上がある。ネットワーク型イベント検知手法における不正アクセスの検知手法の改良として、検知のために測定したルート遅延時間、NTP リクエストの遅延時間、NTP レスポンスの遅延時間の値の範囲から、検知条件の閾値を自動的に設定する機能追加がある。理由は、一般的に不正アクセスは検知されるのを困難にするために、攻撃時に比べてシステム負荷やネットワーク負荷が抑制されため、ネットワーク型イベント検知手法が測定する値とその範囲は小さくなり、不正アクセスの手口により検知条件の閾値を調整する必要性があるためである。さらに、不正アクセスに対する検知精度の向上として、不正アクセスに悪用されるポートアクセス監視の機能追加がある。本検知手法には既に、パケットキャプチャ機能を保有しており、ポートアクセス監視の機能追加は容易である。不正アクセスで悪用される既知のポート例として、不正な遠隔操作（rpc #111/tcp）、不正な認証（auth #113/tcp）、ワームを悪用し感染活動（epmap #135/tcp、 microsoft-ds #445/tcp）、TLS/SSL の脆弱性試験（https #443/tcp）などがある。

将来に向けた研究の発展性として、攻撃ノードの特定と IoT の安全確認がある。（ 図 5.1 を参照）ただし、これら研究の展開には、IPv6 アドレスの普及と通信プロバイダ、インターネット接続事業者、企業、各組織の協力とフィールド実験が望まれる。攻撃ノードの特定については、本論文のネットワーク型イベント検知手法を

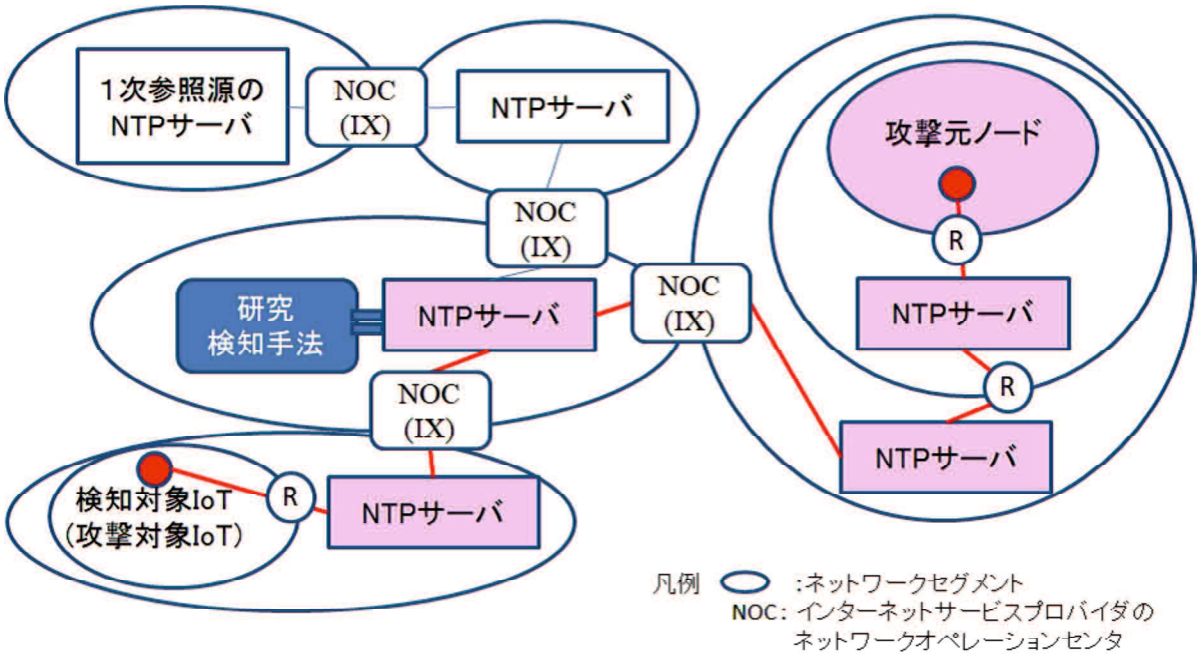

図 5.1　将来に向けた研究の発展性

発展させ、検知対象の IoT デバイスから 1 次参照源までで中継する各階層の NTP サーバ間における NTP データのリクエストおよびリプライ通信時間の遅延時間を収集し、遅延時間が最も大きい階層を特定することで、攻撃ノードが活動するネットワークドメインを絞り込む研究である。IoT の安全確認についても、本論文のネットワーク型イベント検知手法を発展させ、検知対象の IoT デバイスの NTP データのリクエストとレスポンスの同期とイベント検知の情報を、集中監視することで、特定の IoT デバイスの生存確認（I'm Alive.）と、攻撃や不正アクセスを受けている IoT デバイスを特定する研究である。これらの研究は、サイバー攻撃を含めた IoT に関連した事故予防にも寄与できると考える。

付録 A

IoT におけるシステム監視による異常検知

A.1　はじめに

本付録の目的は、予備実験として、IoT デバイスにおいて従来のシステム監視を用いて、検知対象をシステムリソースとした異常検知を行い、検知結果を評価し、イベント検知としての有用性を確認することである。

A.2　原理

予備実験は、IoT デバイスにおいて、平時の場合と疑似攻撃を与えた場合で、システムリソースのシステム監視を行い、監視結果を記録し、そのデータの外れ値、異常部位、変化点から異常を判定する。異常検知の結果は、疑似攻撃を与えた場合の正しい検知と見逃しについて、平時の場合の誤検知について評価する。システム監視のイベント検知として有用性は、異常をイベントとし、疑似攻撃をインシデントして、検知結果からインシデントに紐づくイベントを正しく判定できるかで確認する。具体的には、以下の環境、条件、手順で行う。システム監視には、コンピュータで汎用的な sar コマンドを用いる。検知対象には、システムのパフォーマンスに大きな影響を与えるシステムリソースとして、CPU、メモリ、ディスクなどのデバイス、ネットワークインタフェースの使用率や使用量を用いる。システム監視の結果であるシステムリソースの時系列データは、IoT デバイスにログファイルとして記録する。ログファイルに記録された時系列データにおける外れ値、異常部位、変化点から異常を判定する。さらに、時系列データについて移動平均法による統計処

理を行い、異常を判定する。以上の結果から、検知対象としてシステムリソースの妥当性と、システム監視のイベント検知としての有用性について確認する。

A.3 実験方法

予備実験では、IoT デバイスにおいて、異常検知として汎用的なシステムコマンドである sar を用い、システムリソースの使用率や使用量をログファイルに記録し、記録されたロググファイルのデータを ARMA モデル（自己回帰移動平均モデル）により解析して、イベントを検知できるか評価する。予備実験の環境を、図 A.1、表 A.1、表 A.2 で示す。予備実験で監視するシステムリソースの要素は、システムとユーザの CPU 使用率、物理メモリの使用率、ネットワークインタフェースの 1 秒あたりの受送信データサイズ、ディスク入出力の 1 秒あたりの読み聞き込み入出リクエスト数の 6 種類とする。

予備実験は、システム監視対象の IoT デバイスにおいて、平時の場合と疑似攻撃を与えた場合で、異常検知を行う。平時の場合として、明示的なユーザ処理が無い場合と、Web カメラ監視の場合がある。Web カメラ監視の場合では、動画ストリーミングソフトウェアである mjpg_streamer を稼働させリモートサーバの Web ブラウザからアクセスする。疑似攻撃を与えた場合として、リモート接続による処理の場合、ネットワーク過負荷の場合、システム過負荷の場合がある。リモート接続による処理の場合は、検知対象の IoT デバイスのシステムに接続して不正にリモートアクセスを行う。具体的には、リモートサーバから VNC（Virtual Network Computing）で監視対象の IoT デバイスにリモートログインして、Web ブラウザで Web サイトを閲覧することで通常のネットワークアクセスを行い、続いてエディタで蔵置しているファイルを閲覧することでディスクアクセスを行う。ネットワーク過負荷の場合は、任意の TCP・UDP/IP パケットを送受信できるツール hping3 で疑似的に DDoS 攻撃を行う。システム過負荷の場合は、負荷テストやパフォーマンスに用いられる stress コマンドで疑似的にシステムを過負荷にする。

システム監視は、検知対象の IoT デバイスにおいて、システム監視コマンド sar コマンドを実行し、1 秒間隔でデータを採取して、実験 (A-1-1)（A-1-2)（A-2-1)（A-2-2) は 20 分間（1200 秒間)、実験 (A-2-3) は 23 分間（1380 秒間）収集し、ログファイルに記録する。記録されたログファイルのデータは、ARMA モデル (自

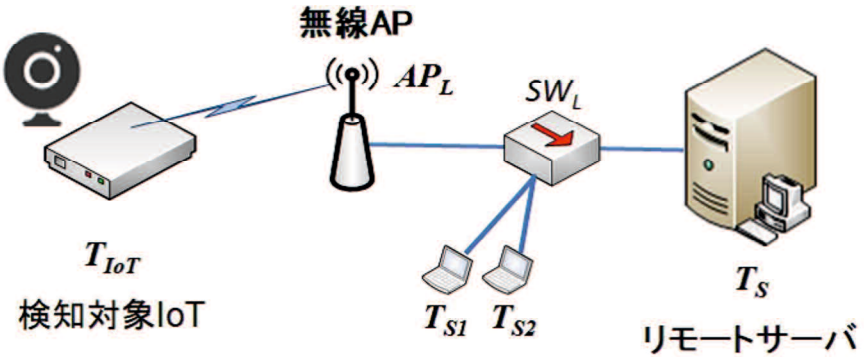

図 A.1　システム監視の環境

表 A.1　システム監視に向けたノードの仕様

ノード	CPU（Clock）	メモリ	OS（Kernel）
T_{IoT}	ARM1176JZF-S ARMv6l（700MHz）	512MB	Raspbian 7.8 （4.1.13+）
T_S T_{S1}、T_{S2}	AMD Athlon X2 5000B（2.6GHz）	8GB	Fedora 21 （4.1.13）

表 A.2　システム監視に向けたネットワーク接続

区間	リンク	規格	最大速度
$T_{IoT} \rightleftharpoons AP_L$	Wiress LAN	IEEE802.11n	300Mbps
T_S、T_{S1}、T_{S2}、AP_L	Wired LAN	IEEE802.3	1Gbps

己回帰移動平均モデル）により、外れ値、異常部位、変化点を判定でき、イベントとして検知できるか評価する。

- 各実験共通のシステム監視

  sar -A -o（ログファイルのディレクトリパス）（間隔）（実験時間）

  -A: 上記システムリソースの要素全て、-o: ログファイル指定間隔: 1 秒単位、実験時間: 20 分間

実験 A-1:　検知対象の IoT デバイスが平時の場合

(A-1-1) 平時の場合

本実験では、検知対象の IoT デバイス T_{IoT} において、実験開始直後から 20 分間システム監視を行うが、その間に実験としての明示的な負荷は与えない。

(A-1-2) Web カメラ監視の場合

本実験では、検知対象の IoT デバイス T_{IoT} において、実験開始直後から 20 分間システム監視を行うが、平時の日常処理として、実験開始 5 分後に、下記コマンドにより Web カメラ監視のサービスを実行して、このサービスに T_{S1} と T_{S2} が 10 分間アクセスしてストリーミング受信を行う。

mjpg_streamer \-i ‘’input_uvc.so -f 10 -r 320x240 -d /dev/video0 -y” \-o ‘’output_http.so -w /usr/local/www -p 8080 -c ID :PW”

実験 A-2: 検知対象の IoT デバイスに疑似攻撃を与えた場合

(A-2-1) リモート接続による処理の場合

本実験は、検知対象の IoT が遠隔操作された場合を想定している。本実験では、検知対象の IoT デバイス T_{IoT} において、実験開始直後から 20 分間システム監視を行うが、疑似攻撃として、実験開始 5 分後に、リモートサーバ T_S から VNC により検知対象の IoT デバイスにリモートログインをして、10 分間に、Web ブラウザを起動してインターネット上の Web サイトを閲覧し、エディタで検知対象の IoT デバイス T_{IoT} に蔵置しているファイルを閲覧することを繰り返して、明示的にシステムに負荷を与える。

(A-2-2) ネットワーク過負荷の場合

本実験では、検知対象の IoT デバイス T_{IoT} において、実験開始直後から 20 分間システム監視を行うが、疑似攻撃として、実験開始 5 分後に、検知対象の IoT デバイス T_{IoT} へ、リモートサーバ T_S からネットワークテストを行う。実験開始 5 分後に、リモートサーバ T_S で下記コマンドを 10 分間実行し、明示的にシステムに負荷を与える。hping コマンドは、送信元 IP アドレスをランダムに変更し、ファイアウォールのス

テートテーブルや他の TCP/IP スタック及びファイアウォールソフトウェア内の各 IP アドレスに基づく動的テーブルに負荷をかける。また、フラグは、SYN flag を ON、TCP 確認応答番号は 0、パケットデータサイズは 512Byte に設定する。

hping3 -I enp0s3 –rand-source -S -L 0 -d 512（検知対象の IoT デバイスの IP アドレス）

(A-2-3) システム過負荷の場合

本実験では、検知対象の IoT デバイス T_{IoT} において、実験開始直後から 23 分間システム監視を行うが、他のシステム処理が無くなるのを待つため 3 分間待機し、疑似攻撃として、実験開始 8 分後に検知対象 IoT デバイス T_{IoT} 内で下記コマンドを実行し、明示的にシステムに負荷を与える。stress コマンドは、10 分間、10 個のワーカで平方根 sqrt() の計算を繰り返し、10 個のワーカでメモリ 256Mbyte（デフォルト）の確保開放処理 malloc()/free() を繰り返し、10 個のワーカでバッファキャッシュ中の更新データのディスク書き込み処理 sync() を繰り返す。

stress –timeout 10m –cpu 10 –vm 512 –io 10

A.4 実験結果

実験結果について、システムが利用している CPU 使用率、ユーザが利用している CPU 使用率、物理メモリの使用率、ネットワークインタフェースの毎秒受信バイト量、ネットワークインタフェースの毎秒送信バイト量、ディスクの 1 秒間の I/O リクエスト数の 6 種類の要素単位に、グラフ化して図 A.2～図 A.31 で示し、下記に分析する。

システムが利用している CPU 使用率

実験 (A-1-1) で約 15%～約 30%、実験 (A-1-2) で約 15%～約 70%、実験 (A3-2-1) で約 15～約 60%、実験 (A-2-2) で数 %～約 20%、実験 (A-2-3) で数 %～約 60% であった。この結果から、システムが利用している CPU 使用率は、平時の場合やネットワーク過負荷の場合においては大きな変化はなく、一定の幅内で変動しているが、その他のテストでは明らかな変化と変動があり、Web カメラ監視の場合と

システム過負荷の場合では大きく変化し大きく変動しており、外れ値、異常部位、変化点として判定できる。

ユーザが利用している **CPU** 使用率

実験 (A-1-1) で約 10%～約 40%、実験 (A-1-2) で約 15%～約 80%、実験 (A-2-1) で約 10%～約 85%、実験 (A-2-2) で数 %～約 90%、実験 (A-2-3) で数 %～約 95% であった。平時の場合においては大きな変化はなく、一定の幅内で変動しているが、その他のテストでは明らかな変化と変動があり、外れ値、異常部位、変化点として判定できる。Web カメラ監視の場合とリモートアクセス接続による処理の場合では大きく変化し大きく変動しているが、両テスト共に、ユーザプロセスとして CPU 使用しており妥当な結果である。リモートアクセス接続による処理の場合、ネットワーク過負荷の場合とシステム過負荷の場合では、テスト終了後に鋭い変動があるが、これは実験以外のユーザプロセスによるものと推測する。

物理メモリの使用率

実験 (A-1-1) 及び実験 (A-1-2) で約 55%～約 60%、実験 (A-2-1) で約 70%～約 100%、実験 (A-2-2) で約 80%～約 90%、実験 (A-2-3) で約 15%～約 85% であった。平時の場合、Web カメラ監視の場合、ネットワーク過負荷の場合では安定または緩やかな微増である。リモートアクセス接続による処理の場合では大きな増加の変化、システム過負荷の場合では明らかな変化と変動があり、外れ値、異常部位、変化点として判定できる。

ネットワークインタフェースの毎秒受信バイト量

実験 (A-1-2) で平均約 5kByte で約 20kByte のピークが数回発生しており、実験 (A-2-1) の平均約 10kByte で約 50kByte と約 80kByte のピークが発生している。Web カメラ監視の場合、リモートアクセス接続による処理の場合で多少の変動があるが、全ての実験で大きな変化はない。

ネットワークインタフェースの毎秒送信バイト量

実験 (A-1-2) で約 50kByte～約 200kByte の変動、実験 (A-2-1) で数 Byte～約 100 kByte の変動がある。平時の場合、ネットワーク過負荷の場合、システム過負荷の場合では大きな変化も変動もない。リモートアクセス接続による処理の場合、

Web カメラ監視の場合で大きな変化と大きな変動がある。

ディスクの 1 秒間の I/O リクエスト数

実験 (A-2-1) で約 20tps～約 400 のピークが頻発し、実験 (A-2-3) で約 10tps～約 90tps の変動と約 200tps～約 300tps のピークが頻発している。平時の場合、Web カメラ監視の場合、ネットワーク過負荷の場合では大きな変化も変動もない。リモートアクセス接続による処理の場合で大きな変化があり、システム過負荷の場合で大きな変化と大きな変動がある。

以上の結果として、ネットワーク過負荷の場合では、いずれのシステムリソースでも、変化や変動はなかった。システム過負荷の場合では、システムが利用している CPU 使用率、物理メモリの使用率、ディスクの 1 秒間の I/O リクエスト数から、外れ値、異常部位、変化点として判定できる。

次に、実験結果を、実験単位に 6 種類の要素の判定からイベントの判定を行う。

(A-1-1) 平時の場合

明示的な負荷がかからないため、システムが利用している CPU 使用率、ユーザが利用している CPU 使用率、物理メモリの使用率、ネットワークインタフェースの毎秒受信バイト量、ネットワークインタフェースの毎秒送信バイト量、ディスクの 1 秒間の I/O リクエスト数の 6 つの要素全てにおいて、異常を検知しない。よって、平時では、異常検知として誤検知せず、イベントも無いと正しく判定できる。

(A-1-2) Web カメラ監視の場合

システムが利用している CPU 使用率、ユーザが利用している CPU 使用率、ネットワークインタフェースの毎秒送信バイト量の 3 つの要素で明確な異常を検知した。物理メモリの使用率、ネットワークインタフェースの毎秒受信バイト量、ディスクの 1 秒間の I/O リクエスト数では、異常を検知しない。よって、Web カメラ監視では、異常検知として誤検知が発生し、イベントの判定も正しく行えない。

(A-2-1) リモート接続による処理の場合

6 つ全ての要素で明確な異常を検知した。よって、リモート接続による処理では、正しく異常を検知し、イベントと正しく判定できる。

(A-2-2) ネットワーク過負荷の場合

システムが利用している CPU 使用率、物理メモリの使用率、ディスクの 1 秒間の I/O リクエスト数の 3 つの要素で異常を検知した。よって、ネットワーク過負荷の場合では、正しく異常を検知し、イベントと正しく判定できる。

(A-2-3) システム過負荷の場合

システムが利用している CPU 使用率、物理メモリの使用率、ディスクの 1 秒間の I/O リクエスト数の 3 つの要素で明確な異常を検知した。ユーザが利用している CPU 使用率、ネットワークインタフェースの毎秒受信バイト量、ネットワークインタフェースの毎秒送信バイト量では、異常を検知しない。よって、ストレステストでは、正しく異常を検知し、イベントと正しく判定できる。

以上のことから、実験結果は、(A-1-2) Web カメラ監視の場合を除いて、正しい異常検知とイベントの判断が可能である。しかし、(A-1-2) Web カメラ監視の場合で、誤検知が発生し、イベントの判断も正しく行えないことから、平時の正常な処理を異常と検知し、イベントと判断してしまう危険性が高い。

システムが利用している CPU 使用率（%）

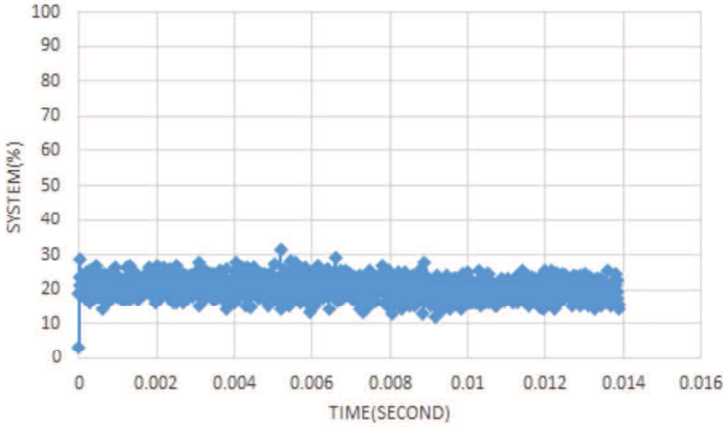

図 A.2　平時の場合

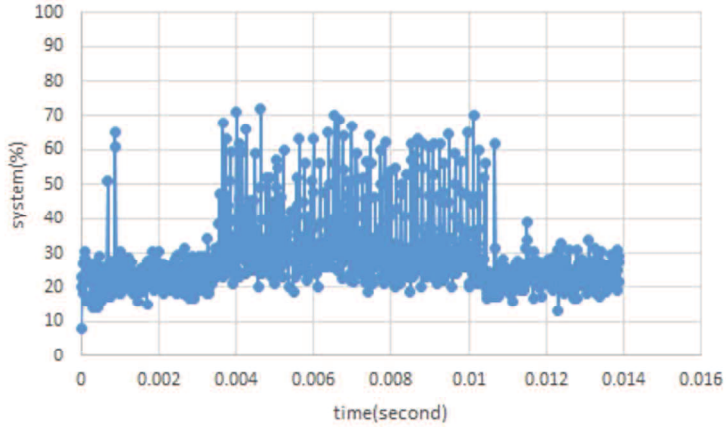

図 A.3　Web カメラ監視の場合

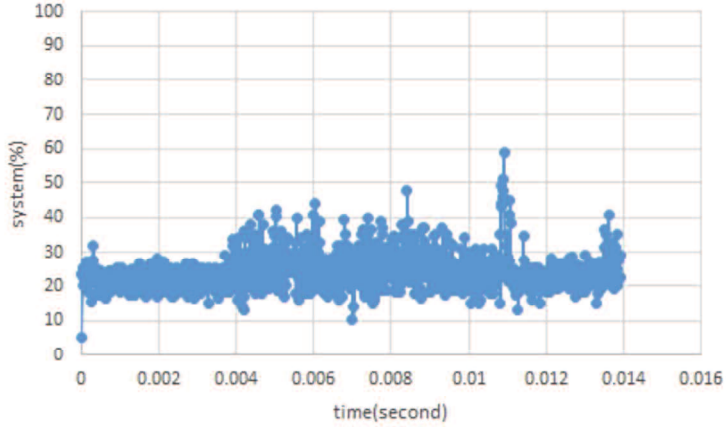

図 A.4　リモート接続による処理の場合

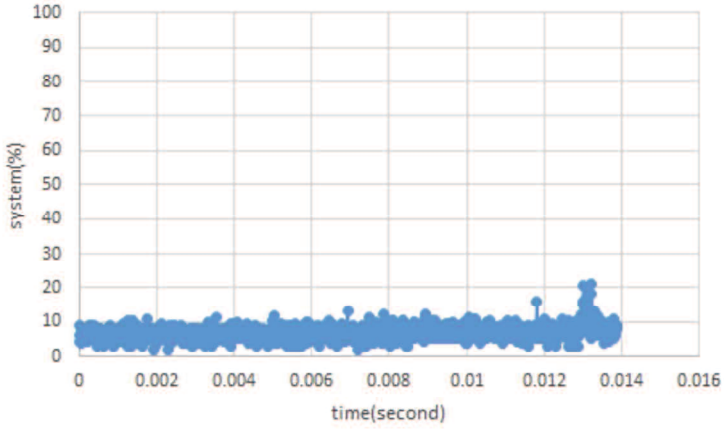

図 A.5　ネットワーク過負荷の場合

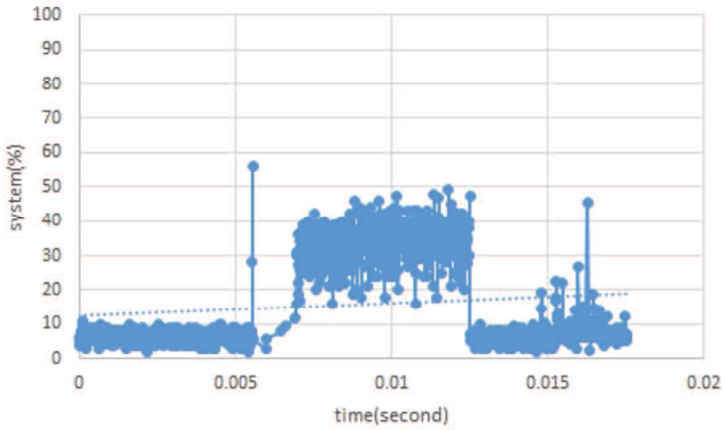

図 A.6　システム過負荷の場合

ユーザが利用している CPU 使用率（%）

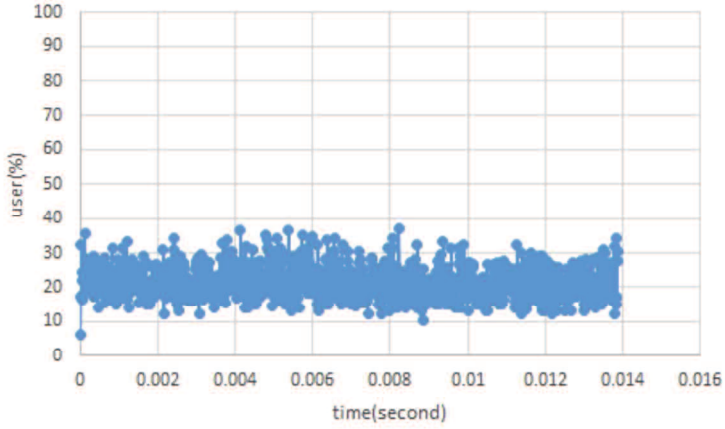

図 A.7　平時の場合

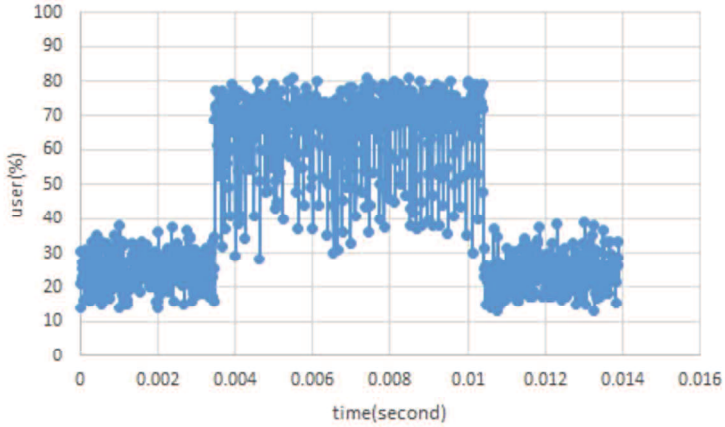

図 A.8　Web カメラ監視の場合

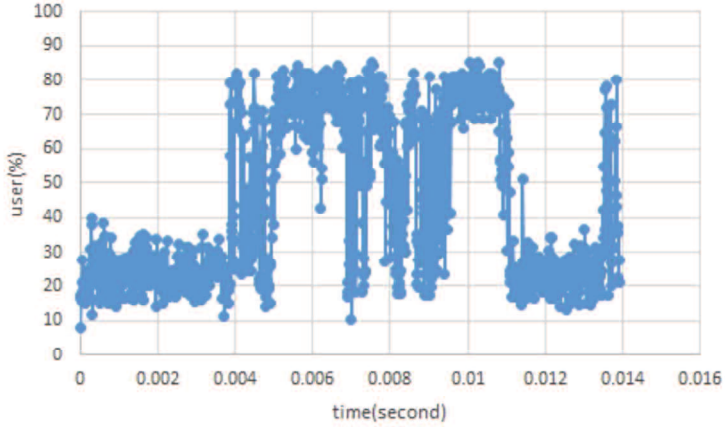

図 A.9　リモート接続による処理の場合

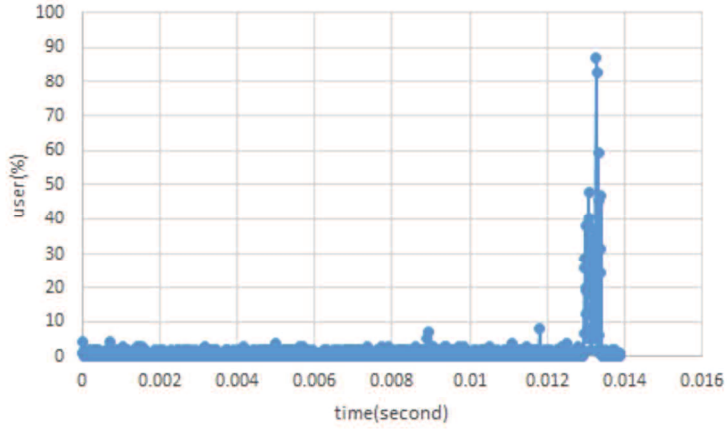

図 A.10　ネットワーク過負荷の場合

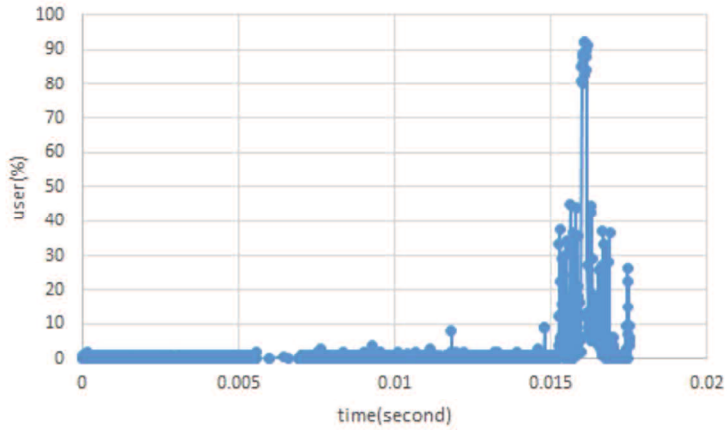

図 A.11　システム過負荷の場合

物理メモリの使用率（%）

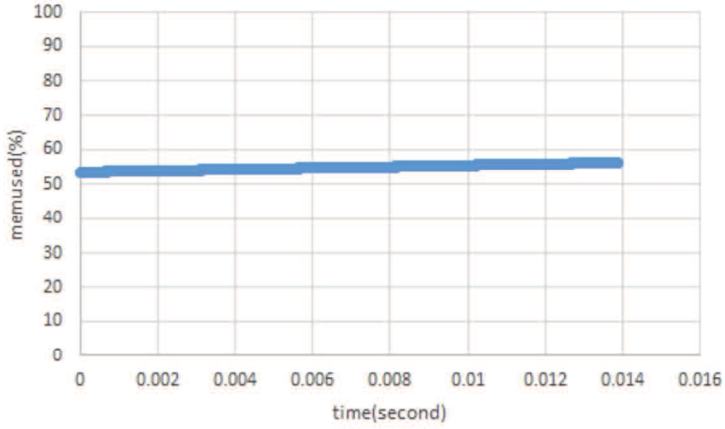

図 A.12　平時の場合

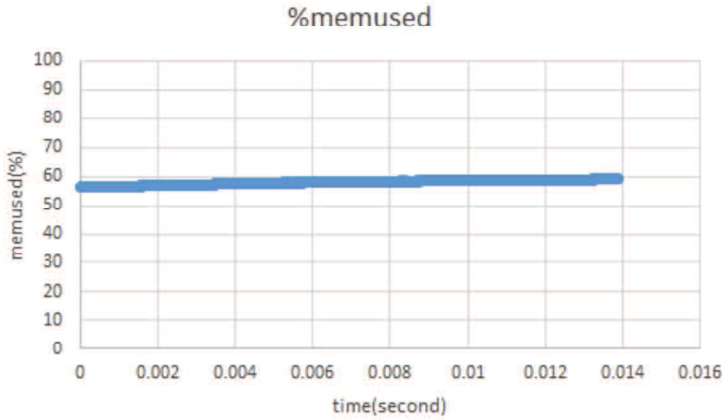

図 A.13　Web カメラ監視の場合

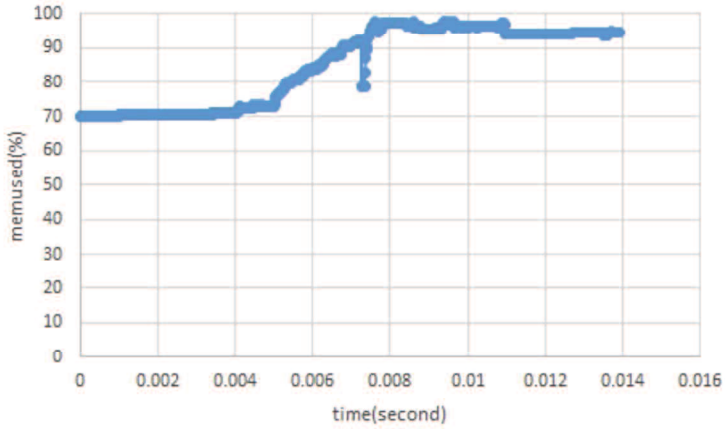

図 A.14　リモート接続による処理の場合

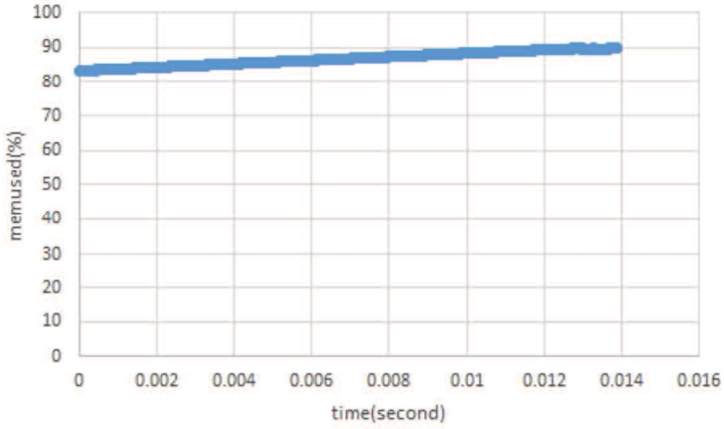

図 A.15　ネットワーク過負荷の場合

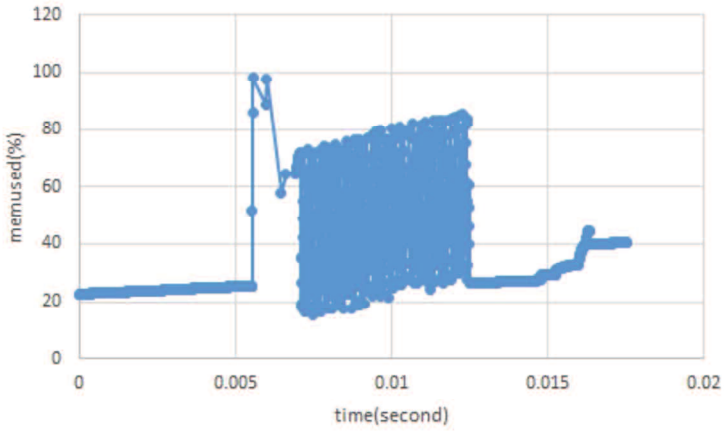

図 A.16　システム過負荷の場合

ネットワークインタフェースの毎秒受信バイト量（**kByte**）

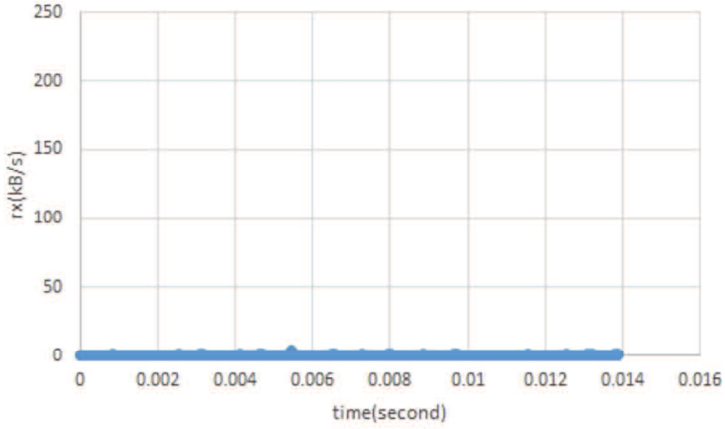

図 A.17　平時の場合

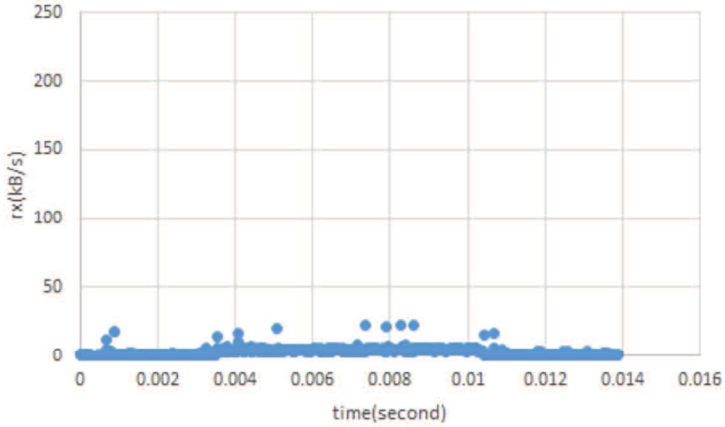

図 A.18　Web カメラ監視の場合

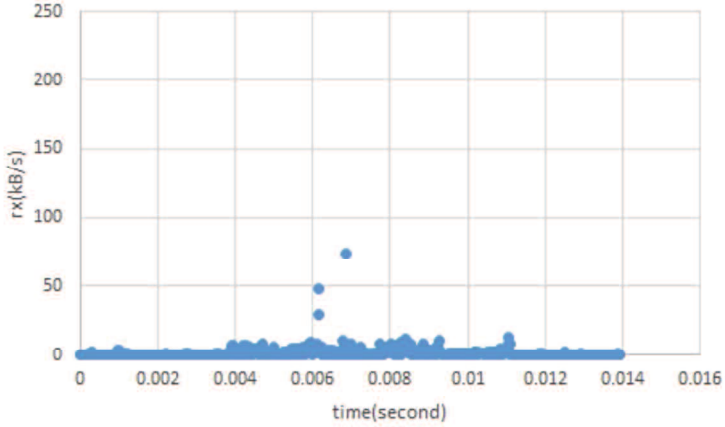

図 A.19　リモート接続による処理の場合

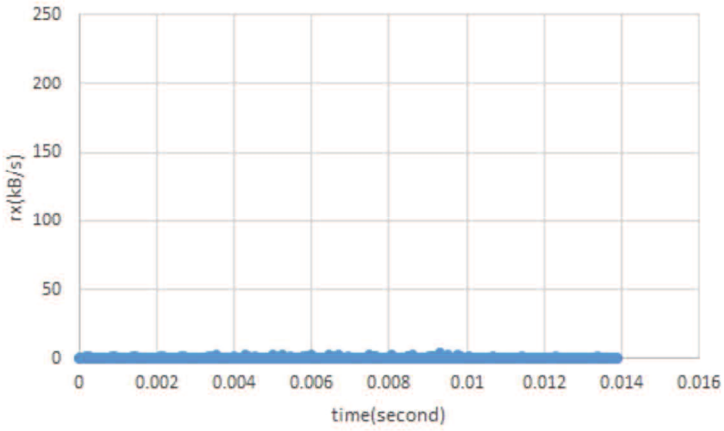

図 A.20　ネットワーク過負荷の場合

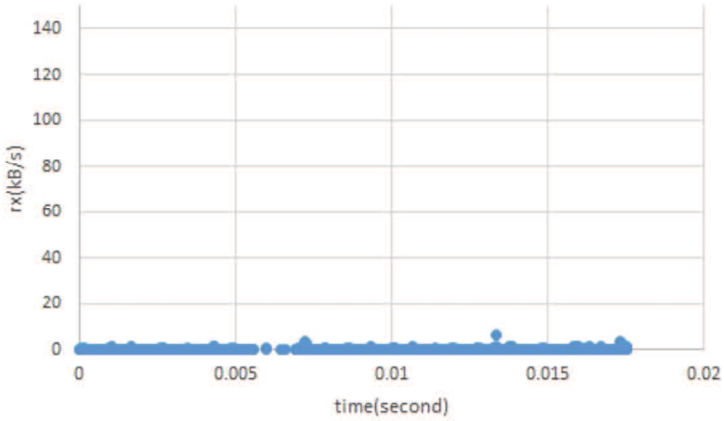

図 A.21　システム過負荷の場合

ネットワークインタフェースの毎秒送信バイト量（**kByte**）

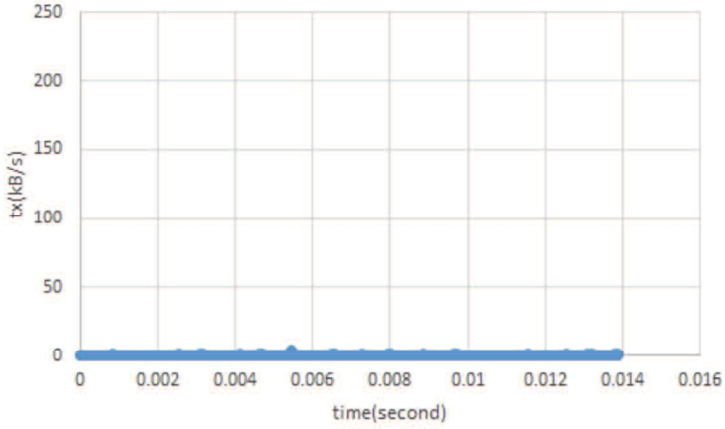

図 A.22　平時の場合

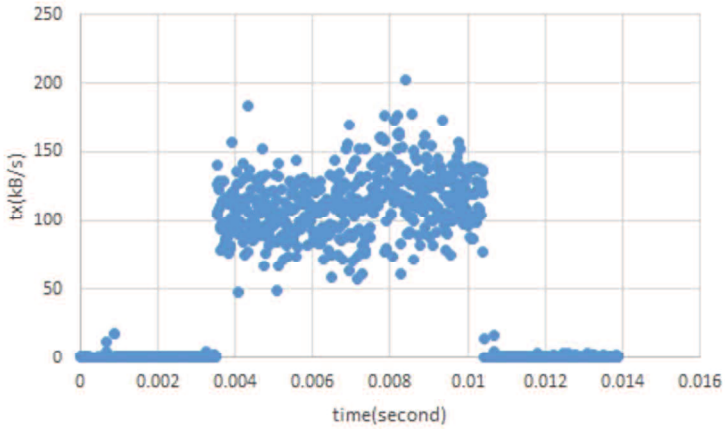

図 A.23　Web カメラ監視の場合

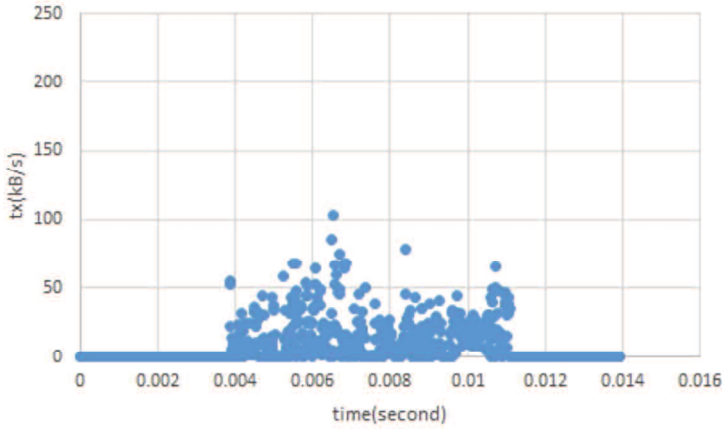

図 A.24　リモート接続による処理の場合

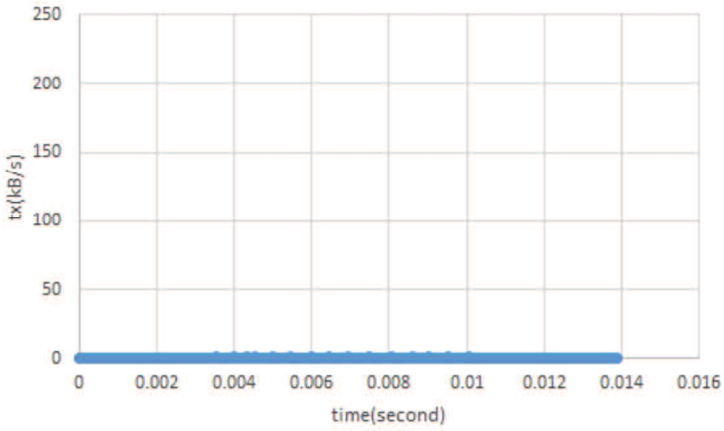

図 A.25　ネットワーク過負荷の場合

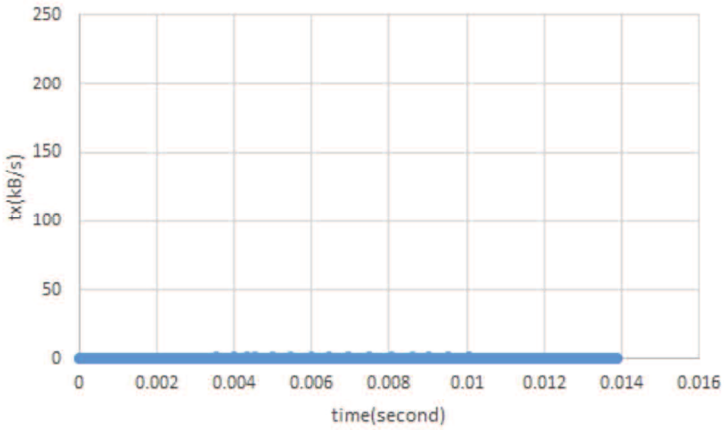

図 A.26　システム過負荷の場合

ディスクの 1 秒間の I/O リクエスト数（転送回数）

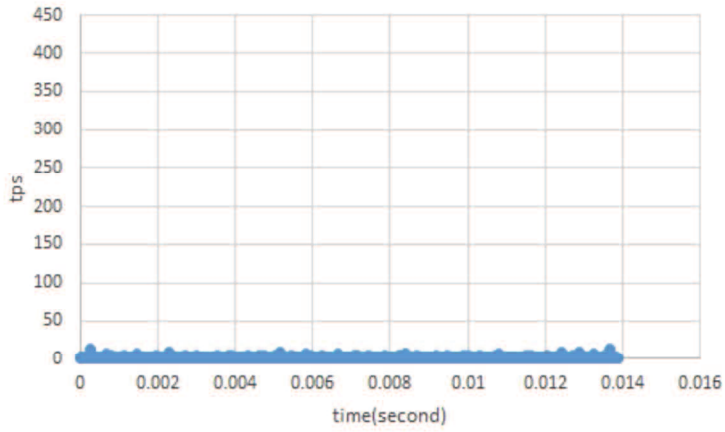

図 A.27　平時の場合

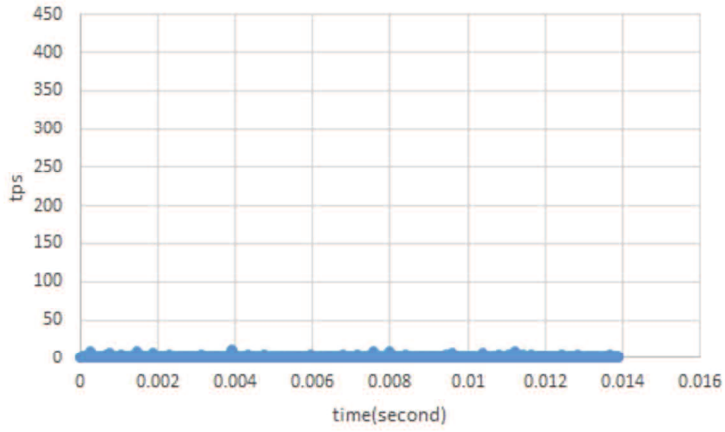

図 A.28　Web カメラ監視の場合

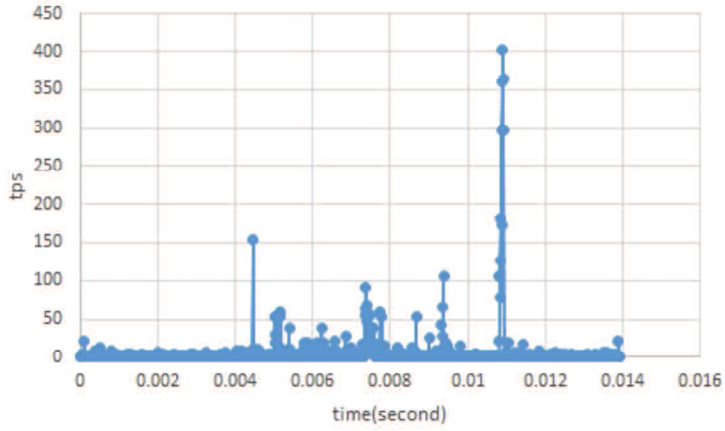

図 A.29　リモート接続による処理の場合

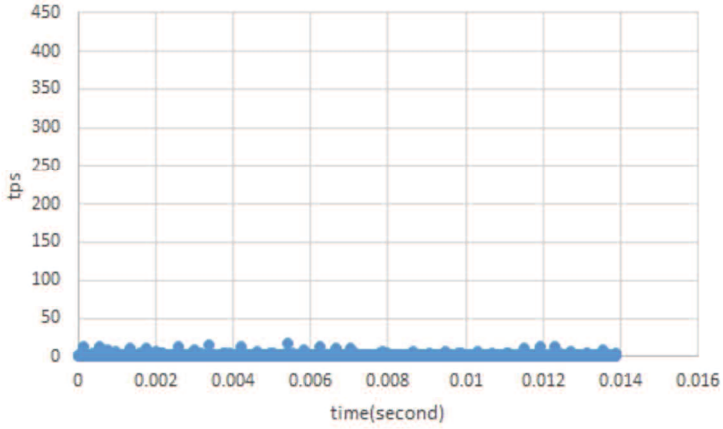

図 A.30　ネットワーク過負荷の場合

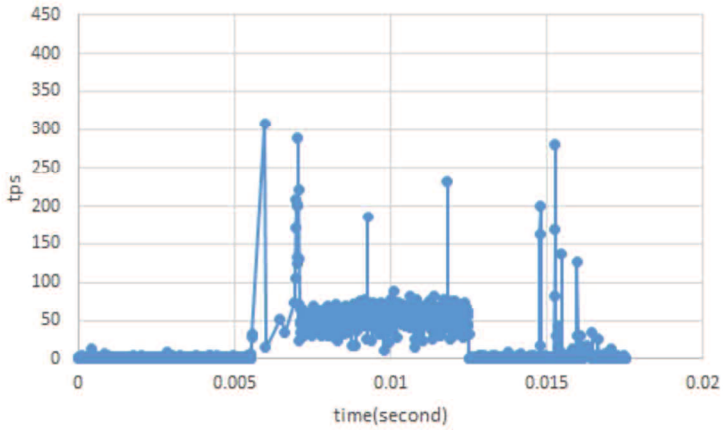

図 A.31　システム過負荷の場合

A.5　考察

実験結果のうち、システムが利用している CPU 使用率において異常が検知された、実験 (A-1-2) Web カメラ監視と、実験 (A-2-3) システム過負荷、の 2 つについて考察する。

まず、実験 (A-1-2) Web カメラ監視において、実験で得られた観測データから、ユーザ処理である Web カメラ監視を異常として誤検知した結果について考察する。観測データに単回帰分析を行い、結果を表 A.3 に示す。推測される回帰式を式 (A.1) に示す。

$$y = 28.507700 - 0.00000514x \tag{A.1}$$

式 (A.1) による回帰モデルにより、時間経過による傾向は殆どない。時刻 t、標本値 $x(t)$、平均 μ、標準偏差 σ、として検定統計量 $\tau(t)$ を式 (A.2) で定義し、2 を有意点として、標本値が $\mu \pm 2\sigma$ 外であれば外れ値とする。Web カメラ監視を実行した 10 分間の観測データを式 (A.2) で検定すると、$\tau(t)$ が 2 以上の外れ値が断続的に存在する。

$$\tau(t) = \frac{(x(t) - \mu)}{\sigma} \tag{A.2}$$

次に、観測データから外れ値と異常部位を判定するため、観測データを 1 スパン 30 データとする移動平均法で平滑し単回帰分析を行い、結果を表 A.4 に示す。推測される回帰式を式 (A.3) に示す。観測データと回帰式による線形近似を第 2 章 図 2.4 に示す。

$$y = 28.750011 - 0.000204x \tag{A.3}$$

移動平均法で平滑したデータから、Web カメラ監視を実行した 10 分間で外れ値を式 (A.2) で検定すると、この区間内の τ (t) は 1.2〜1.5 で明確に外れ値が存在するとはいえないが、やはり異常と誤検知する可能性が高い。

次に、実験 (A-2-3) システム過負荷において、実験から得られた観測データから、システム過負荷（疑似攻撃）を異常と正しく検知した結果について考察する。観測データに単回帰分析を行い、結果を表 A.5 に示す。推測される回帰式を式 (A.4) に

示す。

$$y = 13.718115 + 0.003101x \tag{A.4}$$

式 (A.4) による回帰モデルにより、時間経過による傾向は殆どない。観測データにおいて、システム過負荷を実行した 10 分間で、外れ値を式 (A.2) で検定すると、$\tau(t)$ が 2 以上の外れ値が断続的に存在する。次に、観測データから異常値と異常部位を判別するため、観測データを 1 スパン 30 データとする移動平均法で平滑し単回帰分析を行い、結果を表 A.5 に示す。推測される回帰式を式 (A.5) に示す。観測データと回帰式による線形近似を第 2 章　図 2.6 に示す。

$$y = 13.835700 + 0.003280x \tag{A.5}$$

移動平均法で平滑したデータから、システム過負荷を実行した 10 分間で外れ値を式 (A.2) で検定すると、この区間内の全 τ (t) が 2 以上の外れ値が連続的する。よって、本実験では外れ値と異常部位から異常を正しく検知できている。

以上の 2 つの予備実験の結果として、sar コマンドを用いたシステム監視では、システム過負荷を正しく異常として検知できる場合もあるが、ユーザ処理である Web カメラ監視を異常と誤検知する場合もある。その他、ネットワーク過負荷の実験で、ネットワーク過負荷による異常を見逃す場合もある。この理由として、システム監視が観測する各システムリソースの値の範囲が、正常時も含めて攻撃や不正アクセスの手口により異なるり、正常と異常を判別する閾値は専門家の判断で調整する必要があるためである。この閾値を、IoT デバイスにおいて定期的または緊急的に変更することは困難なことから、sar コマンドを用いたシステム監視による異常検知を、IoT デバイスのイベント検知として使用することは困難であると言える。以上の内容を、第 2 章の裏付けとする。

表 A.3　Web カメラ監視におけるシステム CPU 使用率の基本統計量

回帰統計				
単相関 R	重決定 R^2	補正 R^2	標準誤差	観測数
0.000186	3.47E-08	-0.00083	9.573403	1200

分散分析表					
	自由度	変　動	分　散	観測された分散比	有意 F
回帰	1	0.003815	0.003815	4.16E-05	0.994853
残差	1198	109796.7	91.65004		
合計	1199	109796.7			

標準誤差				
	係数	標準誤差	t	$P-$ 値
切片	28.5077	0.553066	51.54481	2.6E-306
x 値	-5.1E-06	0.000798	-0.00645	0.994853
	下限 95%	上限 95%	下限 99%	上限 99%
切片	27.42261	29.59279	27.08082	29.93458
x 値	-0.00157	0.00156	-0.00206	0.002053

表 A.4　Web カメラ監視におけるシステム CPU 使用率の移動平均法による基本統計量

回帰統計				
単相関 R	重決定 R^2	補正 R^2	標準誤差	観測数
0.0.013718	0.000188	-0.000667	5.030284	1171

分散分析表					
	自由度	変　動	分　散	観測された分散比	有意 F
回帰	1	5.567504	5.567504	0.220027	0.639107
残差	1169	29580.1	25.30376		
合計	1170	29585.66			

標準誤差				
	係数	標準誤差	t	$P-$ 値
切片	28.75001	0.294186	97.72726	0
x 値	-0.0002	0.000435	-0.46907	0.639107
	下限 95%	上限 95%	下限 99%	上限 99%
切片	28.17282	29.3272	27.991	29.50902
x 値	-0.00106	0.000649	-0.00133	0.000918

表 A.5　システム過負荷におけるシステム CPU 使用率の基本統計量

回帰統計				
単相関 R	重決定 R^2	補正 R^2	標準誤差	観測数
0.093366	0.008717	0.007998	13.180914	1380

分散分析表					
	自由度	変　動	分　散	観測された分散比	有意 F
回帰	1	2105.33	2105.334479	12.117974	0.000515
残差	1378	239408.90	173.736503		
合計	1379	241514.24			

標準誤差				
	係数	標準誤差	t	$P-$ 値
切片	13.718115	0.710022	19.320678	8.62E-74
x 値	0.003101	0.000891	3.481088	0.000515
	下限 95%	上限 95%	下限 99%	上限 99%
切片	12.32573	15.110956	11.886682	15.549548
x 値	0.001353	0.004848	0.000803	0.005400

表 A.6　システム過負荷におけるシステム CPU 使用率の移動平均法による基本統計量

回帰統計				
単相関 R	重決定 R^2	補正 R^2	標準誤差	観測数
0.102156	0.010436	0.009702	12.466096	1351

分散分析表					
	自由度	変　動	分　散	観測された分散比	有意 F
回帰	1	2210.82	2210.820573	14.226320	0.000169
残差	1349	209639.39	155.403550		
合計	1350	211850.21			

標準誤差				
	係数	標準誤差	t	$P-$ 値
切片	13.8356700	0.678694	20.385774	9.62E-81
x 値	0.003280	0.000870	3.771779	0.000169
	下限 95%	上限 95%	下限 99%	上限 99%
切片	12.50429	15.167110	12.085023	15.586376
x 値	0.001574	0.004986	0.001037	0.005523

A.6　まとめ

本付録では、汎用的なシステム監視が、IoT デバイスにおけるイベント検知として有用であるかを予備実験により評価した。実験結果から、システム監視により、システムリソースの値の変化や変動から異常を検出することは可能であるが、見逃しや誤検知も発生するため検知精度は低く、異常の原因がユーザ処理によるものかサイバー攻撃によるものかも判別できないことが確認された。よって、システム監視を IoT デバイスのイベント検知として使用することは困難であると言える。

謝辞

山口大学大学院創成科学研究科 福士 将 准教授には、本研究や論文投稿を進めるにあたり、常にあたたかいご指導、ご理解と激励のもと、多大なるお時間とご腐心を賜り、心より感謝の意を表します。山口大学大学院創成科学研究科 浜本 義彦 教授には、研究での気付きの姿勢や心得から、論文採録への難関克服のための具体的な手解きまで、五里霧中であった私に山口大学大学院後期課程入学前から燈明を照らしてくださり、心からお礼を申し上げます。山口大学大学院創成科学研究科 平野 靖 准教授ならびに藤田 悠介 准教授には、研究へ適切なご助言や貴重な参考資料を授けていただき、長きにわたりご尽力を頂き、深く感謝いたします。山口大学大学院創成科学研究科 河村 圭 准教授には、研究テーマを模索し、苦慮していたころより糸口を見つけ出すまで、多岐にわたりご助言にらびにご指導を頂き、深く感謝いたします。私が、研究のチャレンジする道を進むにあたり、橋渡しをしていただき折につけ激励していただいた西日本電信電話株式会社 鈴木 一義 氏に、心よりお礼申し上げます。最後に、入学より 7 年余の長きにわたり、これまで私をあたたかく応援してくれた両親、私を明るく励まし続けてくれた妻 秀子、娘、長男夫妻、そして孫たちに心から感謝します。

参考文献

[1] 鄭 立, “IoT 無線通信技術 LPWA と 5G の最新動向 ～SIGFOX, LoRaWAN, LTE-M, NB-IoT および 5G の現状と将来 ～,” 信学技報, vol. 118, no. 125, pp. 57–61, 2018.

[2] T. Gomes, F. Salgado, S. Pinto, J. Cabral, A. Tavares, “A 6LoWPAN Accelerator for Internet of Things Endpoint Devices,” IEEE Internet of Things Journal, vol. 5, no. 1, pp. 371–377, 2018.

[3] C. M. Ramya, M. Shanmugaraj, R. Prabakaran, “Study on ZigBee technology,” 2011 3rd International Conference on Electronics Computer Technology, Kanyakumari, India, pp. 297–301, 2011.

[4] S. Rani, S. H. Ahmed, R. Talwar, J. Malhotra, H. Song, “IoMT: A Reliable Cross Layer Protocol for Internet of Multimedia Things,” IEEE Internet of Things Journal, vol. 4, no. 3, pp. 832–839, 2017.

[5] S. T. Zargar, J. Joshi, D. Tipper, “A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks,” IEEE Communi-cations Surveys Tutorials, vol. 15, no. 4, pp. 2046–2069, 2013.

[6] B. Genge, I. Kiss, P. Haller, “A System Dynamics Approach for Assessing the Impact of Cyber Attacks on Critical Infrastructures,” International Journal of Critical Infrastructure Protection, vol. 10, pp. 3–17, 2015.

[7] H. Sinanovi, S. Mrdovic, “Analysis of Mirai Malicious Software,” 25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM), pp. 1–5, 2017.

[8] C. Kolias, G. Kambourakis, A. Stavrou, J. Voas, “DDoS in the IoT: Mirai and Other Botnets,” Computer, vol. 50, no. 7, pp. 80–84, 2017.

[9] R. A. Bridges, K. M.T. Huffer, C. L. Jones, M. D. Iannacone, J. R.

Goodall, “Cybersecurity Automated Information Extraction Techniques: Drawbacks of Current Methods, and Enhanced Extractors,” 6th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 437–442, 2017.

[10] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, Y. Zhou, “Understanding the Mirai Botnet,” 26th USENIX Security Symposium, pp. 1093–1110, 2017.

[11] Internet Initiative Japan Inc. 2016.

[12] S. Behal, K. Kumar, M. Sachdeva, “D-FACE: An anomaly based distributed approach for early detection of DDoS attacks and flash events,” Journal of Network and Computer Applications, vol. 111, pp. 49–63, 2018.

[13] K. Bhardwaj, J. C. Miranda, A. Gavrilovska, “Towards IoT-DDoS Prevention Using Edge Computing,” Georgia Institute of Technology, pp. 1–7, 2018.

[14] 総務省，“平成 30 年版 情報通信白書,” 2018.

[15] 総務省，“IoT セキュリティ総合対策,” 2017.

[16] S. Garg, A. Chaudhary, “A study of performance analysis of signaling protocols in MPLS,” 3rd International Conference on Computational Intelligence & Communication Technology (CICT), pp. 1–5, 2017.

[17] A. Zammali, A. de Bonneval, Y. Crouzet, P. Izzo, J. Massimi, “Communication integrity for future helicopters Flight Control Systems,” IEEE/AIAA 34th Digital Avionics Systems Conference (DASC), pp. 1–20, 2015.

[18] A. L. Filho, D. Vettorazzi, A. Cruz, C. Lima, “ATM: a New Heuristic Algorithm Based on Genetic Algorithm and Betting Theory,” IEEE Latin America Transactions , vol. 3, no. 3, pp. 510–516, 2017.

[19] 日本規格協会，“JIS Q 27001: 2014 情報技術-セキュリティ技術-情報セキュリティマネジメントシステム-要求事項”.

[20] 日本情報経済社会推進協会 (JIPDEC)，“ISMS ユーザーズガイド-JIS Q 27001: 2014 (ISO/IEC 27001: 2013) 対応”.

[21] 日本情報経済社会推進協会 (JIPDEC)，“JIP-CSAC100-0.8a サイバーセキュリティマネジメントシステム (Cyber Security Management System) CSMS 認証機関認定基準及び指針”.

[22] IEC TC 65, “IEC 62443-2-1: 2010 Industrial communication networks - Network and system security - Part 2-1: Establishing an industrial automation and control system security program,” 2010.

[23] 日本規格協会，“JIS Q 27000: 2014 情報技術-セキュリティ技術-情報セキュリティマネジメントシステム-用語,” 2014.

[24] 渡邉晴方, 原田要之助，“情報資産の分類に基づくラベル付けと営業秘密に関する考察,” 情報処理学会 研究報告電子化知的財産・社会基盤（EIP），vol. 2013-EIP-62 (10)，pp. 1–6，2013.

[25] British Standards Institution (BSI), United Kingdom of Great Britain and Northern Ireland, London, “BS 7799 Information Security Management Systems - Specification with guidance for use.,” 1999.

[26] ISO/IEC JTC 1/SC 27, Switzerland, Geneva, “ISO/IEC 17799: 2000 Information Technology - Code of practice for information security management,” 2000.

[27] ISO/IEC JTC 1/SC 27, Switzerland, Geneva, “ISO/IEC 27001: 2005 Information technology – Security techniques – Information security management systems – Requirements,” 2005.

[28] ISO/IEC JTC 1/SC 27, Switzerland, Geneva, “ISO/IEC 27001: 2013 Information technology – Security techniques – Information security management systems – Requirements,” 2013.

[29] V. Moreno, J. Ramos, P. M. S. del Rio, J. L. Garcia-Dorado, F J. Gomez-Arriba, “Commodity Packet Capture Engines: Tutorial, Cookbook and Applicability,” IEEE Communications Surveys & Tutorials, vol. 17, no. 3, pp. 1364–1390, 2015.

[30] S. Tennina, O. Gaddour, A. Koubaa, F. Royo, M. Alves, M. Abid, “Z-

Monitor: A protocol analyzer for IEEE 802.15.4-based low-power wireless networks," Computer Networks, vol. 95, pp. 77–96, 2016.

[31] W. Konikiewicz, M. Markowski, "Analysis of Performance and Efficiency of Hardware and Software Firewalls," Journal of Applied Computer Science Methods, vol. 9, no. 1, pp. 49–63, 2018.

[32] S. Kubler, K. Framling, A. Buda, "A standardized approach to deal with firewall and mobility policies in the IoT," Pervasive and Mobile Computing, vol. 20, pp. 100–114, 2014.

[33] O. Al-Jarrah, A. Arafat, "Network Intrusion Detection System using attack behavior classification," 5th International Conference on Information and Communication Systems (ICICS), pp. 1–6, 2014.

[34] B. B. Zarpelao, R. S. Miani, C. T. Kawakani, S. C. de Alvarenga, "A survey of intrusion detection in Internet of Things," Journal of Network and Computer Applications, vol. 84, pp. 25–37, 2017.

[35] T. Mokoena, T. Zuva, "Malware Analysis and Detection in Enterprise Systems," IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC), pp. 1304–1310, 2017.

[36] S. W. Soliman, M. A. Sobh, A. M. Bahaa-Eldin, "Taxonomy of malware analysis in the IoT," 12th International Conference on Computer Engineering and Systems (ICCES), pp. 519–529, 2017.

[37] M. Aldwairi, A. M. Abu-Dalo, M. Jarrah, "Pattern matching of signature-based IDS using Myers algorithm under MapReduce framework," EURASIP Journal on Information Security, pp. 1–11, 2017.

[38] A. Meena, U. Sinthuja, "Data Mining in Network Security using Intrusion Detection System," Networking and Communication Engineering, vol. 10, no. 1, 2018.

[39] X. Clotet, J. Moyano, G. Leon, "A real-time anomaly-based IDS for cyber-attack detection at the industrial process level of Critical Infrastructures,"

International Journal of Critical Infrastructure Protection, 2018.

[40] P. Alaei, F. Noorbehbahani, “Incremental anomaly-based intrusion detection system using limited labeled data,” 3th International Conference on Web Research (ICWR), 2017.

[41] S. Kumar, “A Pattern Matching Model for Misuse IntrusionDetection,” DEPARTMENT OF COMPUTER SCIENCE TECHNICAL REPORTS, vol. 94-071, 1994.

[42] D. E. Denning, “An Intrusion-Detection Model,” IEEE Transactions on Software Engineering, vol. SE-13, no. 2, pp. 222–232, 1987.

[43] D. Yeung, Y. Ding, “Host-based intrusion detection using dynamic and static behavioral models,” Pattern Recognition, vol. 36, no. 1, pp. 229–243, 2003.

[44] A. K. Ghosh, A. Schwartzbard, M. Schatz, “Learning Program Behavior Profilesfor Intrusion Detection,” Proceedings of the Workshop on Intrusion Detectionand Network Monitoring, 1999.

[45] IoT 推進コンソーシアム 総務省 経済産業省, “IoT セキュリティガイドライン ver. 1.0,” 2016. http://www.soumu.go.jp/main_content/000428393.pdf

[46] P. Bagade, A. Chandra, A. B. Dhende, “Designing performance monitoring tool for NoSQL Cassandra distributed database,” nternational Conference on Education and e-Learning Innovations, 2012.

[47] R. Andresen, “Monitoring Linux with native tools,” 2004.

[48] G. Vigna, R. A. Kemmerer, “NetSTAT: a network-based intrusion detection approach,” Proceedings 14th Annual Computer Security Applications Conference, 1998.

[49] S. Raza, L. Wallgren, T. Voigt, “SVELTE: Real-time intrusion detection in the Internet of Things,” Ad Hoc Networks, vol. 11, no. 8, pp. 2661–2674, 2013.

[50] E. Hodo, X. Bellekens, A. Hamilton, “Threat analysis of IoT networks using artificial neural network intrusion detection system,” International

Symposium on Networks, Computers and Communications (ISNCC), 2016.

[51] A. Abduvaliyev, A. S. K. Pathan, J. Zhou, “On the Vital Areas of Intrusion Detection Systems in Wireless Sensor Networks,” IEEE Communications Surveys & Tutorials, vol. 15, no. 3, pp. 1223–1237, 2013.

[52] D. Yeung, Y. Ding, “Host-based intrusion detection using dynamic and static behavioral models,” Pattern Recognition, vol. 36, no. 1, pp. 229–243, 2003.

[53] H. Zhengbing, L. Zhitang, W. Junqi, “A novel Network Intrusion Detection System (NIDS) based on signatures search of data mining,” Proceedings of the 1st international conference on Forensic applications and techniques in telecommunications, information, and multimedia and workshop, 2008.

[54] A. P. Singh, M. D. Singh, “Analysisof Host-BasedandNetwork-Based Intrusion Detection System,” I. J. Computer Network and Information Security, pp. 41–47, 2014.

[55] 星野裕，“国際標準と国際市場性,” 標準化研究，vol. 4，no. 1，pp. 71–83，2006.

[56] ISO/TMB/WG, “ISO 31000: 2018 Risk management - Guidelines,” 2018.

[57] 日本規格協会，“JIS Q 31000: 2010 リスクマネジメント－原則及び指針,” 2010.

[58] 井上邦夫，“リスクマネジメントと危機管理: コミュニケーションの視点から,” 経営論集，vol. 86，pp. 101–111，2015.

[59] ISO/IEC JTC 1/SC 27, “ISO/IEC 27001: 2013 Information technology – Security techniques – Information security management systems – Requirements,” 2013.

[60] ISO/IEC JTC 1/SC 27, “ISO/IEC 27002: 2013 Information technology – Security techniques – Code of practice for information security controls,” 2013.

[61] ISO/IEC JTC 1/SC 27, “ISO/IEC 27033-1: 2015 Information technol-

ogy – Security techniques – Network security – Part 1: Overview and concepts," 2015.

[62] ISO/IEC JTC 1/SC 27, "ISO/IEC 27032: 2012 Information technology – Security techniques – Guidelines for cybersecurity," 2012.

[63] ISO/IEC JTC 1/SC 27, "ISO/IEC 15408-1: 2009 Information technology – Security techniques – Evaluation criteria for IT security – Part 1: Introduction and general model," 2009.

[64] IEC TC 65 - Industrial-process measurement, control and automation, "IEC 62443-2-1: 2010 Industrial communication networks - Network and system security - Part 2-1: Establishing an industrial automation and control system security program," 2010.

[65] ISO/IEC JTC 1/SC 40 IT Service Management and IT Governance, "ISO/IEC 20000-1: 2011," 2011.

[66] The National Institute of Standards and Technology (NIST), "Guide to Intrusion Detection and Prevention Systems (IDPS)," 2007.

[67] The National Institute of Standards and Technology (NIST), "SP 800-94 Guide to Intrusion Detection and Prevention Systems (IDPS)," 2007.

[68] R. Ross, R. Graubart, D. Bodeau, R. McQuaid, "SP 800-160 Vol. 2 Systems Security Engineering: Cyber Resiliency Considerations for the Engineering of Trustworthy Secure Systems," 2018.

[69] Interagency International Cybersecurity Standardization Working Group (IICS WG), "NISTIR 8200 (DRAFT) Interagency Report on Status of International Cybersecurity Standardization for the Internet of Things (IoT)," 2018.

[70] 独立行政法人情報処理推進機構 (IPA) 産業サイバーセキュリティセンター, "制御システムのセキュリティリスク分析ガイド 第 2 版セキュリティ対策におけるリスクアセスメントの実施と活用," 2017.

[71] 総務省 総合通信基盤局 電気通信事業部 電気通信技術システム課, "情報通信ネットワーク安全・信頼性基準用 別表第 4 危機管理計画策定のための指針," 2008.

[72] H. Zimmermann, "OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection," IEEE Transactions on Communications, vol. 28, no. 4, pp. 425–432, 1980.

[73] F. Guo, T. Chiueh, "Sequence Number-Based MAC Address Spoof Detection," Recent Advances in Intrusion Detection, pp. 309–329, 2005.

[74] M. Mohsin, R. Prakash, "IP address assignment in a mobile ad hoc network," MILCOM 2002. Proceedings, 2002.

[75] J. Gadge, A. A. Patil, "Port scan detection," 16th IEEE International Conference on Networks, 2008.

[76] D. Esposito, M. Rennhard, L. Ruf, A. Wagner, "Exploiting the potential of web application vulnerability scanning," The Thirteenth International Conference on Internet Monitoring and Protection, pp. 22–29, 2018.

[77] D. Dykstra, J. Blomer, B. Blumenfeld, A. De Salvo, A. Dewhurst, V. Verguilov, "Web Proxy Auto Discovery for the WLCG," IOP Conf. Series: Journal of Physics: Conf., 2018.

[78] E. S. Finn, X. Shen, D. Scheinost, M. D. Rosenberg, J. Huang, M. M. Chun, X. Papademetris, R. T. Constable, "Functional connectome fingerprinting: identifying individuals using patterns of brain connectivity," Nature Neuroscience, vol. 18, pp. 1664–1671, 2015.

[79] F. S. Duarte, F. Sikansi, F. M. Fatore, S. G. Fadel and F. V. Paulovich, "Nmap: A Novel Neighborhood Preservation Space-filling Algorithm," IEEE Transactions on Visualization & Computer Graphics, vol. 20, no. 12, pp. 2063–2071, 2014.

[80] M. Woolmanab, A. Zarrine-Afsar, "Platforms for rapid cancer characterization by ambient mass spectrometry: advancements, challenges and opportunities for improvement towards intrasurgical use," Analyst, vol. 12, 2018.

[81] S. Jajodia, S. Noel, B. O' Berry, "Topological Analysis of Network Attack Vulnerability," Massive Computing, vol. 5, pp. 247–266, 2006.

[82] M. Moore, "Penetration Testing and Metasploit," Computer Science De-

partmentJackson State University, 2017.

[83] P. A. Karger, R. R. Schell, "Multics security evaluation: vulnerability analysis," 18th Annual Computer Security Applications Conference 2002 Proceedings, 2002.

[84] Y. Makino, V. Klyuev, "Evaluation of web vulnerability scanners," IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 2015.

[85] L. Dukes, X. Yuan, F. Akowuah, "A case study on web application security testing with tools and manual testing," Proceedings of IEEE Southeastcon, 2013.

[86] G. Trifonov, "Reducing the number of security vulnerabilities in web applications by improving software quality," 5th International Symposium on Applied Computational Intelligence and Informatics, 2009.

[87] C. Joshi, U. K. Singh, "Performance Evaluation of Web Application Security Scanners for More Effective Defense," International Journal of Scientific and Research Publications, vol. 6, no. 6, pp. 660–667, 2016.

[88] S. Kals, E. Kirda, C. Kruegel, N. Jovanovic, "SecuBat: a web vulnerability scanner," Proceedings of the 15th international conference on World Wide Web, pp. 247–256, 2006.

[89] F. Chaudhari, S. Patel, "A Survey: Trojan horse Detection Techniques in Network," International Journal of Computer & Mathematical Sciences (IJCMS), vol. 6, no. 9, pp. 117–119, 2017.

[90] S. Aggarwal, S. Houshmand, M. Weir, "New Technologies in Password Cracking Techniques," Cyber Security: Power and Technology, pp. 179–198, 2018.

[91] S. Bhowal, S. R. Dutta, S. Mitra, "An efficient reduced set brute force attack technique for a particular steganographic tool using vername algorithm," Fourth International Conference on Image Information Processing (ICIIP), 2017.

[92] Y. Kobayashi, N. Yanai, K. Yoneyama, T.i Nishide, G. Hanaoka, K. Kim, E. Okamoto, “Provably Secure Gateway Threshold Password-Based Authenticated Key Exchange Secure against Undetectable On-Line Dictionary Attack,” IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences, vol. E100-A, no. 12, pp. 2991–300, 2017.

[93] L. Zhang, C. Tan, F. Yu, “An Improved Rainbow Table Attack for Long Passwords,” Procedia Computer Science, vol. 107, pp. 47–52, 2017.

[94] Y. Ji, X. Zhang, T. Wang, “Backdoor attacks against learning systems,” IEEE Conference on Communications and Network Security (CNS), 2017.

[95] M. Backes, S. Bugiel, O. Schranz, P. von Styp-Rekowsky, S. Weisgerber, “ARTist: The Android Runtime Instrumentation and Security Toolkit,” IEEE European Symposium on Security and Privacy (EuroS&P), 2017.

[96] Z. Cui, F. Xue, X. Cai, Y. Cao, G. Wang, “Detection of Malicious Code Variants Based on Deep Learning,” IEEE Transactions on Industrial Informatics, vol. 14, no. 7, pp. 3187–3196, 2018.

[97] S. N. Bukhari, M. A. Dar, U. Iqbal, “Reducing attack surface corresponding to Type 1 cross-site scripting attacks using secure development life cycle practices,” Fourth International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB), 2018.

[98] G. Wen, Y. Hu, C. Jiang, N. Cao, “A image texture and BP neural network basec malicious files detection technique for cloud storage systems,” IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2017.

[99] B. Rajesh, Y. R. J. Reddy, B. D. K. Reddy, “A Survey Paper on Malicious Computer Worms,” International Journal of Advanced Research inComputer Science & Technology, vol. 3, no. 2, pp. 161–167, 2015.

[100] 浦辻和也, 松重雄大, 甲斐博, “マルウェア可視化システムの実装について,”

情報科学技術フォーラム，vol. 4，pp. 159–162，2015.

[101] C. C. Zou, W. Gong, D. Towsley , “Code red worm propagation modeling and analysis,” Proceedings of the 9th ACM conference on Computer and communications security, pp. 138–147, 2002.

[102] M. Lad, X. Zhao, B. Zhang, D. Massey, L. Zhang, “Analysis of BGP Update Surge during Slammer Worm Attack,” International Workshop on Distributed Computing, pp. 66–79, 2003.

[103] W. G. J. Halfond, A. Orso, “AMNESIA: analysis and monitoring for NEutralizing SQL-injection attacks,” Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering, pp. 174–183, 2005.

[104] P. Bisht, V. N. Venkatakrishnan, “XSS-GUARD: Precise Dynamic Prevention of Cross-Site Scripting Attacks,” International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, pp. 23–43, 2008.

[105] C. J. Chae, Y. J. Shin, K. Choi, K. B. Kim, K. N. Choi, “A privacy data leakage prevention method in P2P networks,” Peer-to-Peer Networking and Applications, vol. 9, no. 3, pp. 508–519, 2016.

[106] S. Kobayashi, M. Goudge, T. Makie, E. Hanada, “A Filter That Prevents the Spread of Mail-Attachment-Type Trojan Horse Computer Worms,” Journal of Medical Systems, vol. 26, no. 3, pp. 221–222, 2002.

[107] B. Parmar, “Protecting against spear-phishing,” Computer Fraud & Security, vol. 2012, no. 1, pp. 8–11, 2012.

[108] S. A. Crosby, D. S. Wallach, “Denial of Service via Algorithmic Complexity Attacks,” USENIX Security Symposium, 2003.

[109] J. Koo, S. Ahn, Y. Lim, Y. Mun, “Evaluation of Network Blocking Algorithm Based on ARP Spoofing and Its Application,” International Conference on Computational Science and Its Applications, pp. 848–855, 2005.

[110] H. R. Zeidanloo, A. A. Manaf, “Botnet command and control mechanisms,” Second International Conference, 2009.

[111] B. Harris, R. Hunt, "TCP/IP security threats and attack method," Computer Communications, vol. 22, no. 10, pp. 885–897, 1999.

[112] C. C. Chiu, C. Y. Tsai, "A Web services-based collaborative scheme for credit card fraud detection," IEEE International Conference on e-Technology, e-Commerce and e-Service, 2004.

[113] W. M. Eddy, "SYN Flood Attack," Encyclopedia of Cryptography and Security, 2011.

[114] R. K. C. Chang, "Defending against flooding-based distributed denial-of-service attacks: a tutorial," Browse Journals & Magazines, vol. 40, no. 10, pp. 42–51, 2002.

[115] C. W. Ten, G. Manimaran, C. C. Liu, "Cybersecurity for Critical Infrastructures: Attack and Defense Modeling," IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 40, no. 4, pp. 853–865, 2010.

[116] M. V. Mahoney, P. K. Chan, "Learning nonstationary models of normal network traffic for detecting novel attacks," Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 376–385, 2002.

[117] R. S. R. Gade, H. Vellalacheruvu, "Performance of Windows XP, Windows Vista and Apple's Leopard Computers under a Denial of Service Attack," Fourth International Conference on Digital Society, 2010.

[118] A. Singh, D. Juneja, "Agent based preventive measure for UDP flood attack in DDoS attacks," International Journal of Engineering Science, 2010.

[119] N. Daswani, M. Stoppelman, "The anatomy of Clickbot.A," HotBots'07 Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, p. 11, 2007.

[120] T. Hirakawa, K. Ogura, B. B. Bista, "A Defense Method against Distributed Slow HTTP DoS Attack," 19th International Conference on Network-Based Information Systems (NBiS), 2016.

[121] C. Jackson, A. Barth, A. Bortz, W. Shao, “Protecting browsers from DNS rebinding attacks,” ACM Transactions on the Web (TWEB), vol. 3, no. 1, 2009.

[122] T. Holz, H. Bos, “Protecting against DNS Reflection Attacks with Bloom Filters,” Detection of Intrusions and Malware, and Vulnerability Assessment, pp. 1–16, 2011.

[123] B. Dowling, D. Stebila, G. Zaverucha, “ANTP: Authenticated NTP Implementation Specification Microsoft Research,” 2015.

[124] X. Geng, A. B. Whinston, “Defeating distributed denial of service attacks,” IEEE IT Professional, vol. 2, no. 4, pp. 36–42, 2000.

[125] C. Shannon, D. Moore, “Characteristics of fragmented IP traffic on internet links,” Proceedings of the 1st ACM SIGCOMM Workshop on Internet measurement, pp. 83–97, 2001.

[126] C. Kaufman, R. Perlman, B. Sommerfeld, “DoS protection for UDP-based protocols,” Proceedings of the 10th ACM conference on Computer and communications security, pp. 2–7, 2003.

[127] F. Audet, C. Jennings, “Protection Against Packet Fragmentation Attacks at 6LoWPAN Adaptation Layer,” International Conference on Convergence and Hybrid Information Technology, 2008.

[128] 基礎・境界ソサイエティ, 電子情報通信学会，“パターン認識問題の数理,” Fundamentals Review，vol. 5，no. 4，pp. 302–311，2012.

[129] W. Lee, S. J. Stolfo, K. W. Mok, “A data mining framework for building intrusion detection models,” Proceedings of the 1999 IEEE Symposium on Security and Privacy, 1999.

[130] 藤田直行，“侵入検知に関する誤検知低減の研究動向,” 電子情報通信学会論文誌 B，vol. J89-B，no. 4，pp. 402–411，2006.

[131] 白鳥則郎, キーゥ哀譽鵐マンスフィールド, 小出和秀, 長尾真宏，“ネットワークモビリティをサポートする新世代ユビキタスネットワーク監視フレームワーク　ー総務省 SCOPE プロジェクトー,” 情処学研報　マルチメディア通信と分散処理，vol. 2007，no. 58，pp. 57–60，2007.

[132] 日本規格協会，“JIS Z 8402-5：2002（ISO 5725-5：1998）測定方法及び測定結果の精確さ（真度及び精度）,” 2002.

[133] TC/SC: ISO/IEC JTC 1/SC 27, “ISO/IEC 27039: 2015 Information technology – Security techniques – Selection, deployment and operations of intrusion detection and prevention systems (IDPS),” 2015.

[134] 日本規格協会，“JIS Q 31000：2010（ISO 31000：2009）リスクマネジメント- 原則及び指針,” 2010.

[135] S. Papadimitriou, H. Kitagawa, P.,B. Gibbons, C. Faloutsos, “LOCI: fast outlier detection using the local correlation integral,” Proceedings 19th International Conference on Data Engineering, 2003.

[136] S. Agrawal, J. Agrawal, “Survey on Anomaly Detection using Data Mining echniques Procedia Computer Science,” Procedia Computer Science, vol. 60, pp. 708–713, 2015.

[137] M. M. Breunig, H. Kriege, R. T. Ng, J. Sander, “LOF: Identifying Density-Based Local Outliers,” SIGMOD ’00 Proceedings of the 2000 ACM SIGMOD international conference on Management of data, pp. 93–104, 2000.

[138] F. T. Liu, K. M. Ting, Z. Zho, “Isolation-Based Anomaly Detection,” ACM Transactions on Knowledge Discovery from Data (TKDD), vol. 6, no. 3, pp. 1–44, 2012.

[139] H. P. Kriegel, M. Schubert, A. Zimek, “Angle-Based Outlier Detection in High-Dimensional Data,” in Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 444–452, 2008.

[140] 陳山鵬，“械設備の異常検知と状態判定基準について（異常検知と変化点検出）,” REAJ 誌，vol. 37，no. 3，2015.

[141] 井手剛，入門 機械学習による異常検知，コロナ社，2015.

[142] A. Daneshpazhouh, A. Sami, “Entropy-based outlier detection using semi-supervised approach with few positive examples,” Pattern Recognition Letters, vol. 49, pp. 77–84, 2014.

[143] 澤村隆志, ベッド B．ビスタ, 高田豊雄，“PC 内のファイル改ざんを行うマ

ルウェアの検知手法," 研究報告コンピュータセキュリティ, Vol.2012-CSEC-56，vol. 11，pp. 1–7，2012.

[144] T. L. Lai, "Sequential Changepoint Detection in Quality Control and Dynamical Systems," Journal of the Royal Statistical Society, vol. 57, no. 4, pp. 613–658, 1995.

[145] V. Chandola, A. Banerjee, V. Kumar, "Anomaly detection: A survey," ACM Computing Surveys, vol. 41, no. 3, 2009.

[146] R. D. Cairns, "Economic Accounting in the Simple Hotelling Model," Resource and Energy Economics, vol. 51, pp. 18–27, 2018.

[147] C. Atewamba, B. Nkuiya, "Testing the Assumptions and Predictions of the Hotelling Model," Environmental and Resource Economics, vol. 66, no. 1, pp. 169–203, 2017.

[148] J. M. Mendel, "Unit Root Tests in ARMA Models with Data-Dependent Methods for the Selection of the Truncation Lag," Proceedings of the IEEE, vol. 79, no. 3, pp. 278–305, 1991.

[149] S. Huang, K. Shih, "Short-term load forecasting via ARMA model identification including non-Gaussian process considerations," IEEE Transactions on Power Systems, vol. 18, no. 2, pp. 673–679, 2003.

[150] H. Z. Moayedi, M. A. Masnadi-Shirazi, "Arima model for network traffic prediction and anomaly detection," International Symposium on Information Technology, 2008.

[151] CRN Magazine, "DDoS Attacks That Made Enterprises Rethink IoT Security," http://www.crn.com/slide-shows/internet-of-things/300084663/8-ddos-attacks-that-made-enterprises-rethink-iot-security.htm, 2017.

[152] A. M. Gamundani, "An impact review on internet of things attacks," IEEE Emerging Trends in Networks and Computer Communications, pp. 114–118, 2015.

[153] 総務省警察庁，"不正プログラムに感染した IoT 機器が発信元と考えられるアクセスの増加等について," 2017.

[154] 牧田大佑, 西添友美, 吉岡克成, 松本勉, 井上大介, 中尾康二，“早期インシデント対応を目的としたDRDoS攻撃アラートシステム,” 情処学論，vol. 57，no. 9，pp. 1974–1985，2016.

[155] 竹尾大輔, 伊藤将志, 鈴木秀和, 岡崎直宣, 渡邊晃，“コネクションベース方式による踏み台攻撃検出手法の提案,” 情処学論，vol. 48，no. 2，pp. 644–655，2007.

[156] 菅原啓介, 千石靖, 八島一司, 地下雅志, 西川弘幸, 岡本栄司，“未知ウイルス検出技術に関する一考察,” The Institute of Electronics, Information and Communication Engineers，2004.

[157] R. Vaarandi, B. Blumbergs, M. Kont, “An unsupervised framework for detecting anomalous messages from syslog log files,” IEEE/IFIP Network Operations and Management Symposium, 2018.

[158] Tenable Network Security, “Nessus Sample Reports”. `https://www.tenable.com/products/nessus/sample-reports`

[159] The Apache Software Foundation, “ab”. `http://httpd.apache.org/docs/2.4/programs/ab.html`

[160] D. A. Grimes, K. F. Schulz, “Refining clinical diagnosis with likelihood ratios,” Lancet, vol. 365, pp. 1500–1505, 2005.

[161] M. Davidson, “The interpretation of diagnostic tests,” Australian J. Physiotherapy, vol. 48, no. 3, pp. 227–232, 2002.

[162] A. S. Glas, J. G. Lijmer, M. H.Prins, G. J. Bonsel, P. M. M. Bossuyt, “The diagnostic odds ratio: a single indicator of test performance,” J. Clinical Epidemiology, vol. 56, pp. 1129–1135, 2003.

[163] 竹本秀樹, 双紙正和, 宮地充子，“DoS攻撃に対する偽造耐性をもつ改良パケットマーキング法の提案と評価,” 情報処理学会研究報告マルチメディア通信と分散処理 (DPS)，vol. 16，pp. 103–110，2007.

[164] 栗木進二, 岩瀬晃盛，“打切標本の標本分散および標本変動係数の評価,” 応用統計学会，vol. 6，no. 1，pp. 35–39，1977.

[165] 塩川孝雄，“しきい値を設定しない相対的なキャパシティ管理の有用性,” 情報処理学会研究報告システム評価 (EVA)，vol. 62，pp. 41–46，2004.