情報処理学会論文誌 Vol.60 No.3 791–802 (Mar. 2019)

車載LANへ侵入するマルウェアの証拠保全を行うカーネル上のフォレンジック機構

大平 修慈[1,a)] 井上 博之[2] 新井 イスマイル[3] 藤川 和利[3]

受付日 2018年6月25日, 採録日 2018年12月4日

概要：インターネットとつながる自動車において，カーナビ等の車載インフォテインメントシステム（IVI）へのマルウェアの侵入等による車載 LAN へのサイバー攻撃が問題となっている．また，マルウェアのような不正なプログラムによる事故が発生した場合，ドライバの過失によるものなのか，マルウェアの車載 LAN への侵入・攻撃によるものなのかを区別する仕組みはない．不具合や事故が起こった後のデジタル・フォレンジックのためには，その原因となった事象の証拠保全を行う機構が必要となる．先行研究として，車載 LAN 上のストレージデバイスを用いて車載 LAN データを保全するフォレンジック機構が提案されているが，車載 LAN データのみ保全するため，いつ感染したのか，どのようなマルウェアなのか等を調べることができない．さらに，マルウェアから証拠保全を妨害されないといった保証もない．本研究では，IVI へ侵入するマルウェアおよび車載 LAN データの証拠保全を行うフォレンジック機構を提案する．マルウェアからの耐性を高める目的として，OS カーネル上にフォレンジック機構を組み込み，マルウェアの証拠データを保全する．フォレンジック機構の評価として，車載 LAN へ侵入するマルウェアとして Mirai に車載 LAN へ DoS 攻撃を行う機能を追加したマルウェアを感染させる実験を行い，その際の証拠データを分析した．実験結果から，保全された証拠データからマルウェアの攻撃時の特徴的なシステムコールや車載 LAN の異常なメッセージの増加，感染時に Telnet 通信が頻繁に行われるといった挙動が観測でき，フォレンジック機構の有効性を確認した．また，フォレンジック機構が，アンチデバッグ等の耐解析手法に対し高い耐性があることと，車載システムにおいて十分なパフォーマンスで動作可能であることを示す．

キーワード：システムセキュリティ，コンピュータウィルス，フォレンジクス，IVI，CAN

Forensics Mechanism in Kernel Land to Preserve Evidence of Malware Intruding into In-vehicle LAN

Shuji Ohira[1,a)] Hiroyuki Inoue[2] Ismail Arai[3] Kazutoshi Fujikawa[3]

Received: June 25, 2018, Accepted: December 4, 2018

Abstract: Cyber-attacks, such as the intrusion of malware on in-vehicle infotainment systems (IVI), are becoming a problem. At present, there is no mechanism to distinguish between an accident caused by a driver's negligence or an accident caused by malware infiltration into an in-vehicle LAN. A mechanism to preserve the evidence data of the accident is needed for the digital forensics following the accident. A previous study has proposed a forensic mechanism for preserving in-vehicle LAN data using storage devices, however the study failed to find out when the system was infected and what kind of malware was used. In addition, there is no guarantee that the storage device will be kept from damage, or that evidence integrity will not be disturbed by malware. In this research, we propose a forensic mechanism on the kernel land that preserves the evidence of the malware invading the in-vehicle LAN. For the purpose of enhancing the resistance to malware, a forensic mechanism was incorporated on the OS kernel and the evidence data of the malware was reserved. As an evaluation of the forensic mechanism, we conducted experiments in which an IVI, incorporating the forensic mechanism, was infected with a malware called Mirai that adds a command to DoS-attack the in-vehicle LAN, and the behavior of the evidence data at that time was observed. From the results, it was found that observation of the characteristic system call at the time of the malware attack and the abnormal message of the in-vehicle LAN from the preserved evidence data is possible. It was also observed that Telnet communication was frequent at the time of infection, therefore, the effectiveness of the forensic mechanism was confirmed. We also showed that the forensic mechanism is highly resistant to debugging and that it can operate with sufficient performance in an in-vehicle system.

Keywords: system security, computer viruses, forensics, IVI, CAN

1. はじめに

インターネットとつながる自動車において，カーナビ等の車載インフォテインメントシステム（以下，IVI; In-Vehicle Infotainment system）へのマルウェアの侵入等による車載 LAN へのサイバー攻撃が問題となっている [1], [2], [3]．さらに，近年の IVI における Linux を使用したプラットフォームの開発 [4] にともない，OS 共通の脆弱性に起因するような脅威が今後いっそう増加すると考えられる．現状では，マルウェアのような不正なプログラムによる異常に起因する事故が発生した場合，ドライバの過失による事故か，マルウェアの車載 LAN への侵入・攻撃による事故かを区別する仕組みはない．事故が起こった後のフォレンジックのためには，サイバー攻撃の証拠保全を行う機構が必要となる．

本研究では，車載 LAN へ侵入するマルウェアの証拠保全を行うカーネル上のフォレンジック機構を提案する．ユーザ空間におけるマルウェアからの証拠保全に対する妨害への対策のために，OS カーネル上にフォレンジック機構を組み込み，マルウェアの挙動の証拠データを保全する．証拠データとして，時刻，システムコール頻度，ファイルアクセス，IP 通信および車載 LAN データを保全する．フォレンジック機構の評価として，フォレンジック機構を組み込んだマシンに，車載 LAN へ不正にアクセスするマルウェアを感染させる実験を行う．実験に用いたマルウェアとしては，ソースコードが開示されており，Linux ベースの装置をターゲットにする Mirai を用い，そこに車載 LAN への DoS 攻撃を行うコマンドを追加したものを使用する．実験結果から，保全された証拠データから Mirai の攻撃時の特徴的なシステムコールや車載 LAN の異常なメッセージの増加，感染時以降 Telnet での通信が頻繁に行われるといった振舞いが保全できていることを確認し，フォレンジック機構の有効性を確認する．さらに，フォレンジック機構はアンチデバッグ等の耐解析手法に対して高い耐性があることを定性評価により示し，マルウェアから検出されにくい機構であることを確認する．また，パフォーマンスの評価として，フォレンジック機構の有無による各システムコールのオーバヘッドの計測と，車載 LAN への DoS 攻撃時のようなシステムに負荷がかかる際のフォレンジック機構の証拠保全性能の計測を行い，フォレンジック機構がある場合のオーバヘッドを比較する．さらに，証拠保全性能の計測を行い，システムの負荷が大きい DoS 攻撃の際の証拠データを取りこぼしなく保全できていることを確認する．

[1] 奈良先端科学技術大学院大学先端科学技術研究科
Graduate School of Science and Technology, Nara Institute of Science and Technology, Ikoma, Nara 630–0192, Japan
[2] 広島市立大学大学院情報科学研究科
Graduate School of Information Sciences, Hiroshima City University, Hiroshima 731–3194, Japan
[3] 奈良先端科学技術大学院大学総合情報基盤センター
Information Initiative Center, Nara Institute of Science and Technology, Ikoma, Nara 630–0192, Japan
[a)] ohira.shuji.ok2@is.naist.jp

2. 関連研究

2.1 車載システムにおける脅威と対策

本節では，本論文内における車載システムの定義と車載システムの構成要素である IVI および車載 LAN の CAN について述べ，それらにおけるセキュリティ上の脅威と対策について述べる．本論文内における車載システムの定義は，車載 LAN，車載 LAN に接続する車載コンピュータである ECU（Electronic Control Unit）や IVI のような自動車における電子機器の構成要素を合わせたものである．

IVI とは，カーナビといった自動車におけるセンタコンソール機器のことである．近年まで，多くの自動車メーカは IVI に独自の OS を使用していたが，コードの再利用性・開発プロセス効率化のためオープンソース化が進み，AGL（Automotive Grade Linux）プロジェクト [4] が開発を行っている Linux ディストリビューションである AGL ディストリビューション[*1]が普及しつつある．これにともなって，独自の OS を使用していたときとは異なり，IVI が OS 共通の脆弱性を持ってしまう点やそれらを狙ったマルウェアの感染の増加が懸念される．

さらに，**図 1** に示すように，携帯電話通信網，Wi-Fi，Bluetooth 経由等，IVI は他のどんな車両コンポーネントよりも外部から接続可能なインタフェースを多数持ち，これらのアタックサーフェスに内在する脆弱性を狙った攻撃を十分に想定する必要がある．外部からの不正アクセスに

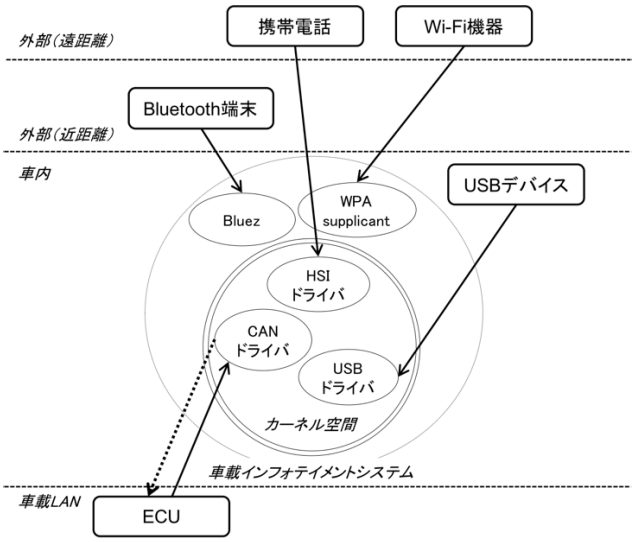

図 1 IVI におけるアタックサーフェス
Fig. 1 Attack surface of in-vehicle infotainment system.

[*1] 本論文中では，AGL という Collaborative Open Source Project を AGL プロジェクトと定義し，AGL プロジェクトで開発を行っている Linux ディストリビューションを AGL ディストリビューションと定義する．

より，IVI を経由し車載 LAN や他の ECU への攻撃を行うことが攻撃シナリオとして考えられる．実際に，報告されている Jeep 車のハッキング [1] は，IVI の脆弱性を利用して車内へ侵入し，IVI 経由でアップデート可能な ECU に対しリプログラミングを行い，車載 LAN へ攻撃した．さらに，Tesla 車のハッキング [3] は，IVI の脆弱性を突き車内へ侵入し，IVI と Ethernet でつながるゲートウェイへバックドアを仕掛け，そこから車載 LAN へ攻撃した．これらの実例における IVI は，図 1 のアーキテクチャとは異なり，IVI の OS が CAN ドライバを保有せず直接 CAN バスにつながっていない．しかし，AGL ディストリビューションのリファレンスハードウェア [5] には，車載ネットワークへのインタフェースがあるため，IVI が直接 CAN バスと接続される車載ネットワークアーキテクチャが想定される．また，実車両に搭載されるハードウェアは，リファレンスハードウェアと異なり，ゲートウェイを挟んで CAN バスへ接続する場合がある．そのため，IVI と CAN バスの間にゲートウェイを設置する車載ネットワークアーキテクチャも想定する．このような車載ネットワークアーキテクチャの場合，IVI 経由でゲートウェイをはじめとする周辺機器への感染活動を行う恐れがあるが，IVI とゲートウェイでマルウェアの感染活動の証拠保全が有効であると考えられる．さらに，AGL プロジェクトは自動車会社の大手数社が参画しており，実車両で運用が開始されていることから，今後普及すると予想される．したがって，本研究では，IVI が直接 CAN バスと接続され，IVI の OS が CAN ドライバを保有する車載ネットワークアーキテクチャと，IVI がゲートウェイを挟んで CAN バスと接続する車載ネットワークアーキテクチャを想定する．

ところで，IVI 経由の車載 LAN への攻撃を想定すると，図 1 の点線の矢印のように攻撃者やマルウェアは，カーネル API であるシステムコールを用いて必ずカーネル空間を経由する必要があり，カーネルにおいて攻撃や感染を検出できる可能性がある．

また，車載 LAN の代表的な規格である CAN [6] では，そのバス上に論理 “0” と論理 “1” が同タイミングで送信されると，“0” が優先される．この性質により，送信先情報の CAN ID と呼ばれる識別子の値が小さな ID が優先度が高い ID として扱われる．CAN のプロトコルでは，送信先情報の CAN ID のみで送信元情報を持たず，また暗号化や認証の仕組みがないため，本質的になりすまし攻撃に対して脆弱である．さらに，帯域 500 kbps と Ethernet 等の LAN と比べると低速であり，優先度の高い大量のメッセージを送信するような DoS 攻撃を比較的性能の低いコンピュータで実現可能である．

以上のように，外部ネットワークにつながる自動車において，車載システムに対するサイバー攻撃の脅威が高まっており，車載システムへのサイバー攻撃の防止・検知といった提案がされている．防止の観点では，CAN におけるメッセージ認証方式 [7] が提案されている．しかし，CAN はペイロードを 8 byte しか持たないため，CAN の仕様を変更せずに実現することは困難である [8] とされている．検知の観点では，CAN メッセージの周期を監視する方式 [9] が提案されているが，CAN には周期を持たないメッセージも存在するため検知可能なメッセージが限られている．以上のように攻撃の防止・検知といった観点で有効な手法は提案されているが，インターネットにおけるサイバー攻撃と同様に，車載システムにおけるサイバー攻撃も完全に無効化することは難しい．仮に未知の脆弱性を突くマルウェアが存在した場合には，マルウェアが車載ネットワークへ攻撃を行うといったシナリオが現実化する恐れがある．以上のことから，車載システムにおいて，攻撃やマルウェア感染を裏付けるデジタル・フォレンジックの必要性が高まってくる．

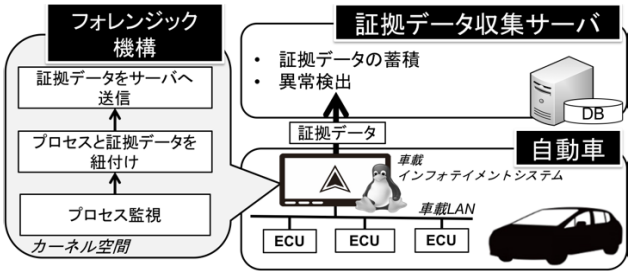

図 2 フォレンジック機構の全体のアーキテクチャ
Fig. 2 Architecture of forensics mechanism.

そこで，デジタル・フォレンジックに活用する証拠データを保全する場所や，証拠保全手法を検討する必要がある．証拠データの保全する場所を，自動車内部の IVI やストレージデバイスにしてしまうと保全可能な証拠データのデータ量がそれらのストレージに依存してしまうため，**図 2** のように，証拠データを外部のサーバ等のストレージへ収集することを考える．そこで課題となるのが，自動車とサーバ間での通信路の安全性である．しかし，ID ベース暗号を用いた認証 [10] や，専用回線を用いた IoT プラットフォームサービス [11] があり，これらを活用すれば通信路の安全性は担保される．そのため，本研究では，自動車とサーバ間での通信路の安全性ではなく，証拠保全手法についてのみにフォーカスを当てる．証拠保全手法では，車載 LAN への攻撃とマルウェアの挙動が統合的に保全可能であることや，マルウェアから証拠保全の妨害を受けないように考慮する必要がある．

以降では，車載 LAN データに対する従来手法とマルウェアの挙動に対する従来手法について述べる．

2.2 車載 LAN データに対する証拠保全手法

車載システムのフォレンジックにおける従来手法として，車載 LAN から得られる速度，エンジン温度，加速度，GPS

情報といった証拠データを犯罪の分析，救助活動，保険金請求に活用することを目的とした証拠収集システム [12] が提案されている．しかし，この手法では自動車のセーフティインシデントの証拠保全を目的としており，マルウェア感染といったサイバー攻撃等のセキュリティインシデントを想定していない．そのため，証拠収集システムが Linux 等の汎用 OS で構築されることもあり，証拠収集システム自体がサイバー攻撃の標的となる可能性がある．

車載 LAN 上のストレージデバイスを用いて車載 LAN データを保全するフォレンジック機構 [13]，スマートフォンと車載ドングルを用いて車載 LAN データを保全するフォレンジック機構 [14] が提案されているが，どちらも車載 LAN データのみ保全するため，どのような機器から車載 LAN に侵入したのか，いつ感染したのか等を調べることができないことや，マルウェア等からの証拠保全を妨害されないといった保証がないという点がある．さらに，車載ドングルを装着することによって自動車へのアタックサーフェスを増やしてしまいスマートフォンや車載ドングルといったデバイスに対するデジタル・フォレンジックも考慮しなければならず，それらすべてのデバイスの証拠データを統合的に集めることは現実的に難しい．

2.3 マルウェアの挙動に対する証拠保全手法

従来のマルウェアの証拠保全は，検体を手に入れ，サンドボックス環境を使って解析・証拠保全を行っていたが，車載システムにおけるマルウェアの解析および挙動の証拠保全は，常時証拠データを保全し，保全した証拠データをフォレンジックに活用可能にしなければならない．そのため，仮想環境を用いたマルウェア解析環境 Alkanet [15] は，仮想環境が前提のため，実環境である車載システムへ直接適用できない．ユーザプロセスを使用してマルウェア解析を行う Cuckoo Sandbox [16], Sysmon [17] は，ユーザプロセスとして動作するため，ユーザ空間で動作するマルウェアから検知されやすい．

一方で，仮想環境を使用せず，ユーザプロセスを使用しない例として，SELinux での Permissive モード [18]，TOMOYO Linux でのマルウェア解析手法 [19]，カーネル空間のデバイスドライバで証拠保全を行う手法 [20] がある．SELinux の有効時のモードとして，Enforcing と Permissive がある．Enforcing モードでは，SELinux ポリシに基づいてシステムコールを用いたファイルやネットワークリソースへのアクセスを制限できる．Permissive モードでは，SELinux ポリシに基づいてシステムコールを用いたファイルやネットワークリソースへの資源へのアクセスのログを残すことが可能である．しかし，SELinux の Permissive モードでは，プロセスがポリシに違反したファイルへアクセスしたときのみ証拠保全されるため，ファイルごとに適切にポリシを定める作業が必要となる．また，マルウェアの活動でポリシに違反しないような行為があった場合は証拠保全が行われないため，証拠保全を目的とするフォレンジック機構において SELinux は適していない．さらに，SELinux ではプロセスがシステムコールを用いてポリシに違反したファイルへアクセスしたときに，システムコールの数値やポインタといった表面的な引数や戻り値のみしか取得できないため，車載システムでのフォレンジックで重要となる車載 LAN のメッセージの内容については取得することができない．TOMOYO Linux でのマルウェア解析手法は，セキュア OS である TOMOYO Linux の持つ振舞い学習機能を用いてマルウェアが行うシステムコールを把握し，マルウェアを機械語レベルではなく抽象度の高いシステムコールレベルで解析可能となる．しかし，この手法は，振舞い学習機能で学習したマルウェアでは有効だが，学習していないマルウェアには有効でない可能性がある．さらに，デバイスドライバで証拠保全を行う手法は，カーネル空間のデバイスドライバに用いて証拠保全を行うため，Cuckoo Sandbox や Sysmon とは異なりユーザプロセスを使用せず，ユーザ空間のマルウェアに対し検知されずに証拠保全を行うことが可能である．しかしながら，Windows 向けに設計・実装されている点と車載システム上ではより厳しいパフォーマンスを求められる点から，車載システムにおいて直接適用することは難しい．

2.2 節では，車載 LAN データに対する証拠保全手法について述べたが，感染したマルウェアの情報を得るには，車載 LAN データのみではなく，IVI の挙動を示す証拠データも必要となる．さらに，マルウェアから証拠保全の妨害を受けないために，マルウェアから証拠保全を検出されない機構でなければならない．本節では，マルウェアの挙動に対する証拠保全手法について述べたが，車載システム向けの証拠保全手法はなく，パフォーマンスや実装上の問題で従来手法を直接適用することは難しい．

以上の議論をふまえ，車載システムにおけるフォレンジック機構の要件を下記のとおりまとめた．

要件 I. 車載システム全体の統合的な証拠保全
要件 II. マルウェアから検知されない機構
要件 III. 車載システムで導入可能なパフォーマンス

3. 提案方式

3.1 フォレンジック機構の設計

本節では，マルウェアから妨害されず車載システムの統合的な証拠保全を行うフォレンジック機構の設計について述べる．提案方式は，従来方式 [13], [14] とは異なり，IVI へフォレンジック機構を組み込むため車載 LAN 上のストレージデバイス等の追加を必要とせず，また，カーネル空間で実装することでユーザ空間で動作するマルウェアに検出されにくいという特徴がある．さらに，車載 LAN データのみではなく IVI の証拠データを保全するため，攻撃だ

けではなく，いつ感染したのか，どのような特徴があるマルウェアなのかを後から証拠データによって確認できるという特徴がある．SELinux や TOMOYO Linux と異なり，正しくポリシを定める作業を必要とせず，ポリシに違反しないマルウェアの活動も証拠保全可能である．さらに，IVI のすべてのシステムコールの統計量を証拠保全するため，後にそれらの証拠データに対し学習アルゴリズム等を用いて異常を検出することが期待できる．以上のような提案方式を 2 章で述べた要件によって設計する．

要件 I より，提案するフォレンジック機構は，車載 LAN への攻撃や攻撃の原因となるマルウェアを認識するために，車載システム全体の統合的な証拠保全を行わなければならない．したがって，フォレンジック機構は，車載 LAN や自動車においてアタックサーフェスとなる IVI の証拠保全を行わなければならない．そのため，車載 LAN や外部ネットワーク等とつながる IVI にフォレンジック機構を組み込むことが望ましい．よって，IVI 上でマルウェアの証拠データや車載 LAN データを保全することで，要件 I を満たす．

要件 II より，提案するフォレンジック機構は，マルウェアからの証拠保全に対する妨害対策のため，マルウェアから検知されない機構でなければならない．そこで，マルウェアを監視するプロセスによって証拠保全を行う機構や仮想化環境でマルウェアの証拠保全を行う機構が考えられる．しかし，プロセスによって証拠保全を行う機構ではマルウェアと同様のユーザ空間で証拠保全するため，マルウェアから証拠保全を検出・妨害される恐れがある．また，仮想化環境でマルウェアの証拠保全を行う機構は，実環境である車載システムへの適用は不可能である問題がある，これより，フォレンジック機構をユーザ空間のマルウェアが触れることのできないカーネル空間上のデバイスドライバとして実装することで，マルウェアからの証拠保全の検出・妨害を低減させ，要件 II を満たす．

要件 III より，提案するフォレンジック機構は，実際の自動車での運用を考えると，車載システムで導入可能なパフォーマンスでなければならない．そこで，フォレンジック機構が引数や戻り値を証拠保全するシステムコールは，マルウェアが使用するシステムコールに絞る．たとえば，マルウェアによるファイルの改ざん等に関しては，ファイルを読み書きするシステムコールが必要である．そのため，open といったシステムコールでファイルパス等の引数の証拠保全を行う．また，フォレンジック機構が引数や戻り値を証拠保全しないシステムコールは，システムコール頻度のみを計測といった簡単な処理を行う．また，後述の CAN における最小送信周期が 222 μs であることから，提案方式による遅延は 10 μs オーダの遅延となるようにしなければならない．以上の設計によって，要件 III を満たす．

3.2 フォレンジック機構における証拠データ

本節では，車載システムにおけるフォレンジック機構の証拠データについて述べる．

IVI では，カーナビの情報（GPS データ，地図データ）の処理や，VICS 情報，オーディオデータ等の多種多様な I/O が発生することが想定される．しかしながら，フォレンジックの対象は車載 LAN へ攻撃するマルウェアの証拠データであるため，マルウェアが CAN ドライバ等のネットワークインタフェースや他のファイルといった資源にシステムコールを用いて作用する際の証拠を保全する必要がある．したがって，提案方式では，フォレンジックに必要となる車載 LAN へ攻撃するマルウェアの証拠データを保全する．対象とする証拠データの種類としては，CAN 通信，IP 通信，ファイルアクセスおよびシステムコールの頻度となり，その情報量は，一般的な車載 LAN における CAN 通信では 1 時間あたり約 40 MB（実トラフィックの観測経験による），IP 通信は携帯電話網を使った定額データ通信として 1 時間あたり約 20 MB（目的地等の検索といった Web 閲覧を想定），ファイルアクセスとシステムコールの頻度はそれぞれ 1 時間あたり約 10 MB のオーダ（実際にシステムを動作させ算出）であると想定する．また，提案方式では，IVI で動作するすべてプロセスのシステムコールをフックして証拠保全を行うため，マルウェアによる DoS 攻撃といった動作を必ず証拠保全可能である点から，車載 LAN へ DoS 攻撃するマルウェアの証拠データを保全できるといえる．したがって，提案方式は車載 LAN へ DoS 攻撃するマルウェアに対し，有効なフォレンジック環境であると考えられる．

さらに，悪意のあるプロセスによって，証拠保全を妨害されたりすることがなく，安全に証拠保全を行う必要もある．悪意のあるプロセスの挙動を安全に証拠データとして保全するためには，マルウェアの耐解析手法による検知からフォレンジック機構を保護しなければならない．そこで，ユーザプロセスが触れることのできないカーネル空間でフォレンジック機構を実装することでこれに対処する．カーネル空間でプロセスの挙動を収集する方法として，プロセスがカーネル空間とのやりとりを行う API であるシステムコールをフックし，そこで証拠保全を実行する方法が考えられる．システムコールをフックし，そこで証拠保全を実行することによって，どのプロセスがどんな引数でどのシステムコールをコールしたかをカーネル空間内で得ることが可能になる．したがって，フォレンジック機構は，システムコールをフックし証拠保全を行う．

図 3 にフォレンジック機構における証拠データの例を示す．まず，フォレンジック機構は時刻やシステム全体の振舞いとして全システムコール頻度を証拠データとして保全する．さらに，フォレンジック機構は，システムコールをフックし証拠保全を行うため，すべてのシステムコールで

```
<システムコール頻度>
[ 3555.977177] [forensics_count] Time:1523341286.494921962 SYSCALL_No.0:65
[ 3555.977180] [forensics_count] Time:1523341286.495034111 SYSCALL_No.1:41
[ 3555.977181] [forensics_count] Time:1523341286.495050548 SYSCALL_No.2:3
[ 3555.977182] [forensics_count] Time:1523341286.495074758 SYSCALL_No.3:4
<ファイルアクセス>
[ 4819.214842] [forensics_fileaccess] Time:1523341286.494921962 PID:3030 Accessed_File:/proc/self/oom_score_adj
[ 4819.214952] [forensics_fileaccess] Time:1523341286.495034111 PID:3030 Accessed_File:/sys/module/fsm/uevent
[ 4819.214968] [forensics_fileaccess] Time:1523341286.495050548 PID:3030 Accessed_File:/sys/module/fsm/uevent
[ 4819.214993] [forensics_fileaccess] Time:1523341286.495074758 PID:3030 Accessed_File:/run/udev/data/+module:fsm
<車載LANデータ>
[ 4821.826686] [forensics_can_recvmsg] Time:1523341289.108072681 PID:2692 CAN_PACKET:45c[8]5c00800000000000
[ 4821.835447] [forensics_can_recvmsg] Time:1523341289.116837540 PID:2692 CAN_PACKET:440[8]40008011010f0f0f
[ 4821.881328] [forensics_can_recvmsg] Time:1523341289.162742328 PID:2692 CAN_PACKET:442[8]42008000000f0f0f
[ 4821.931797] [forensics_can_recvmsg] Time:1523341289.213237319 PID:2692 CAN_PACKET:44d[8]4d00800000000000
<IP通信>
[ 4821.826503] [forensics_ip] Time:1523341289.107888652 PID:3031 IP:3.0.0.0 Port:51900
[ 4821.826573] [forensics_ip] Time:1523341289.107960283 PID:3031 IP:3.0.0.0 Port:51900
[ 4821.835404] [forensics_ip] Time:1523341289.116792885 PID:3031 IP:3.0.0.0 Port:51900
[ 4821.881296] [forensics_ip] Time:1523341289.162707590 PID:3031 IP:3.0.0.0 Port:51900
```

図 3 フォレンジック機構における証拠データの例
Fig. 3 Example of evidence data in forensics mechanism.

引数や戻り値の証拠保全が可能である．しかしながら，すべてのシステムコールで引数や戻り値の証拠保全を行うと，証拠データのデータ量の増大や，システムのパフォーマンスの低下を招く．そこで，フォレンジック機構は，効率的にマルウェアの挙動を観測するためにシステムコールの引数や戻り値の証拠保全を行うシステムコールを限定する．引数や戻り値の証拠保全を行うシステムコールとして，車載 LAN データの送信と受信の証拠保全のためにそれぞれ write と recvmsg システムコール，プロセスのファイルアクセスの証拠保全のために open システムコール，プロセスの IP 通信の証拠保全のために sendto システムコールの引数を保全する．また，フォレンジック機構は，IVI が接続されている車載 LAN に流れているすべての通信を取得し，IVI が送信したデータと IVI が受信したデータの識別が可能である．なお，車両内部で CAN バスが複数ある場合は，IVI に直接接続されている CAN バスのみが証拠保全の対象となる．

3.3 フォレンジック機構の実装

フォレンジック機構は，マルウェアからのアンチデバッグ等の耐解析手法からの検知を保護するためカーネル空間に実装する．フォレンジック機構の実装については，デバイスドライバとして実装し，システムコールのたびに証拠データを保全する．

フォレンジック機構の実装の概要について述べる．実装したフォレンジック機構のフローを図 4 に示す．フォレンジック機構は，デバイスドライバとしてカーネルにロードされ，ロードされる際に，カーネル空間のシステムコール関数の関数ポインタが格納されている配列のシステムコールテーブルをフォレンジック機構の関数ポインタに書き換える．システムコールテーブルをフォレンジック機構のシステムコール関数の関数ポインタに書き換えると，プロセスからシステムコールが発行された際，フォレンジック機構の処理が実行され，フォレンジック機構内で正規のシステムコールを発行することで通常どおりの OS カーネルの処理が実行される．提案するフォレンジック機構では，システムコールの頻度やプロセスのファイルアクセスやプロセスの IP 通信，車載 LAN データ等の証拠データを保全する．その実装方法は，open や sendto 等といった特定のシステムコールが発行されるとシステムコールの引数をカーネルバッファに出力するような方法である．システムコールの発行元のプロセスを特定するために getpid システムコールをフォレンジック機構内で呼び出している．

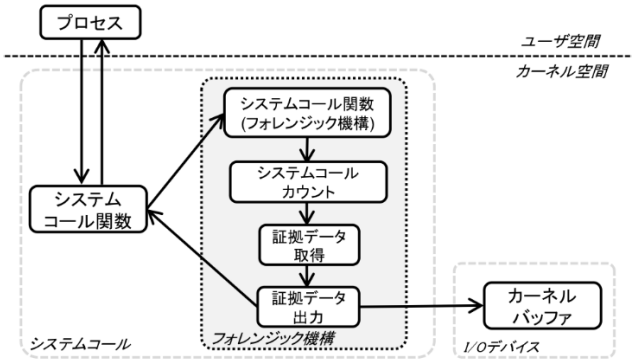

図 4 フォレンジック機構のフロー
Fig. 4 Flow of forensics mechanism.

以上のようなフォレンジック機構をデバイスドライバとして実装し，証拠データを保全した．

4. 評価

4.1 車載 LAN へ侵入するマルウェアを用いた評価

本節では，提案方式を組み込んだ車載システムがマルウェアに感染した際の評価を行う．本研究で想定している車載ネットワークアーキテクチャは，今後普及が予想される AGL ディストリビューションを想定したアーキテクチャであり，IVI が直接 CAN バスと接続されるアーキテク

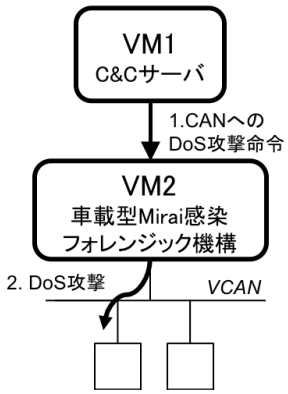

図 5 実験環境
Fig. 5 Experiment environment.

チャと，IVI がゲートウェイを挟んで CAN バスと接続するアーキテクチャである．このアーキテクチャで想定される脅威として，IVI やゲートウェイの脆弱性を突き侵入し，車載 LAN へ DoS 攻撃するマルウェア等が考えられる．そこで，提案方式の評価として，IoT 機器における代表的なマルウェアである Mirai [21] を感染させ提案方式の有効性を評価する．Mirai はボットと呼ばれる種類のマルウェアで，C&C（Command & Control）サーバと呼ばれる攻撃命令を行うサーバから命令を受けて攻撃等を行う．Mirai はソースコードが公開されており [22]，誰でもそのソースコードを取得し，実行することが可能なため，Mirai に類似したマルウェアが多数報告されている [23]．本研究では，車載システムを標的とした Mirai の類似マルウェアを想定し，評価に用いる検体を，Mirai の SYN フラッド攻撃をはじめとする 10 種類の DoS 攻撃コマンドに車載 LAN である CAN への DoS 攻撃コマンドを追加したマルウェアとした．本来，Mirai は複数の IoT 機器に感染し，標的となるサーバへ DDoS 攻撃を行うマルウェアであるが，本研究の標的は Ethernet 等よりも低速な 500 kbps の CAN であり，Mirai が感染させた 1 台の機器でバスの占有が十分可能であるため，本評価に採用した．さらに，実際のマルウェアを用いることで，そのマルウェア特有の挙動が証拠データから観測可能か確認できることも採用の根拠となる．以降，CAN への DoS 攻撃コマンドを追加したマルウェアを車載型 Mirai と呼ぶ．

車載型 Mirai の特徴として，車載 LAN への DoS 攻撃を行う特徴と，改造前の Mirai の特徴と同様に Telnet 通信を使って他の IoT 機器に対して自身の感染の拡大を行う点やプロセスを精査し競合するマルウェアがいないか調査する点があげられる．提案方式において車載型 Mirai の列挙した特徴が保全できているかを評価する．まず，実験環境について述べる．実験環境を**図 5** に示す．実験開始から 20 s ごとに非感染時，感染時，攻撃時の状態になるように実験を行った．図 5 のように，攻撃時には C&C サーバと見立てた VM1 から車載 LAN への DoS 攻撃命令を車載型

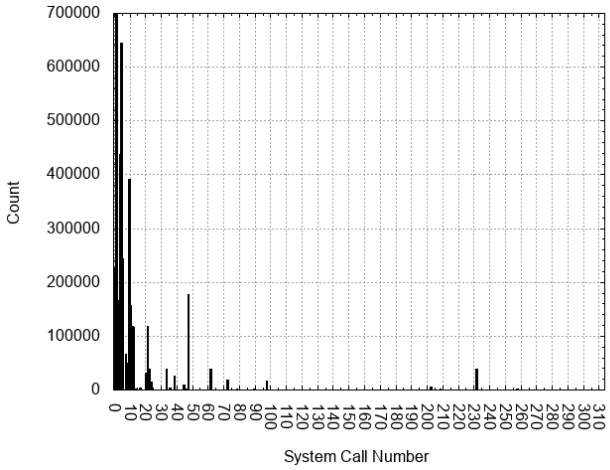

図 6 非感染時，感染時，攻撃時の全システムコール
Fig. 6 All system call during no infection, infection, attack.

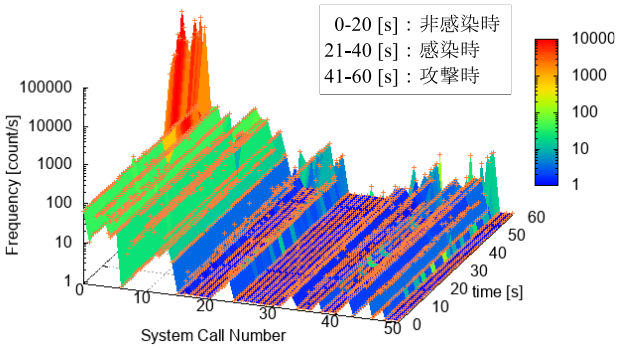

図 7 非感染時，感染時，攻撃時の 0–50 番の時系列システムコール
Fig. 7 No.0–50 system call frequency during no infection, infection, attack.

Mirai 感染した VM2 に送る．攻撃命令を受け VM2 は車載 LAN へ DoS 攻撃を行う．今回の実験では，仮想環境で車載 LAN のインタフェースを起動できる VCAN（Virtual CAN）を用いて仮想環境上で車載システム環境を構築した．

実験結果として，1 分間の全プロセスが呼ぶシステムコールの累積回数が得られた．全プロセスが呼ぶシステムコールを**図 6** に示す．図 6 より，0〜50 番のシステムコールが主に呼ばれ，それ以外の 51 番以上のシステムコールは比較的呼び出し回数が少ないことが分かった．そこで，0〜50 番のシステムコールに着目し，それらを時系列にしたシステムコールを**図 7** に示す．攻撃時である 40 s のときに，0–10 番あたりのシステムコールが急激に増加していることが分かった．これは，攻撃時に車載型 Mirai によって write のシステムコールが急激に呼ばれたためである．

さらに，車載型 Mirai がファイルアクセスを行ったことを示す証拠データが得られたかどうか確認する．車載型 Mirai のプロセスがファイルアクセスを行ったときの証拠データを**図 8** に示す．図 8 より，車載型 Mirai のプロセスが他のプロセスを調査するために PID の昇順に/proc/PID/exe のファイルにアクセスしていることが分かる．これより，

```
<システムコール頻度>
[  568.334481] [forensics_count] Time:1528974433.932550875 System Call No.0 : 70
[  568.334484] [forensics_count] Time:1528974433.932553929 System Call No.1 : 19504
[  568.334487] [forensics_count] Time:1528974433.932557002 System Call No.2 : 51
[  568.334490] [forensics_count] Time:1528974433.932560077 System Call No.3 : 66
<ファイルアクセス>
[ 1104.431343] [forensics_fileaccess] Time:1526258414.036488189 PID:3151 Accessed_File:/usr/lib/gvfs/gvfs-gphoto2-volume-monitor
[ 1104.431356] [forensics_fileaccess] Time:1526258414.036502861 PID:3151 Accessed_File:/proc/2490/exe
[ 1105.431754] [forensics_fileaccess] Time:1526258415.037399786 PID:3151 Accessed_File:/usr/lib/gvfs/gvfs-afc-volume-monitor
[ 1105.431764] [forensics_fileaccess] Time:1526258415.037411258 PID:3151 Accessed_File:/proc/2501/exe
[ 1106.438534] [forensics_fileaccess] Time:1526258416.044683357 PID:3151 Accessed_File:/usr/lib/x86_64-linux-gnu/gconf/gconfd-2
[ 1106.438550] [forensics_fileaccess] Time:1526258416.044699957 PID:3151 Accessed_File:/proc/2504/exe
[ 1107.438469] [forensics_fileaccess] Time:1526258417.045118750 PID:3151 Accessed_File:/usr/lib/gvfs/gvfsd-trash
[ 1107.438480] [forensics_fileaccess] Time:1526258417.045130798 PID:3151 Accessed_File:/proc/2563/exe
<車載LANデータ>
[  568.334450] [forensics_can_write] Time:1528974433.932520207 PID:3945 CAN_PACKET:000[8]0000000000000000
[  568.334453] [forensics_can_write] Time:1528974433.932523262 PID:3945 CAN_PACKET:000[8]0000000000000000
[  568.334456] [forensics_can_write] Time:1528974433.932526327 PID:3945 CAN_PACKET:000[8]0000000000000000
[  568.334459] [forensics_can_write] Time:1528974433.932529409 PID:3945 CAN_PACKET:000[8]0000000000000000
<IP通信>
[  385.444864] [forensics_ip] Time:1528973624.512205228 PID:2706 IP:34.115.102.164 Port:2323
[  385.444867] [forensics_ip] Time:1528973624.512208262 PID:2706 IP:148.52.22.51 Port:23
[  385.444870] [forensics_ip] Time:1528973624.512211487 PID:2706 IP:206.197.15.76 Port:23
[  385.444874] [forensics_ip] Time:1528973624.512214570 PID:2706 IP:99.117.88.178 Port:23
```

図 8　車載型 Mirai 感染時・攻撃時の証拠データ
Fig. 8　Evidence data during infection, attack of in-vehicle type Mirai.

提案方式によって改造前の Mirai の特徴であるプロセスを精査し競合するマルウェアがいないか調査を行う特徴が保全されていることが確認できた．

次に，車載型 Mirai が IP 通信を行ったことを示す証拠データが得られたかどうか確認する．図 8 より，車載型 Mirai のプロセスがランダムな IP アドレスに 23 番および 2323 番ポートに通信を行っていたことが分かった．これより，改造前の Mirai 特徴である Telnet での通信を行う振舞いが保全されていることが確認できた．

最後に，車載型 Mirai が車載 LAN へメッセージ送信を行ったことを示す証拠データが得られたかどうか確認する．図 8 より，車載型 Mirai のプロセスが車載 LAN へ CAN ID およびペイロードがすべて “0” となる最も優先度が高いメッセージで DoS 攻撃を行っていたことが分かった．これより，車載型 Mirai 特徴である車載 LAN へ DoS 攻撃を行う振舞いが保全されていることが確認できた．また，図 8 のシステムコール頻度によって 1 番の write システムコールが DoS 攻撃メッセージが観測された時刻と近い時刻に 2 万回近く呼ばれていることが確認され，IVI 経由の攻撃であることが示せた．

以上の車載型 Mirai を用いた実験より，提案方式によって，従来手法では保全できていなかった IVI の DoS 攻撃時の挙動と DoS 攻撃時の車載 LAN データが同時に得られることが証拠データから示すことができた．

4.2　耐解析手法に対する評価

本節では，アンチデバッグをはじめとする耐解析手法に対する評価について述べる．対象とする耐解析手法は，Alkanet [15] と同様に，実行時間による検出手法は Branco らによる分類 [24] に基づいている．しかし，Branco らによる分類は Windows を対象としている．このことから，Windows 上の解析ツールの検出手法を提案方式に対して評価することは不可能であるため，Linux の標準的なデバッガである GDB の検出手法 [25] で提案方式の耐解析手法に対する定性的な評価を行う．

表 1　耐解析手法に対する定性評価結果
Table 1　Qualitative evaluation results against anti-debugging techniques.

耐解析手法	回避可能
(1) 仮想化環境検出手法	
全ての手法	○
(2) GDB の検出手法	
ptrace	○
readlink	○
ブレイクポイント	○
(3) 実行時間による検出手法	
RDTSC 命令	○
API による時間計測	○
システムコールによる時間計測	△

表 1 に提案方式の耐解析手法に対する定性評価結果を示す．(1) 仮想化環境検出手法に関して，提案方式はデバイスドライバとして実装されており，実環境でも組み込むことが可能であるため，すべての仮想化環境検出手法を回避することができるといえる．

次に，(2) GDB の検出手法に関して述べる．ptrace はプロセスをトレースするシステムコールであり，ptrace されているプロセスをさらに ptrace すると，−1 を返す．GDB は ptrace をコールしているため，以上の原理から GDB を検出することが可能となる．提案方式では，ptrace のようなプロセスを使用せず証拠保全を行うため ptrace を用いた GDB の検出手法を回避可能である．readlink はシンボリックリンクの値を取得するシステムコールであり，動作しているプロセスの実行ファイルへのシンボリックリンク /proc/PID/exe を引数に実行することで，/usr/bin/gdb が実行されているか検出できる．ptrace の場合と同様に，

表 2 計測環境
Table 2 Measurement environment.

	VM2(ゲスト)	ホスト
OS	Ubuntu 14.04 64bit	macOS 10.13.1
CPU	Intel Core i5 2.7GHz	Intel Core i5 2.7GHz
RAM	4GB	8GB
Storage	10GB	250GB

提案方式では，プロセスを使用しないためこの検出手法も回避可能であるといえる．ブレイクポイントは，GDB がプロセスを停止させる int3 オペコード（0xCC）と呼ばれるオペコードを，任意のアドレスを上書きすることで実現される．したがって，ブレイクポイントを用いた GDB の検出手法は，オペコードと 0xCC を比較することで検出できる．提案方式は，ブレイクポイントでプロセスを停止させて証拠保全を行うような機構でないため，この検出手法も回避可能であるといえる．

最後に，(3) 実行時間による検出手法に関して述べる．Alkanet と同様に，Time Stamp Counter と呼ばれるタイムスタンプを参照する RDTSC 命令や，API による時間計測では，提案方式がシステムコール発行時にのみ動作するため，検知されない．一方で，time や clock_gettime 等のシステムコールによる時間計測の場合，提案方式の処理の遅延により，検知される恐れがある．しかし，このような検出手法に関しては，提案方式内の処理にかかった時間を計測しておき，その分減算した値に返り値を改ざんすることで検出を防ぐことが今後の課題となる．さらに，改ざんを行う機構が車両の安全性に影響を与えないようにする必要もある．提案方式による改ざんの値が，IVI で想定される処理に対して許容できる範囲であるかを，4.3 節のパフォーマンス評価で議論し，車両の安全性に影響を与えないことを確認する．また，証拠保全する時刻は改ざんされていない値とすることで，提案方式で得られた証拠データは実際の挙動を反映したものとなり，証拠データ自体の安全性に影響はないと考えられる．

4.3 パフォーマンス評価

本節では，提案方式の有無による各システムコールのオーバヘッドの計測と，車載 LAN への DoS 攻撃時のようなシステムに負荷がかかる際の提案方式の証拠保全性能の計測を行い，提案方式のパフォーマンスについての評価を行う．

計測を行った環境を**表 2** に示す．オーバヘッドの計測を行う項目は，ファイルアクセスの open システムコール，車載 LAN 送受信の write と recvmsg システムコール，IP 通信の sendto システムコールである．各システムコールを 100 回呼び出す操作を 100 回試行した時間の平均値を計測値とし，提案方式の有無による時間を計測した．**図 9** に提案方式の有無によるシステムコールの計測値を示す．こ

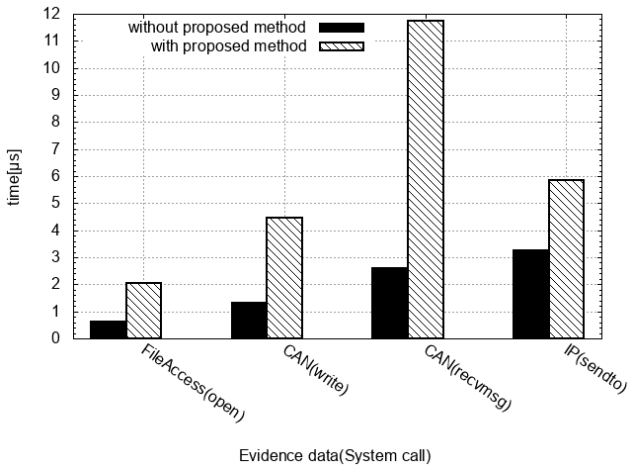

図 9 オーバヘッド計測結果
Fig. 9 Result of overhead measurement.

の差分が各システムコールの保全に要するオーバヘッドとなる．各システムコールのオーバヘッドは，ファイルアクセスの open ついては 1.40 μs，車載 LAN 送受信の write と recvmsg ではそれぞれ 3.14 μs と 9.14 μs，IP 通信の sendto では 2.61 μs であることが分かった．

車載システムにおいてこの遅延が許容範囲であるか車載 LAN の伝送速度と比較して考察を行う．まず，車載 LAN において代表的な規格である CAN の場合を考える．CAN の帯域 500 kbps でペイロードが最大長の 64 bit であるようなメッセージが帯域の限界まで送信されることを考えると，CAN メッセージは最大の 111 bit となるため，単位時間あたり 500×10^3 [bit/s]/111 [bit] = 4,504 [メッセージ] が送信される．したがって，それらのメッセージ間隔は，1 [s]/4,504 [メッセージ] = 222 [μs] となる．CAN の帯域の限界まで最大の 111 bit メッセージを送信したときのメッセージ間隔 222 μs と，提案方式ありの場合の遅延を比べると，提案方式にかかる遅延はどのシステムコールにおいても非常に小さいことが確認できた．

また，AGL ディストリビューションのリファレンスハードウェア [5] より，IVI で実行される処理としては，インターネットや CAN といったネットワーク通信，Bluetooth，GPS，USB といったものが想定される．これらの中で遅延を許容する処理が何かを明確にし，それらの影響を考察する．

USB は，地図の更新や音楽再生で使用される．地図の更新は，ディーラに行った際等に更新されるため，遅延の影響はない．USB による音楽再生，インターネット通信，そして，音声データをストリーミングする際の手順等が定義されている Bluetooth プロファイルの A2DP [26] はバッファリングを用いて通信を行う時点で，遅延が許容されていると考えられる．また，IVI では，ハンズフリーでの通話を可能とする Bluetooth プロファイルの HFP [27] が用いられている．HFP は，最も高音質・広帯域なコーデック

である mSBC でも 64 kbps 程度で，500 kbps の CAN よりも低速である．そのため，提案方式による遅延は許容できる範囲と考えられる．また，これまでに考察した IVI の処理は，自動車の制御情報のような重要な情報をやりとりしない処理であるが，次に，遅延が許されない制御情報に関する処理について考察する．

IVI において，自動車の制御情報を扱うような遅延が許されない処理として，ルート案内に用いる GPS 処理，GPS 補正のための車速パルス等の処理，自動車の制御情報をやりとりする CAN があげられる．一般に，GPS 処理は，GPS 受信機からシリアル通信を用いて緯度経度といった情報を取得しアプリケーションへ反映させる．この際，標準的な GPS フォーマットである NMEA で定められているシリアル通信のビットレートは 4,800 bps である．一方で，CAN は 500 kbps であり，GPS 処理におけるビットレートよりもはるかに高速なため，CAN で要求される遅延の方が，GPS 処理で要求される遅延よりもはるかに厳しい条件となる．さらに，GPS 補正のための車速パルス等も数十 ms オーダの信号であるため，CAN の遅延の許容時間の 10 μs オーダの方がより制限された条件といえる．したがって，3.1 節で述べた CAN の遅延の許容時間である 10 μs オーダであれば，想定する IVI での処理に対し影響がなく，提案方式を組み込めると考えられる．

IVI 自体のシステム動作への影響として，提案方式の CPU 実行命令時間より，考察を行う．提案方式の命令数は，最もオーバヘッドが大きかった CAN の受信を行うシステムコール recvmsg で算出を行った．最もオーバヘッドが大きいシステムコール recvmsg において，提案方式によって追加された最大の命令数は x86_64 の CPU で 72 命令であった．この実行に要する時間は，AGL ディストリビューションのリファレンスハードウェアの 1 つである Intel MinnowBoard Turbot Quad-Core の動作周波数 1.91 GHz において，前述の CAN の遅延の許容時間の 10 μs に比べて十分に小さい値である．したがって，提案方式は AGL ディストリビューションのリファレンスハードウェアを用いた IVI 単体という評価環境においては，IVI での CAN や GPS のメッセージ通信処理に影響を与えることはないといえる．なお，より現実的な評価環境として，実車両上での提案方式の検証が今後の課題となる．

続いて，車載 LAN への DoS 攻撃時のようなシステムに負荷がかかる際の提案方式の証拠保全性能の計測も同様の表 2 の環境で行った．証拠保全性能の計測は，車載 LAN への DoS 攻撃メッセージを 1,000 個送信し，提案方式が取りこぼしなく証拠保全できているかを計測する．

計測した結果，1,000 個の DoS 攻撃メッセージに対し，提案方式は 1,000 個取りこぼしなく証拠保全できていることを確認した．これより，車載 LAN への DoS 攻撃時のようなシステムに負荷がかかる場合であっても提案方式の証拠保全性能は十分であることが確認された．

以上のパフォーマンスの評価によって，車載 LAN の最小送信周期よりも提案方式ありの場合の遅延は小さく，DoS 攻撃メッセージを取りこぼしなく証拠保全できることから，提案方式は IVI 単体での検証においては，実現可能であることを確認した．

5. おわりに

5.1 まとめ

本研究では，車載 LAN に侵入するマルウェアの証拠保全を行うフォレンジック機構の提案を行った．提案方式をカーネル空間にデバイスドライバとして実装し，Linux を搭載した IoT 機器に感染する代表的なマルウェアである Mirai を用いて提案方式の評価を行った．評価実験には，Mirai の類似マルウェアを想定し，Mirai に車載 LAN への DoS 攻撃コマンドを追加したマルウェア車載型 Mirai を実装し，実験に用いた．実験結果より，提案方式によって車載型 Mirai 感染・攻撃時の車載 LAN データや IVI の挙動といった車載システム全体の証拠が統合的に保全されていることを確認できた．さらに，フォレンジック機構はアンチデバッグ等の耐解析手法に対して高い耐性があることを定性評価により示し，マルウェアから検出されにくい機構であることを確認した．また，パフォーマンスの評価として，提案方式ありの場合の遅延が車載 LAN の最小送信周期よりも小さいことを確認した．提案方式はシステムに負荷がかかるような DoS 攻撃時のメッセージでさえも取りこぼしなく証拠保全できることから，提案方式は IVI 単体での検証においては，実現可能であることを確認した．

5.2 今後の課題

今後の課題として，4.3 節で述べた実車両上での提案方式のパフォーマンス検証，車載 LAN データの保全場所の検討，他の CPU アーキテクチャでの実装があげられる．本研究では，証拠データをローカルに保全する実装となっているが，フォレンジック機構に証拠データを外部のサーバへ安全に送信し蓄積する機能を追加することでより実用的な実装となる．その際の課題として，一般的な車載 LAN データのデータ量が 1 時間あたり 40 MB と膨大であることがあげられる．そこで，筆者らが以前に提案した車載 LAN 特有の特徴に着目したリアルタイムデータ圧縮方式 [28] を組み込み，データ量の課題を解決することが考えられる．なお，提案方式は，マルウェアによる耐解析手法であるシステムコールによる時間計測さえ対策できれば，マルウェアに検出されることなく証拠保全可能な環境といえるが，あくまで定性的な評価であるため，実装してすべての耐解析手法によって検出されないかを確認することが必要となる．さらに，提案方式は，証拠保全のみ行うため，マルウェアの検知・駆除を行わない．しかし，提案方式で得ら

れた証拠データからマルウェアを検出できた場合，駆除することが自動車における安全性の観点から最も望ましいといえる．したがって，IVI にすべての耐解析手法の回避機能を実装し，実機によるパフォーマンス評価することと，マルウェアの検出・駆除する手法を検討し実装・評価することが今後の課題となる．また，今回の実装は x86_64 の CPU アーキテクチャであるが，組み込みシステムでよく使用される ARM や V850 のような CPU における，Linux のカーネルコードでも提案したフォレンジック機構が実装可能であるか確認する必要がある．

参考文献

[1] Miller, C. and Valasek, C.: Remote Exploitation of an Unaltered Passenger Vehicle, *Black Hat USA 2015*, pp.1–91 (Aug. 2015).

[2] Ezaki, T., Date, T. and Inoue, H.: An Analysis Platform for the Information Security of In-vehicle Networks Connected with the External Networks, *The 10th International Workshop on Security* (*IWSEC2015*), *Advances in Information and Computer Security* (LNCS 9241), pp.301–315 (Aug. 2015).

[3] Nie, S., Liu, L. and Du, Y.: Free-Fall: Hacking Tesla from Wireless to CAN Bus, *Black Hat USA 2016*, pp.1–16 (Aug. 2016).

[4] Automotive Grade Linux: Automotive Grade Linux, available from ⟨https://www.automotivelinux.org⟩ (accessed 2018-05-30).

[5] Automotive Grade Linux Wiki: AGL Distribution, available from ⟨https://wiki.automotivelinux.org/agl-distro⟩ (accessed 2018-11-10).

[6] International Organization for Standardization: Road vehicles, controller area network (CAN), Part 1: Data link layer and physical signaling, ISO IS11898-1 (2015).

[7] 倉地　亮，松原　豊，高田広章，上田浩史，堀端啓史：メッセージ認証を用いた CAN の集中監視システム，電子情報通信学会論文誌 A，Vol.J99-A, No.2, pp.118–130 (2016).

[8] Dariz, L., Selvatici, M., Ruggeri, M., Costantino, G. and Martinelli, F.: Trade-off analysis of safety and security in CAN bus communication, *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems* (*MT-ITS*), pp.226–231 (June 2017).

[9] 倉地　亮, 高田広章, 上田浩史, 堀端啓史：車載制御ネットワークにおける送信周期監視システムの提案，*SCIS2015*, pp.1–7 (Jan. 2015).

[10] 金森健人，江崎貴也，手柴瑞基，井上博之，小畑博靖，石田賢治：車載器とサーバ間の相互認証に ID ベース暗号を用いた車載 LAN データ収集システム，コンピュータセキュリティシンポジウム 2016 論文集，pp.391–396 (Oct. 2016).

[11] SORACOM: SORACOM Direct, available from ⟨https://soracom.jp/services/direct/⟩ (accessed 2018-05-30).

[12] Chae, K., Kim, D., Jung, S., Choi, J. and Jung, S.: Evidence Collecting System From Car Black Boxes, *Consumer Communications and Networking Conference* (*CCNC*), *2010 7th IEEE*, pp.1–2, IEEE (2010).

[13] Nilsson, D.K. and Larson, U.E.: Conducting forensic investigations of cyber attacks on automobile in-vehicle networks, *Proc. 1st International Conference on Forensic Applications and Techniques in Telecommunications, Information, and Multimedia and Workshop*, p.8, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering) (2008).

[14] Mansor, H., Markantonakis, K. and Akram, R.N.: Log Your Car: The Non-invasive Vehicle Forensics, *The 13th IEEE ISPA 2015*, pp.1–8 (2016).

[15] 大月勇人, 瀧本栄二, 齋藤彰一, 毛利公一：マルウェア観測のための仮想計算機モニタを用いたシステムコールトレース手法，情報処理学会論文誌，Vol.55, No.9, pp.2034–2046 (2014).

[16] Cuckoo Foundation: Cuckoo Sandbox, available from ⟨https://cuckoosandbox.org⟩ (accessed 2017-12-01).

[17] Microsoft: Windows Sysinternals – Sysmon, available from ⟨https://docs.microsoft.com/en-us/sysinternals/downloads/sysmon⟩ (accessed 2017-12-01).

[18] NSA: SELinux, available from ⟨https://www.nsa.gov/what-we-do/research/selinux/⟩ (accessed 2018-09-08).

[19] 吉村豪康，橋本正樹，辻　秀典，田中英彦：TOMOYO Linux を用いた Linux マルウェアの動的解析環境構築に向けた初期的検討，コンピュータセキュリティシンポジウム 2016 論文集，pp.518–525 (Oct. 2016).

[20] 竹久達也, 牧田大佑, 神宮真人, 丑丸逸人, 福森大喜, 津田侑, 遠峰隆史, 井上大介：デバイスドライバを用いたプロセス挙動保全ツールの提案，コンピュータセキュリティシンポジウム 2017 論文集，pp.480–486 (Oct. 2017).

[21] トレンドマイクロ：Mirai, available from ⟨http://blog.trendmicro.co.jp/?s=mirai⟩（参照 2018-05-30）.

[22] Github: Mirai-Source-Code, available from ⟨https://github.com/jgamblin/Mirai-Source-Code⟩ (accessed 2017-11-01).

[23] 警察庁：宛先ポート 80/TCP に対する Mirai ボットの特徴を有するアクセスの増加について，available from ⟨http://www.npa.go.jp/cyberpolice/important/2018/201806131.html⟩（参照 2018-06-16）.

[24] Branco, R.R., Barbosa, G.N. and Neto, P.D.: Scientific but not academical overview of malware anti-debugging, anti-disassembly and anti-vm technologies, *Black Hat USA 2012* (Aug. 2012).

[25] Schallner, M.: Beginners guide to basic linux anti anti debugging techniques, *Code-Break Magazine, Security & Anti-Security – Attack & Defense* (2006).

[26] Bluetooth: Bluetooth Profile A2DP, available from ⟨https://www.bluetooth.org/docman/handlers/DownloadDoc.ashx?doc_id=260859&vId=290074⟩ (accessed 2018-09-28).

[27] Bluetooth: Bluetooth Profile HFP, available from ⟨https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=292287⟩ (accessed 2018-09-28).

[28] 大平修慈，金森健人，井上博之，石田賢治：車載 LAN トラフィック監視システム向けのリアルタイムデータ圧縮方式，コンピュータセキュリティシンポジウム 2017 論文集，pp.1469–1474 (Oct. 2017).

大平 修慈

2018 年広島市立大学情報科学部情報工学科卒業．2018 年より奈良先端科学技術大学院大学先端科学技術研究科博士前期課程，自動車におけるマルウェア対策や車載ネットワークの IDS・IPS の研究開発に従事．

井上 博之 （正会員）

1987 年大阪大学工学部電子工学科卒業．1989 年大阪大学大学院工学研究科電子工学専攻修了．1989～2000 年住友電気工業株式会社．1998 年奈良先端科学技術大学院大学情報科学研究科博士後期課程単位取得退学，博士（工学）．2000～2007 年株式会社インターネット総合研究所および株式会社 IRI ユビテック．2007 年より広島市立大学大学院情報科学研究科，現在は同大学准教授．広域ネットワークにつながる家電や自動車の情報セキュリティについて，その脆弱性やセキュアな通信プロトコルに関する研究開発に従事．電子情報通信学会，IEEE 各会員．

新井 イスマイル （正会員）

2002 年明石工業高等専門学校専攻科機械・電子システム工学専攻修了．2008 年奈良先端科学技術大学院情報科学研究科博士課程修了．博士（工学）．2008 年立命館大学ポスドク研究員，2011 年明石工業高等専門学校助教等を経て，2016 年より奈良先端科学技術大学院大学総合情報基盤センター准教授．屋内測位システム，センサネットワーク，データ解析等，ユビキタスコンピューティングの研究開発に従事．電子情報通信学会，IEEE，ACM 各会員．

藤川 和利 （正会員）

1988 年大阪大学基礎工学部情報工学科卒業．1991 年同大学大学院基礎工学研究科博士後期課程退学後，同年大阪大学基礎工学部助手等を経て，2002 年奈良先端科学技術大学院大学情報科学センター助教授，2005 年同大学大学院情報科学研究科助教授，2011 年同大学大学院情報科学研究科教授，現在に至る．博士（工学）．分散処理システム，マルチメディアシステム，ユビキタスコンピューティングの研究開発に従事．電子情報通信学会，IEEE，ACM 各会員．