車載インフォテインメントシステムにおけるホワイトリストと遅延付加によるCANバス上のDoS攻撃緩和手法

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

概要：コネクテッドカーのIVIがMiraiのようなアプリケーションレベルのマルウェアに感染し，C&Cサーバによって IVI から CAN バスおよび ECU に DoS 攻撃が発せられる問題がある．防御手法として，ある ECU が受信する CAN ID での DoS 攻撃に対し，CAN ID を動的に変更し受信 ECU の誤動作を防ぐ手法が提案されているが，DoS 攻撃は継続されるため，最高優先度の CAN ID パケットの大量送信を行う Traditional DoS 攻撃とランダムな CAN ID の大量送信を行う Random DoS 攻撃が防げない．そこで，本論文はマルウェアに感染した送信元に DoS パケット送信を抑止させる手法を提案する．ホワイトリスト内の CAN ID のパケットはそのまま送信し，そうでないパケットには遅延を付加することで，診断パケットの送信や使用する CAN ID を増やすといった拡張を行なった場合でさえもホワイトリスト外となる正常パケットを遮断させず，大量の DoS パケットを抑制できる．評価実験より，提案方式が帯域占有率 70%の DoS 攻撃を通常トラフィックと同様の約 20%まで緩和できることを確認した．また，提案方式によるオーバーヘッドは約 100μs と，CAN の最小送信周期以下であることから，車載器で動作可能なパフォーマンスであることを確認した．

キーワード：車載ネットワーク，DoS 攻撃，CAN，デバイスドライバ

DoS Attack Mitigation Method on CAN Bus by Whitelist and Delay Addition in In-vehicle Infotainment System

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

Abstract: There is a problem that a malware infect IVI of a connected car, then a C&C server issues a DoS attack from the IVI to CAN bus and ECU. Though the measure of the problem which is dynamically hopping CAN ID method for avoiding Targeted DoS attacks has proposed, this method cannot avoid Traditional DoS attacks and Random DoS attacks. Therefore, this paper proposes a method for suppressing DoS packets transmission of the source node. Packets with CAN ID in the whitelist is transmitted as is, then the others are added delay to mitigate the DoS attacks. This mitigation method is better than blacklisting when the packet not in the whitelist is a diagnostic message or future enhanced system use a new CAN ID. As a results of experiments, we confirmed that the band occupancy ratio got down from 70% to 20%, in other words, the proposed method can mitigate the DoS attacks. In addition, since the overhead of the proposed method is about 100 μs, which is less than the minimum transmission cycle of CAN, we confirmed that it works a vehicle-mounted device.

Keywords: In-vehicle Network, DoS Attack, CAN, Device Driver

[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
[2] 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
[3] 広島市立大学大学院 情報科学研究科
Graduate School of Information Sciences, Hiroshima City University

1. はじめに

コネクテッドカーの車載インフォテインメントシステム（以下，IVI; In-Vehicle Infotainment system）が Mirai [1] のようなアプリケーションレベルのマルウェアに感染し，C&C サーバによって IVI から CAN バスおよび ECU に DoS 攻撃が発せられる問題がある [2], [3]．車載ネットワークへの大量のパケットを送信する DoS 攻撃によって，ステアリング機能の異常を誘発，メーターや各種警告ランプの誤作動，自動運転機能の誤作動等が可能であり，これらの対策が急務となっている．しかしながら，従来提案されてきた方式の多くは，データとして得られた車載ネットワークトラフィックにおいて DoS 攻撃を検出する方式であり，先に挙げた異常や誤作動を直ちに防ぐことはできない．したがって，DoS 攻撃を検出し，防御する機構が必要となってくる．DoS 攻撃の防御手法として，ある ECU が受信する CAN ID での DoS 攻撃に対し，CAN ID を動的に変更し受信 ECU の誤動作を防ぐ手法 [4] が提案されているが，DoS 攻撃は継続されるため，最高優先度の CAN ID パケットの大量送信を行う Traditional DoS 攻撃とランダムな CAN ID の大量送信を行う Random DoS 攻撃が防げない．そこで，本論文はマルウェアに感染した送信元に DoS パケット送信を抑止させる手法を提案する．本研究では，アプリケーションレベルのマルウェアである Mirai のようなボットが IVI に感染することを想定し，マルウェアといった攻撃者に妨害されにくい機構を目標とする．提案方式では，ホワイトリスト内の CAN ID のパケットはそのまま送信し，そうでないパケットには遅延を付加することで，診断パケットの送信や使用する CAN ID を増やすといった拡張を行なった場合でさえもホワイトリスト外となる正常パケットを遮断させず，大量の DoS パケットを抑制できる．評価実験によって，提案方式を組み込んだマシンが実車トラフィックが正常に送信できることと車載ネットワークへの DoS 攻撃が緩和されていることを確認する．また，提案方式が攻撃者によって妨害されにくい機構であることを耐解析手法に耐性があることにより示す．さらに，車載器によって動作可能なパフォーマンスであることを実験的に示す．

2. 車載ネットワークにおける脅威と対策

2.1 Controller Area Network

CAN（Controller Are Network）[5] は自動車に搭載されている主要な車載ネットワークプロトコルであり，工場や医療現場などでも活用されている．CAN が開発された背景には，自動車の高機能化によって増加し，配線が複雑化していた自動車内部のコンピュータである ECU（Electronic Control Unit）を少ないハーネスで相互に接続して，安全性・経済性・利便性を高める目的がある．CAN によって自動車内部の配線等の問題は改善されたが，一方で，CAN の情報セキュリティ上の問題が指摘されている．CAN 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．この性質から，送信先情報の CAN ID と呼ばれる識別子の値が小さい ID ほど優先度が高い ID として送信される．また，CAN には，送信先情報の CAN ID のみで送信元情報を持たない．これらの性質から，CAN における ID の優先度を悪用した DoS 攻撃や，送信先情報を使ったなりすまし攻撃が問題となっている．さらに，帯域最大 1Mbps と Ethernet 等の LAN と比べると低速であり，大量パケット送信を行う DoS 攻撃を比較的性能の低いマシンで実現可能である．

2.2 DoS 攻撃の分類

本論文では，3 つのクラスの DoS 攻撃を定義する．

(1) **Traditional DoS 攻撃**

CAN における ID の優先度を悪用し，攻撃者は容易に CAN バスを占有することができる．例えば，CAN において最も優先度が高い CAN ID 0x000 と 0 が 8byte 続くペイロードを連続送信することで CAN バスを占有可能となる．このような種類の DoS 攻撃を Traditional DoS 攻撃と定義する．Traditional DoS 攻撃を対策するためには，攻撃検知後，攻撃者の送信を緩和や停止することが考えられる．しかし，車載ネットワークトラフィックがデータとして得られるアプリケーション層やデータリンク層といったレイヤでの対策ではバス上に攻撃が送信されてしまうため，物理的なレイヤと密接に関わるデバイスドライバなどで対策する必要がある．

(2) **Random DoS 攻撃**

ランダムな CAN ID やペイロードを用いた DoS 攻撃は特定の ECU を対象としておらず，Traditional DoS 攻撃と同様に，CAN バスの占有によって ECU の通常の動作を妨害する．このような種類の DoS 攻撃を Random DoS 攻撃と定義する．また，Targeted DoS 攻撃と切り分けるため，バス上の CAN ID は使用されないことを前提とする．Traditional DoS 攻撃と同様に，プリケーション層やデータリンク層といったレイヤでの対策ではバス上に攻撃が送信されてしまうため，物理的なレイヤと密接に関わるデバイスドライバなどで対策する必要がある．

(3) **Targeted DoS 攻撃**

特定の ECU への DoS 攻撃として，特定の ECU が受信する CAN ID を用いて DoS 攻撃を行う Targeted DoS 攻撃がある．例えば，エアバッグシステムの起動のトリガーとなる ECU へ大量のパケットを送信しエアバッグシステムの発生を遅らせることが可能で

ある．先行研究である ID-Hopping Mechanism [4] では，この Targeted DoS 攻撃の対策を ECU 間で使用する CAN ID を動的に変更することによって実現している．

2.3 関連研究

本節では，車載ネットワークにおける IDS・IPS，メッセージ認証方式，DoS 攻撃対策に関する関連研究について述べる．それぞれの関連研究において，3 つのクラスの DoS 攻撃の検出と防止が可能という要件を満たすかどうかを議論する．

2.3.1 IDS・IPS

車載ネットワークにおける IDS（Intrution Detection System）として，CAN における送信周期を用いた教師あり・教師なしでの検知方式 [6] や，物理的な特徴から送信元ノードを特定する方式 [7], [8] が提案されている．これらの手法は，3 つのクラス全ての DoS 攻撃を検知可能だが，検知後 3 つのクラスの DoS 攻撃に対してどのように対応するか考慮しなければならない．車載ネットワークにおける IPS（Intrution Prevention System）として，エラーフレームを用いた正規の送信 ECU での不正送信阻止機構 [9] が提案されているが，なりすまし攻撃に対する IPS であり，バスを占有する Traditional DoS 攻撃と Random DoS 攻撃は考慮されていない．

2.3.2 メッセージ認証

CAN におけるメッセージ認証方式がいくつか提案されている．LiBra-CAN [10] では，CAN の次世代規格 CAN-FD で動作可能であり，CAN では帯域の問題から正常に動作できないという課題がある．ハッシュチェーンに基づく Source Authentication Protocol [11] は，なりすまし攻撃やリプレイ攻撃に対しては有効だが，帯域を占有する攻撃に対しては有効ではない．また，CaCAN [12] は集中管理ノードで CAN バスを監視するため，3 つの DoS 攻撃全ての検知は可能である．しかしながら，エラーフレームによる DoS 攻撃の送信阻止を行うことでよりトラフィックが増大してしまうという問題がある．

2.3.3 DoS 攻撃対策の先行研究

DoS 攻撃対策の先行研究として，ID-Hopping Mechanism [4] が提案されている．ID-Hopping Mechanism はセントラルゲートウェイのように車載ネットワークに組み込まれ，Targeted DoS 攻撃を検知後，CAN バス上で使用される CAN ID を全てホップさせる方式である．CAN バス上で使用される全ての CAN ID をオフセットを用いてホップさせるため，CAN バス上で使用される CAN ID の優先順位の関係を保ったまま Targeted DoS 攻撃下であっても通信が可能となる．しかし，ID-Hopping Mechanism は，Targeted DoS 攻撃のみの対策手法であり，CAN バスの帯域増加を許してしまうため，Traditional DoS 攻撃と Random DoS 攻撃に対してこの手法は有効ではない．

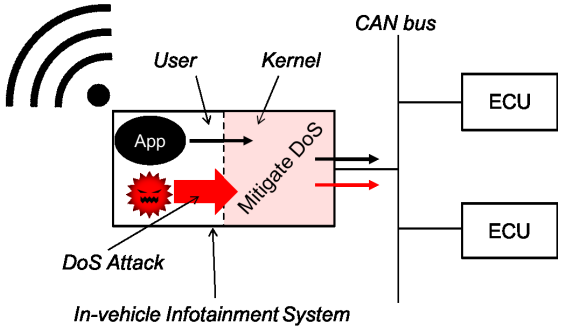

図 1　提案方式のアーキテクチャ

そのため，提案方式は，図 1 のように，攻撃者が踏み台とすると考えられる IVI のような外部ネットワークと車載ネットワークのゲートウェイとなる機器に組み込むことを想定し，Traditional DoS 攻撃と Random DoS 攻撃を緩和することを目的とする．また，提案方式を攻撃者が踏み台とする機器に組み込むため，攻撃者によって妨害されにくい機構であることを耐解析手法に耐性がある必要がある．さらに，車載器によって動作可能なパフォーマンスでなければならない．

3. 提案方式

3.1 提案方式の要件

本節では，提案方式の要件について述べる．2 章での議論より，以下に要件をまとめた．

I. **Traditional DoS 攻撃，Random DoS 攻撃を緩和可能**

先行研究の ID-Hopping Mechanism[4] では，Targeted DoS 攻撃に対してのみ有効であった．そこで，本研究では，Traditional DoS 攻撃，Random DoS 攻撃を緩和可能な方式を要件とし，ID-Hopping Mechanism[4] と提案方式を組み合わせて，車載ネットワーク上の全ての DoS 攻撃を対策可能とすることを目指す．Traditional DoS 攻撃では CAN ID 0x000 が使用されるため，0x000 に対して送信を遅延させることが考えられる．しかしながら，Random DoS 攻撃では，使用される CAN ID が不規則なためブラックリスト的に送信を遅延させることは難しい．そこで，提案方式を組み込む ECU の送信する CAN ID をホワイトリストとして保持しておき，ホワイトリスト内の CAN ID には緩和する処理を実行する．これにより，Traditional DoS 攻撃，Random DoS 攻撃を緩和可能という要件を満たす．

II. **攻撃者に妨害されにくい機構**

提案する方式は，外部接続点となる IVI に組み込まれるため，攻撃者から妨害されにくい設計が必要不

可欠となる．方式として，送信を監視するプロセスによって送信間隔の緩和を適用する機構が考えられる．しかし，プロセスによって緩和を適用する機構では攻撃者と同様のユーザ空間で動作するため，攻撃者から提案方式を検出・妨害される恐れがある，これより，提案方式をユーザ空間の攻撃者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，攻撃者から提案方式の検出・妨害を低減し，要件を満たす．

III. **通常パケット送信における許容可能なオーバーヘッド**
提案する方式は，実際の自動車での運用を考えると，車載器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，提案方式を車載器として見立てたRaspberry Pi に実装し，提案方式の有無による CAN パケット送信のオーバーヘッドを計測する．また，DoS 攻撃下において，提案方式による遅延付加がある場合であっても，CAN パケットが送信可能であることを確認する．

3.2 提案方式における緩和アルゴリズム

本節では，提案方式における Traditional DoS 攻撃，Random DoS 攻撃の緩和アルゴリズムについて述べる．緩和アルゴリズムを Algorithm 1 に示す．緩和アルゴリズムでは，送信周期を用いて DoS 攻撃を判定するため，現在の時刻 CurrentTime を保持し，以前の時刻 PrevTime との差 TransmitCycle を算出する．TransmitCycle が，送信周期となる．そして，ユーザプロセスから与えられた CAN パケットの CAN ID がホワイトリストに含まれるならば，直ちに送信を行う．そうでなければ，送信周期 TransmitCycle が閾値 DoS Threshold より小さい値であれば，DoS 攻撃であると判定し，遅延を挿入する．最後に，送信した後，PrevTime を更新し，次の送信を待つ．以上が提案する防御機構における緩和アルゴリズムとなる．

以降では，閾値 DoS Threshold の算出方法について述べる．DoS Threshold は，式（1）で算出される．式（1）の AvgNormalCycle と AvgDoSCycle は，それぞれ通常トラフィック送信時の平均送信周期と DoS 攻撃送信時の平均送信周期である．今回用いた通常トラフィックは CAN バス全体の ECU が送信するトラフィックであり，単一の ECU が送信するトラフィックではより大きな送信周期となることから，CAN バス全体の ECU が送信するトラフィックで DoS 攻撃判定可能ならば，単一の ECU が送信するトラフィックでも DoS 攻撃判定可能と言える．

$$\text{DoS Threshold} = \frac{AvgNormalCycle + AvgDoSCycle}{2} \quad (1)$$

3.3 提案方式の実装

本節では，提案方式の実装について述べる．提案方式は

Algorithm 1 Mitigation Algorithm in Proposal Method

```
Ensure: Mitigating Traditonal, Random DoS Attack
  CANpacket ⇐ TxData
  CurrentTime ⇐ GetTime
  TransmitCycle ⇐ CurrentTime − PrevTime
  if CANpacket.CANID in White List then
    Transmit(CANpacket)
  else
    if TransmitCycle < DoS Threshold then
      Delay(5ms)
    end if
    Transmit(CANpacket)
  end if
  PrevTime ⇐ CurrentTime
```

Linux カーネルの CAN デバイスドライバである mcp251x.c [13] に組み込まれる．mcp251.c における CAN パケット送信時のハンドラとなる関数 mcp251x_tx_work_handler に，Algorithm 1 を実装し，そのカーネルモジュールを Raspberry Pi にインストールした．また，DoS 攻撃パケットによる送信キューの溢れが原因となる通常パケットの送信不可を防ぐために，CAN インターフェースの txqueuelen を 10000 とし，パケット送信キューの溢れを防止した．

4. 評価

4.1 DoS 攻撃時における CAN の帯域による評価

提案方式の緩和性能を評価するために，提案方式の有無による DoS 攻撃時における CAN の帯域を比較する．また，通常の実車トラフィック再生時において提案方式の有無によって CAN の帯域に変化がないことも確認する．まず，実験環境について述べる．実験に使用した装置を図 2 に示す．Raspberry Pi は，組み込み Linux として知られており，車載器においても組み込み Linux [14] が用いられていることから，提案方式をテストする妥当な環境であると言える．

図 3 に実車トラフィック再生時における帯域占有率を示す．図 3 より，提案方式有の場合でも，提案方式無の場合と同様に実車トラフィックは正常に送信されていることが確認できた．次に，図 4 に Traditional DoS 攻撃時の帯域占有率を示す．図 4 より，提案方式無の場合は Traditional DoS 攻撃によって帯域が 70%程度占有されているが，提案方式有の場合は帯域が 20%前後で抑えられていることが確認できた．また，帯域が 70%程度しか占有されていないのは，実験環境の CPU 性能が原因である．最後に，図 5 に Random DoS 攻撃時の帯域占有率を示す．Traditional DoS 攻撃の結果と同様に，提案方式無の場合は Random DoS 攻撃によって帯域が 70%程度占有されているが，提案方式有の場合は帯域が 20%前後で抑えられていることが確認できた．

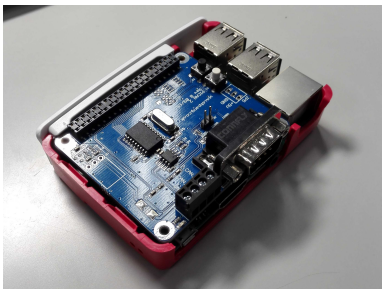

図 2　実験に使用した装置 (Raspberry Pi 3 + PiCAN 2)

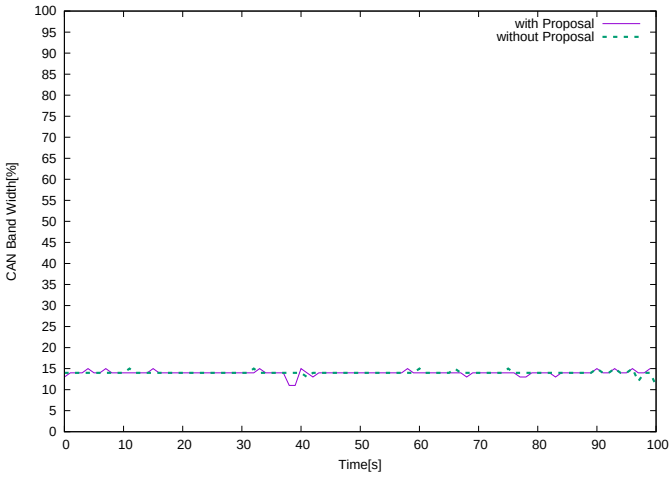

図 3　実車トラフィック再生時の帯域占有率

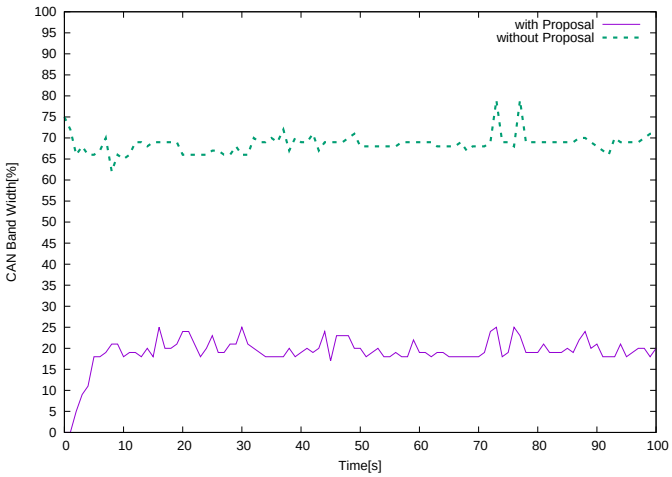

図 4　Traditional DoS 攻撃時の帯域占有率

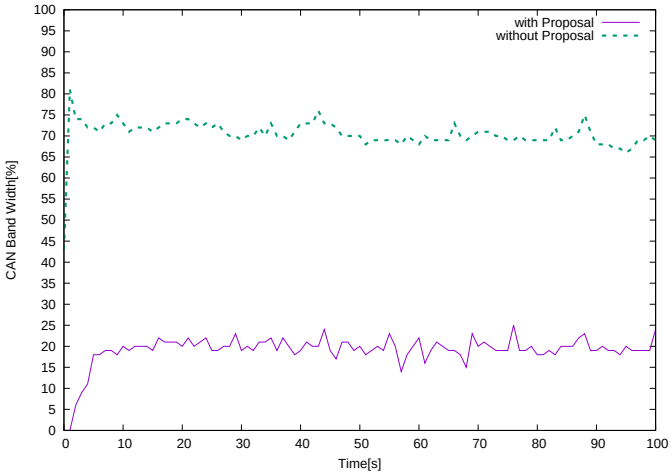

図 5　Random DoS 攻撃時の帯域占有率

表 1　対解析手法に対する定性評価結果

対解析手法		回避可能
GDB	ptrace	○
による	readlink	○
検出手法	breakpoint	○

4.2　耐解析手法に対する評価

本節では，アンチデバッグを始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，Linux の標準的なデバッガである GDB の検出手法 [15] に基づいている．GDB による検出手法を用いて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 に提案方式の耐解析手法に対する定性評価結果を示す．

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デバイスドライバで対策を行うため 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 などが実行されているか検出できる．ptrace の場合と同様に，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言える．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 DoS 攻撃を検出する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言える．

4.3　パフォーマンス評価

提案方式は，実際の自動車での運用を考えると，車載器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，提案方式を車載器として見立てた Raspberry Pi に実装し，提案方式の有無による実車トラフィック送信におけるオーバーヘッドを計測する．また，DoS 攻撃下において提案方式による遅延付加がある場合であっても，実車トラフィック送信可能であることを確認する．通常の CAN バスにおけるオーバーヘッドの計測として，提案方式を組み込んだ Raspberry Pi で CAN パケットの送信を 100 回呼び

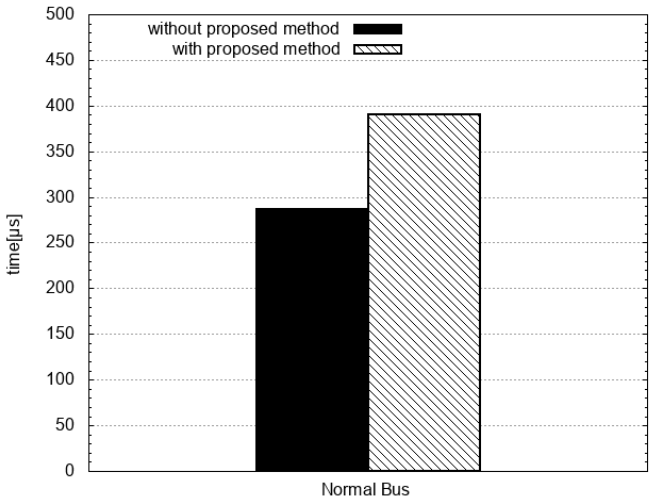

図 6 実車トラフィックの送信におけるオーバーヘッド

出す操作を 100 回施行し，その送信時間の平均値を算出した．提案方式の有無による実車トラフィックの送信のオーバーヘッドを図 6 に示す．図 6 より，実車トラフィックの送信におけるオーバーヘッドは，104μs であった．これより，オーバーヘッドは最大帯域 1Mbps の CAN の最小送信周期の 111μs よりも小さいという結果が得られ，許容可能な値だと言える．次に，DoS 攻撃下において提案方式による遅延付加がある場合であっても，実車トラフィックが送信可能であることを確認する．DoS 攻撃下において提案方式を組み込んだ Raspberry Pi で CAN パケットの送信を 100 回呼び出す操作を 100 回施行し，全ての CAN パケットが送信されることを確認した．これより，DoS 攻撃下において，提案方式による遅延付加がある場合であっても，CAN パケットが送信可能であると言える．

5. おわりに

本論文は，コネクテッドカーの IVI が Mirai のようなアプリケーションレベルのマルウェアに感染することを想定し，マルウェアに感染した送信元で DoS パケット送信を抑止を行う手法の提案を行った．提案方式は，ホワイトリスト内の CAN ID のパケットはそのまま送信し，そうでないパケットには遅延を付加することで，後に機能拡張してホワイトリスト外となる正常パケットを遮断させず，大量の DoS パケットを抑制できる．Linux カーネルの CAN デバイスドライバに提案方式を実装し，評価実験を行ったところ，提案方式が帯域占有率 70%の DoS 攻撃を通常トラフィックと同様の約 20%まで緩和できることを確認した．また，提案方式が攻撃者によって妨害されにくい機構であることを耐解析手法に耐性があることを用いて示した．さらに，提案方式によるオーバーヘッドが約 100μs であり，CAN の最小送信周期以下であることから，車載器で動作可能なパフォーマンスであることを確認した．今後の課題として，ペイロードの振る舞いや正規の送信周期を用いた Targeted DoS 攻撃も含めた全ての DoS 攻撃を緩和可能な手法を検討していく．

参考文献

[1] Jerry Gamblin, “Mirai-Source-Code,” https://github.com/jgamblin/Mirai-Source-Code, 参照, Apr. 1, 2018.

[2] C. Miller, and C. Valasek, “Remote Exploitation of an Unaltered Passenger Vehicle,” DEFCON23, pp.1-91, Aug. 2015.

[3] Takaya Ezaki, Tomohiro Date, and Hiroyuki Inoue, “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.

[4] Abdulmalik Humayed, Bo Luo, ”Using ID-Hopping to Defend Against Targeted DoS on CAN,” Proceedings of the 1st International Workshop on Safe Control of Connected and Autonomous Vehicles, pp.19-26, Apr. 2017.

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

[6] Takuya Kuwahara, Yukino Baba, Hisashi Kashima, Takeshi Kishikawa, Junichi Tsurumi, Tomoyuki Haga, Yoshihiro Ujiie, Takamitsu Sasaki, Hideki Matsushima. Supervised and Unsupervised Intrusion Detection Based on CAN Message Frequencies for In-Vehicle Network. Journal of Information Processing, 2018.

[7] Cho, K.T., Shin, K.G, ”Fingerprinting Electronic Control Units for Vehicle Intrusion Detection,” 25th USENIX Security Symposium (USENIX Security 16), USENIX Association (2016) 911–927

[8] Cho, K.T., Shin, K.G, ”Viden: Attacker Identification on In-Vehicle Networks,” Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. CCS ’ 17, ACM (2017) 1109–1123

[9] 畑 正人, 田邉 正人, 吉岡 克成, 松本 勉, ”CAN における不正送信阻止方式の実装と評価,” 電子情報通信学会技術研究報告, vol.112, no.342, pp15-22, Dec. 2012.

[10] Bogdan Groza, Stefan Murvay, Anthony Van Herrewege, and Ingrid Verbauwhede. 2017. LiBrA-CAN: Lightweight broadcast authentication for controller area networks. ACM Trans. Embed. Comput. Syst. 16, 3, Article 90 (April 2017), 28 pages.

[11] Ki-Dong Kang, Youngmi Baek, Seonghun Lee, Sang Hyuk Son, ”An Attack-Resilient Source Authentication Protocol in Controller Area Network,” 2017 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), pp.1-10, May. 2017.

[12] 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.

[13] Linux kernel, ”mcp251x.c,” https://github.com/torvalds/linux/blob/master/drivers/net/can/spi/mcp251x.c, 参照 July. 17, 2018.

[14] The Linux Foundation, ”Automotive Grade Linux,” https://www.automotivelinux.org/, 参照, Apr. 1, 2018.

[15] M. Schallner “Beginners guide to basic linux anti anti debugging techniques,” Code-Break Magazine, Security & Anti-Security - Attack & Defense, 2006.