Protocol-Compliant DoS Attacks on CAN: Demonstration and Mitigation

Wei Si*, David Starobinski*, and Moshe Laifenfeld†

*Dept. of Electrical and Computer Engineering, Boston University, USA
{weisi, staro}@bu.edu
†moshel@spacegate.com

***Abstract*—The Controller Area Network (CAN) is a shared medium, priority-based communication protocol, widely used in the automotive industry for interconnecting electrical components. Although allowing messages to take priority over others in accessing the shared medium is naturally desirable for vehicular applications, it also provides a vulnerability for Denial-of-Service (DoS) attacks. This paper studies the impact of such priority-based DoS attacks and proposes a mitigating scheme. We find that implementation details have a significant impact on the efficiency of priority-based DoS attacks. Nevertheless, with a proper configuration, a single attacker can block an entire CAN network and deem it unusable. To mitigate this problem, we propose integrating a wireless interface and design a hybrid wired/wireless protocol that schedules packet transmissions on the wired and wireless links. Our testbed results show that the hybrid wired/wireless protocol improves the throughput under a two-node DoS attack by a factor of four. Additional experimental results demonstrate that our hybrid wired/wireless protocol is robust to jamming attacks on the wireless link.**

I. Introduction

The Controller Area Network (CAN [1]) is a shared-medium, priority-based protocol widely used for interconnecting electrical components such as sensors, actuators and controllers in modern vehicles. The CAN protocol is designed in a priority-based manner such that a high-priority message gets access to the shared medium before a low-priority message. Naturally, priority is desirable for vehicular applications. For example, airbag system sensors, which are critical for the safety of passengers, should have higher priority than climate control sensors.

Medium access priority is implemented by using a message identifier field. Since the dominant bit (logical zero) overrides the recessive bit (logical one) on the shared medium (AND channel), the smaller value the identifier field assumes, the higher the priority of the message. During the arbitration process, transmitters monitor the shared medium during their recessive bit transmissions and if they detect an overriding dominant bit, they infer that higher priority message is being transmitted and therefore cease their own transmission.

Although priority-based communication is highly desirable for an intra-vehicular network, it also provides a vulnerability for Denial-of-Service (DoS) attacks. A straightforward way is to transmit high-priority messages with small identifiers [2]. Recent research shows that intra-vehicular components can be compromised remotely [3] and therefore can be hijacked and used to inject high-priority messages. This protocol-compliant DoS attack is harder to detect and is the focus of our work.

The potential of such protocol-compliant attacks leads to several questions. For instance, can one attacker cause all legitimate nodes to be unable to communicate over the CAN bus? If not, how many attackers are needed to take down the whole CAN system?

In this work, we aim to answer these questions by studying the effects of protocol-compliant DoS attacks on the CAN communication system. We implement a testbed using CAN transceivers and perform tests of the protocol-compliant DoS attacks on the testbed. It turns out that when the capacity of the MAC-layer queue is one, a single attacker cannot take down the CAN system. More specifically, no matter the rate at which high-priority messages are generated, legitimate nodes can still share half of the CAN bandwidth. We show that if one adds one more attacker, then the two attackers are able to bring down the throughput of legitimate nodes to less than 2% of the bus bandwidth. Moreover, if the implementation of the MAC-layer queue is modified, then a single DoS attacker can achieve the same efficiency as two attackers.

Next, we consider a solution. Since wireless communication links already exist in vehicles (e.g., the tire pressure monitoring system (TPMS) and the backup camera system), we propose utilizing the wireless links to mitigate the DoS attacks. We propose a hybrid wired/wireless protocol to protect the legitimate nodes from protocol-compliant DoS attacks. The hybrid wired/wireless protocol is backward compatible with the CAN protocol since no modification of the CAN protocol is needed. The hybrid wired/wireless protocol monitors the link quality on the CAN bus. When it detects that the CAN link quality is bad, it schedules packet transmissions on the wireless link. We implement the hybrid testbed using the aforementioned CAN transceivers and ZigBee transceivers. Empirical results from the testbed show that the hybrid wired/wireless protocol can increase the throughput by a factor of four. We stress that the hybrid protocol cannot only help in providing resilience against DoS attacks, but also help in relieving congestion from the CAN bus under normal operations. We also test and demonstrate the resilience of the hybrid protocol to wireless jamming attacks [4].

We summarize the contributions of this paper as follows:

- We implement a testbed using CAN transceivers and ZigBee transceivers to evaluate the impact of protocol-compliant DoS attacks on the CAN bus;
- We discover that the MAC-layer queue capacity has a significant impact on the efficiency of the DoS attack;
- We design and implement a hybrid wired/wireless protocol to protect against DoS attacks on the CAN bus as well as against wireless jamming attacks.

The rest of the paper is organized as follows. Section II reviews related work on the security of the CAN protocol. In Section III, we first review relevant specifications of the CAN protocol and demonstrate the effects of various DoS attacks. In Section IV, we describe our proposed hybrid wired/wireless protocol and thereafter, in Section V, we provide experimental results of the resiliency of the hybrid wired/wireless protocol under DoS attacks on the CAN bus and jamming attacks on the wireless link. Finally, in Section VI, we conclude the paper and discuss future directions.

II. Related work

The vulnerability analysis and assessment of in-vehicle communication system has been gaining a lot of attention recently. The authors in [2] show a comprehensive list of malicious attacks that can be carried out by spoofing CAN messages through the On-Board Diagnostics (OBD-II) port. For example, attackers can lock and unlock doors, and control lights, brakes and windshield wipers. Even the engine can be shut off when the car is driving. Although their experiments are performed by physically connecting a laptop to the OBD-II port through a CAN-to-USB cable, the injection of malicious CAN messages can be done by remotely compromising and controlling the radio, the tire pressure monitoring system or the Bluetooth component [3]. Some cars, which are connected to the Internet for purposes of entertainment and navigation, can be hacked and controlled from the Internet [5]. More examples of attacks on the vehicular system can be found in [6].

To address vulnerabilities of intra-vehicular CAN communication systems, research has been conducted on designing message authentication mechanisms. For instance, [7] proposes a backward-compatible broadcast authentication protocol for CAN bus. More work along the same lines can be found in [8–11].

While these works focus on the message authentication aspect of CAN system, little work has addressed *availability* issues. Different from previous works, our paper takes on this problem.

III. DoS attacks on CAN

In this section, we first review relevant details of the CAN protocol. Then, we describe our implementation of the DoS attack and explain how the MAC-layer queue capacity affects the attack efficiency. In the end, we provide experimental results from our testbed to demonstrate the effects of various attack implementations.

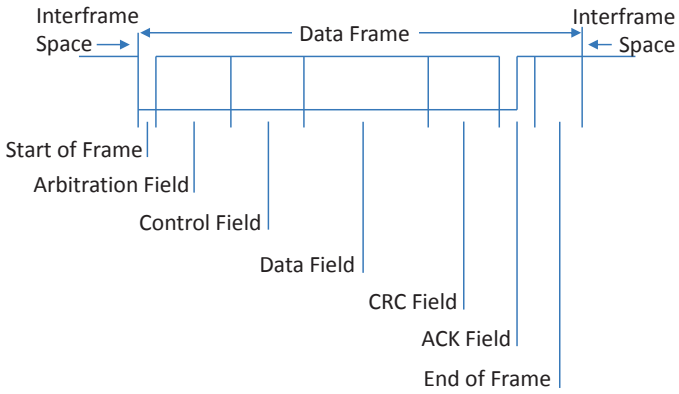


Fig. 1: The format of a CAN message. The identifier is contained in the arbitration field and determines the priority of the message.

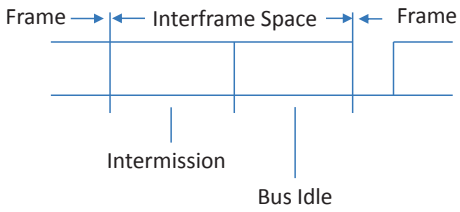


Fig. 2: The interframe after the data frame. After the intermission period, all nodes can start to transmit pending messages.

A. CAN basics

The CAN protocol was introduced by Robert Bosch GmbH [1] and has been widely used in modern car models as well as other industry applications. In a CAN network, nodes are connected to a CAN bus. The CAN bus has one of the two complementary logical values: *dominant* or *recessive*. During simultaneous transmission of dominant and recessive bits, the resulting bus value will be dominant. In a typical implementation, the dominant bit is represented by a logical zero and the recessive bit is represented by a logical one.

The MAC layer of CAN is based on Carrier Sensing Multiple Access with Collision Detection (CSMA/CD). Whenever the bus is free, any node can transmit a CAN message. During transmission, nodes monitor bits on the CAN bus. If a node has transmitted a recessive bit but monitors a dominant bit, then it detects the bus contention and withdraws without sending more bits.

The CAN protocol resolves bus contention based on the identifier field, which is contained in the arbitration field of the CAN message (see Fig. 1). A smaller identifier field overrides a larger identifier field. Thus, the priority of a message gets higher as the identifier gets smaller. When a high-priority node and a low-priority node transmit CAN messages simultaneously, the low-priority node detects that a high-priority message is being transmitted, stops transmission and defers its transmission to the next opportunity.

After transmission of a CAN message on the bus, the message transmitter and the message receivers (basically all

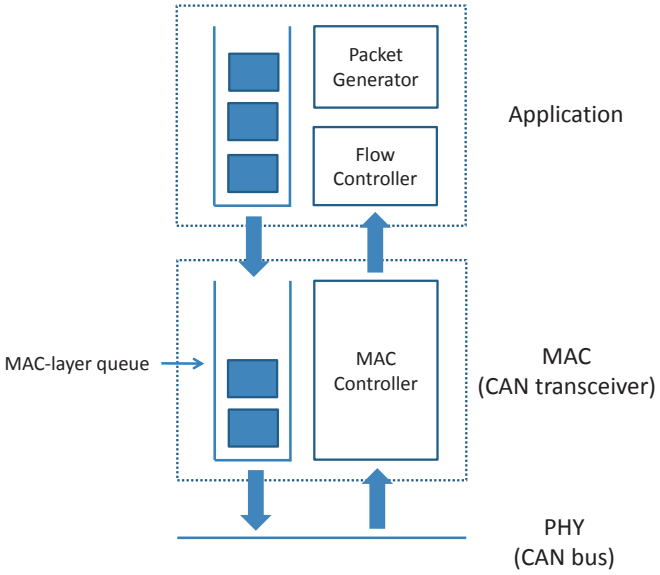


Fig. 3: Implementation of the DoS attack. High-priority messages flow from the application-layer queue to the MAC-layer queue. The flow controller controls the transferring process such that the number of packets in the MAC layer does not exceed the MAC-layer queue capacity.

other nodes on the bus since CAN is a broadcast medium) wait for a period called the *interframe space*, as shown in Fig. 2. At the beginning of the interframe space, there is an *intermission* period, which lasts for three bits. After the intermission period, all nodes can start to transmit pending messages.

B. Implementation of the DoS attack

We implement the priority-based DoS attack at the application layer as it generates and transfers packets to the CAN transceiver. The DoS attacker consists of a queue, a packet generator and a flow controller (see Fig. 3). The queue stores the high-priority messages that wait to be transferred to the MAC layer. The packet generator periodically generates packets and injects them into the queue if the queue is not full. The flow controller keeps count of the number of packets in the MAC-layer queue. When a packet is transferred to the MAC layer, the counter increments; when a notification of completed transmission is received, the counter decrements. The flow controller controls the transfer of packets, such that the counter does not exceed the maximum number of packets allowed in the MAC-layer queue, referred to as the *MAC-layer queue capacity*.

Typically, a packet sending application transfers a packet to the MAC layer and marks the packet transceiver as busy. When the ACK is received, the packet sending application marks the transceiver as idle and transfers another packet to the MAC layer. This is equivalent to a MAC-layer queue capacity set to one. In the CAN protocol, after the intermission period, all nodes with pending messages start to contend for the bus access. Suppose the attacker has a MAC-layer queue with capacity set to one and the transmission of a message is just completed. Then, the MAC-layer queue will have a new pending message only after the flow controller receives the notification of completed transmission and transfers a new packet. Although the gap may be very small, a low-priority legitimate node can start message transmission during that gap.

We can improve the efficiency of this DoS attack in two ways. The first way is to add a second DoS attacker transmitting high-priority messages. In this case, when one attacker finishes transmission and does not have a pending message available, the other attacker, which already has a pending message, can contend for the bus. The second way is to increase the capacity of the MAC-layer queue to contain at least two packets. As long as the attacker is generating packets at a sufficiently high rate, the MAC-layer queue will always have pending messages.

C. Evaluation metrics

We next describe the metrics used in our experiments. We refer to the *packet generation rate* as the rate at which a legitimate node generates low-priority messages. We define the *attacking rate* as the rate at which the attacker generates high-priority messages.

Suppose a test lasts for T seconds. Let N denote the total number of packets that a legitimate node generates in the test. The sink receives N_u packets from the legitimate node, excluding duplicates, and let $\mathcal{S}_d$ represent the set of the delivered packets.

The *delivery rate* of the legitimate node is defined as

$$\text{Delivery rate} = \frac{N_u}{N} \cdot 100\%.$$

The *throughput* of the legitimate node is defined as

$$\text{Throughput} = \frac{N_u}{T} \text{ pkts/sec}.$$

The delay of a packet D_i is defined as the time elapsing from its generation at the source node to its delivery at the sink. The *average delay* of packets from the legitimate node is calculated as follows:

$$\text{Average delay} = \frac{1}{N_u} \sum_{i \in \mathcal{S}_d} D_i.$$

D. Demonstration

To demonstrate the DoS attacks on the CAN bus, we implement a testbed, as shown in Fig. 4. In the testbed, we use the VN1610 CAN interface [12] by Vector Informatik GmbH as the CAN transceiver. A node is a Windows Presentation Foundation (WPF) application running on a PC. The testbed consists of three types of nodes: legitimate node (representing a sensor), DoS attacker and sink node (representing an ECU). A legitimate node periodically generates CAN messages and transfers them to the CAN transceiver for transmission on the CAN bus. An attacker periodically generates CAN messages with a smaller identifier than the legitimate node. The implementation of the legitimate node is similar to that of the attacker, except for the identifier. The sink receives the CAN messages from legitimate nodes and records the events of message receptions.

Each test is run for five times to obtain an average and a 95% confidence interval for the metrics. Each run lasts

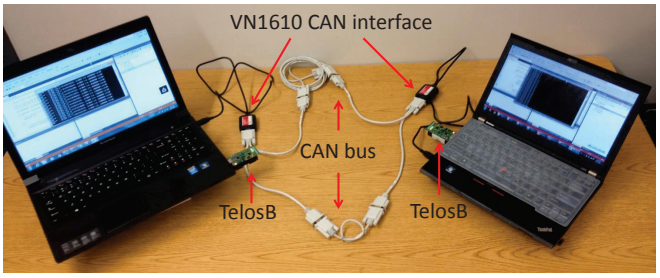
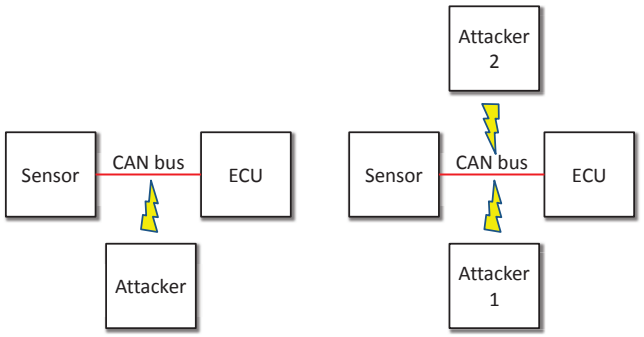


Fig. 4: The testbed for demonstration of DoS attacks on the CAN bus. In the network, a node is a WPF application, which can transmit and receive packets using the CAN and ZigBee transceivers.



(a) One attacker. (b) Two attackers.

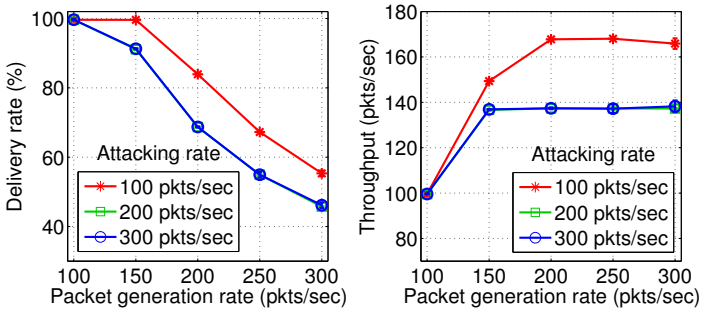
Fig. 5: Setup of DoS attacks on the CAN bus. The sensor (the legitimate node) transmits low-priority messages to the ECU (the sink) while the attacker(s) transmit high-priority messages on the CAN bus.

from three to five minutes. The baud rate of the CAN bus is configured to 33333 baud. A CAN message is 108 bits. Taking the intermission period (three bits) into consideration, the maximum throughput supported by the CAN bus is $33333 \text{ baud}/(108 \text{ bits} + 3 \text{ bits}) \approx 300$ pkts/sec.

The setup for testing a DoS attack of one attacker with MAC-layer queue capacity equal to one is shown in Fig. 5(a). The legitimate node generates CAN messages with identifier 0x49 while the attacker generates CAN messages with identifier 0x11. The delivery rate and throughput of the legitimate node are shown in Fig. 6. Fig. 6(a) shows that the delivery rates of the legitimate node under attacking rates of 200 pkts/sec and 300 pkts/sec are very close to each other. This indicates that when the attacking rate of the attacker is high enough, no more harm can be done on the CAN system by further increasing the attacking rate.

Fig. 6(b) shows that the throughput of the legitimate node achieves its maximum, around 137 pkts/sec, when its packet generation rate exceeds 200 pkts/sec. The maximum achievable throughputs under attacking rates of 200 pkts/sec and 300 pkts/sec, are similar, a little less than half of the CAN bandwidth. Thus, even if we further increase the attacking rate of the attacker, the legitimate node can still share almost half of the CAN bandwidth.

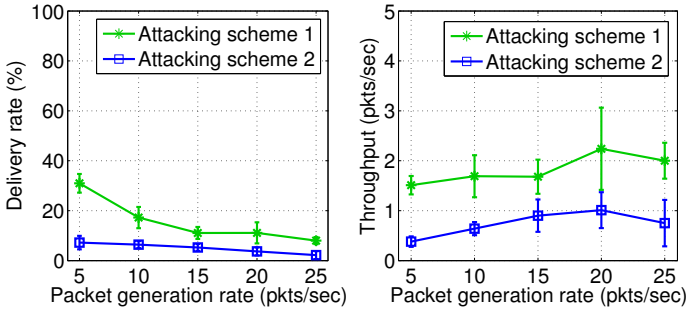
Next we demonstrate the effects of two other DoS attacking



(a) Delivery rate of the legitimate node versus its packet generation rate.

(b) Throughput of the legitimate node versus its packet generation rate.

Fig. 6: Experimental results of one attacker (with the MAC-layer queue capacity equal to one) attacking at different rates. The legitimate node can achieve the maximum throughput of around 137 pkts/sec under the attacking rate of 200 pkts/sec. Further increasing the attacking rate to 300 pkts/sec does not reduce the maximum achievable throughput of the legitimate node.



(a) Delivery rate of the legitimate node versus its packet generation rate.

(b) Throughput of the legitimate node versus its packet generation rate.

Fig. 7: Experimental results of attacking scheme 1 (two attackers with the MAC-layer queue capacity equal to one) and attacking scheme 2 (one attacker with the MAC-layer queue capacity equal to two), both attacking at a total rate of 300 pkts/sec. The throughput of the legitimate node under the two DoS attacking schemes are both under 4 pkts/sec (less than 2% of the bus bandwidth).

schemes: (1) two attackers with MAC-layer queue capacity equal to one, and (2) a single attacker with MAC-layer queue capacity equal to two. For the scenario of two attackers, as shown in Fig. 5(b), we add another attacker, which generates CAN messages with identifier 0x12. The attacking rates of the two attackers are 150 pkts/sec each, summing up to a total attacking rate of 300 pkts/sec. For the scenario of one attacker with MAC-layer queue capacity equal to two, the attacking rate is 300 pkts/sec.

The delivery rate and throughput of the legitimate node under the two DoS attacking schemes are shown in Fig. 7. For the case of two attackers, Fig. 7(a) shows that even when the legitimate node is generating packets at a rate as low as 5 pkts/sec, its

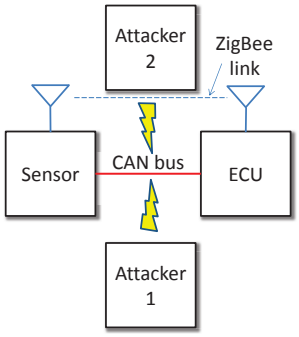


Fig. 8: Adding an independent ZigBee link to protect against DoS attacks on the CAN bus.

delivery rate is only 30.96%. This result stands in contrast to the case of one attacker with MAC-layer queue capacity equal to one, where the legitimate node can achieve a delivery rate of more than 99% at a packet generation rate of 100 pkts/sec. Fig. 7(b) shows that the DoS attacks of two attackers can throttle the throughput of the legitimate node to under 4 pkts/sec (less than 2% of the CAN bandwidth).

With a small change to the MAC-layer queue capacity, a single attacker can be even more efficient than two attackers. Thus, the throughput of the legitimate node is brought down to under 2 pkts/sec if the MAC-layer queue capacity of the attacker is set to two. Note that the low-priority legitimate node can still transmit a few packets to the sink. We conjecture that it is because our testbed is implemented using the Windows operating system, which is not real-time.

In summary, the results demonstrate that when the DoS attacker uses a MAC-layer queue capable of holding only one packet, it can at most take down half of the CAN bandwidth and a second attacker is needed to take down most of the rest of the bus bandwidth. However, with a small implementation change (i.e., increasing the capacity of the MAC-layer queue), a single DoS attacker can bring the throughput of the legitimate node down to a very low level.

IV. Hybrid wired/wireless protocol

To protect against DoS attacks on the CAN bus, we propose combining an independent wireless link (i.e., ZigBee) at safety-critical nodes and at the sink, thus establishing a hybrid wired/wireless network (see Fig. 8). For this hybrid network, we design and implement a protocol for scheduling transmission of packets on the wired and wireless links. Besides enhanced resiliency, this protocol can also help relieving congestion from the CAN bus under normal operations. In this section, we describe the design of the hybrid wired/wireless protocol.

Our hybrid wired/wireless protocol is designed based on the Backpressure Collection Protocol (BCP [13, 14]), which is used for data collection in multi-hop wireless sensor networks. BCP achieves high throughput and low average number of transmissions (ETX). Similar to BCP, our hybrid wired/wireless protocol makes use of ETX as a measure of the link quality of the CAN bus and of the ZigBee link. ETX is calculated by counting the number of retransmissions before receiving an ACK from the receiver. When a link is congested, the MAC delay increases. An ACK timeout at the transmitter triggers a retransmission. Therefore, ETX increases as the quality of the link gets worse.

Algorithm 1 Hybrid wired/wireless protocol

procedure CAN_INTERFACE_HANDLER
  CAN_busy ← false
  while $\mathcal{Q} > 0$ **do**
    Compute the weight w^{CAN}
    if $w^{CAN} > 0$ **and** (ZigBee_busy = true **or** $w^{CAN} \geq w^{ZigBee}$) **then**
      CAN_busy ← true
      Transmit one packet to the sink over the CAN interface
      Update $\overline{ETX}^{CAN}$ and $\overline{\mathcal{R}}^{CAN}$
      CAN_busy ← false
    else
      Wait for a retry period
    end if
  end while
end procedure

procedure ZIGBEE_INTERFACE_HANDLER
  ZigBee_busy ← false
  while $\mathcal{Q} > 0$ **do**
    Compute the weight w^{ZigBee}
    if $w^{ZigBee} > 0$ **and** (CAN_busy = true **or** $w^{ZigBee} \geq w^{CAN}$) **then**
      ZigBee_busy ← true
      Transmit one packet to the sink over the ZigBee interface
      Update $\overline{ETX}^{ZigBee}$ and $\overline{\mathcal{R}}^{ZigBee}$
      ZigBee_busy ← false
    else
      Wait for a retry period
    end if
  end while
end procedure

The hybrid wired/wireless protocol consists of two handlers handling the CAN interface and the ZigBee interface respectively. Next, we describe the handler of interface I, where $I \in \{CAN, ZigBee\}$. Let $\mathcal{Q}$ represent the backlog at a hybrid node. Let $\overline{\mathcal{R}}^I$ denote the estimated link rate from the hybrid node to the sink over interface I and let $\overline{ETX}^I$ be the average number of transmissions needed for a packet to be successfully sent over the interface. The interface handler of the hybrid node calculates the *weight* of interface I as follows:

$$w^I = (\mathcal{Q} - V \cdot \overline{ETX}^I) \cdot \overline{\mathcal{R}}^I.$$

The weight represents the quality of the link: a higher weight means a link of higher quality. A necessary condition for the interface handler to transmit a packet over interface I is that $w^I > 0$. When both interface handlers are idle, an additional

condition is that the weight of interface I is the larger one of the two interfaces. If any of the conditions is not satisfied, then the interface handler waits for the next computation of the weight for that interface. A pseudo-code of the hybrid wired/wireless protocol is given in Algorithm 1.

Based on the weights on the CAN and ZigBee interfaces, w^{CAN} and w^{ZigBee}, the hybrid wired/wireless protocol schedules packet transmissions in the following way:

1. When $w^{CAN} \leq 0$ and $w^{ZigBee} > 0$, the next packet is transmitted on the ZigBee link;
2. When $w^{CAN} > 0$ and $w^{ZigBee} \leq 0$, the next packet is transmitted on the CAN link;
3. When $w^{CAN} \leq 0$ and $w^{ZigBee} \leq 0$, the next packet is not transmitted;
4. When $w^{CAN} > 0$ and $w^{ZigBee} > 0$, whether the next packet will be transmitted on the CAN interface or the ZigBee interface depends on whether the CAN/ZigBee interface handlers are busy transmitting packets. If both interface handlers are idle, the next packet is scheduled on the link with the larger weight. If one of the interface handlers is busy, the next packet is transmitted on the interface which is idle.

V. Performance Evaluation

In this section, we evaluate the performance of the hybrid wired/wireless protocol under a DoS attack launched by two attackers with MAC-layer queue capacity equal to one. In addition, we provide experimental results of the hybrid wired/wireless protocol under wireless jamming attacks.

A. DoS attack with two attackers

In the hybrid testbed, as shown in Fig. 4, we use the VN1610 CAN interface as the CAN transceiver and the TelosB mote as the ZigBee transceiver.

The setup of the DoS attack on the hybrid wired/wireless protocol is shown in Fig. 8. Similar to the demonstration in Section III, each of the attackers generates high-priority messages at the rate of 150 pkts/sec. For comparison, we refer to the *native CAN protocol* as the transmission scheme in which the legitimate node periodically generates data packets and transfers them to the CAN transceiver.

Fig. 9 shows the average delay of packets generated by the legitimate node under the native CAN protocol and the hybrid wired/wireless protocol. We can see that the average delay of the native CAN protocol under a two-node DoS attack reaches high values exceeding 1000 ms (e.g., 1122 ms at a packet generation rate of 5 pkts/sec). As explained in Section III, the two high-priority attackers take turns on transmitting messages on the CAN bus. The low-priority legitimate node is almost starved and must wait for a long time between each successful transmission. This result indicates that a two-node DoS attack dramatically increases the MAC delay of the legitimate node.

On the other hand, under a DoS attack launched by two high-priority attackers, the hybrid wired/wireless protocol measures a high ETX on the CAN link and schedules packet transmissions on the ZigBee link instead. Therefore, the hybrid wired/wireless protocol achieves higher packet delivery rate and higher throughput than the CAN protocol, as shown in Fig. 10(a) and 10(b). More specifically, the hybrid wired/wireless protocol achieves a throughput of 19.47 pkts/sec at a rate of 20 pkts/sec, more than four times of that of the native CAN protocol. These results demonstrate that the wired/wireless protocol can protect the CAN bus against DoS attacks.

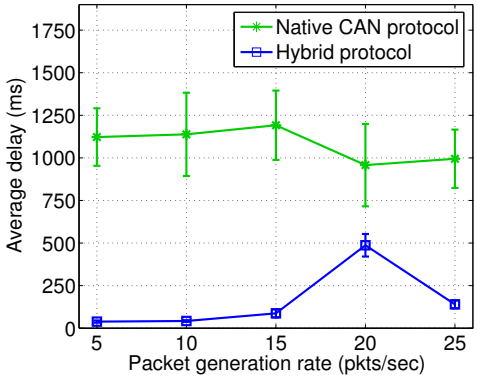
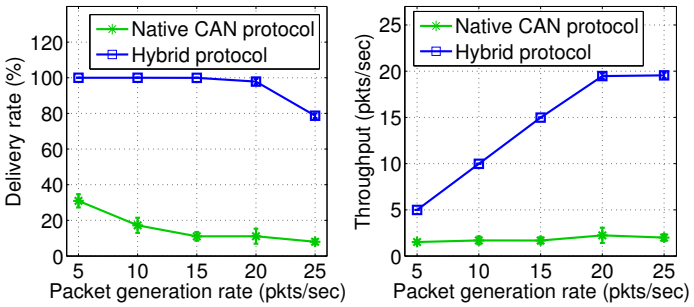


Fig. 9: Average delay of the native CAN protocol and the hybrid wired/wireless protocol under a two-node DoS attack (with the MAC-layer queue capacity equal to one) at the attacking rate of 300 pkts/sec. The two-node DoS attack causes a large MAC delay for the legitimate node.



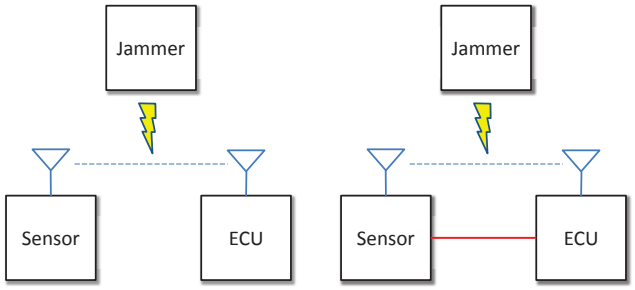
(a) Delivery rate of the legitimate node versus its packet generation rate.

(b) Throughput of the legitimate node versus its packet generation rate.

Fig. 10: Performance of the hybrid wired/wireless protocol with two attackers (with the MAC-layer queue capacity equal to one) attacking at a total rate of 300 pkts/sec. The hybrid wired/wireless protocol measures a high ETX on the CAN link and schedules packet transmissions on the ZigBee link. Thus it achieves higher delivery rate and throughput than the native CAN protocol.

B. Wireless jamming

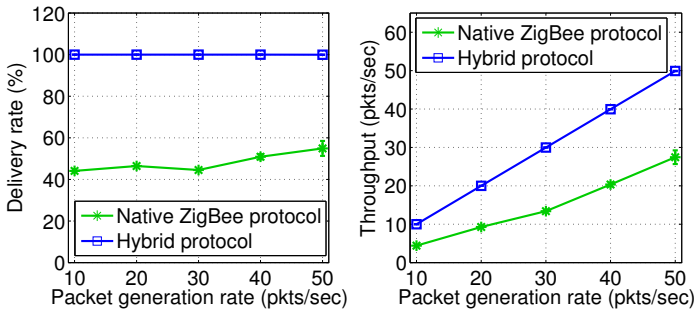
In the wireless jamming tests, a wireless jammer performs protocol-compliant jamming attacks on the ZigBee link. The jammer periodically generates packets and broadcasts them on the ZigBee link. In our tests, the attacking rate of the wireless jammer is 100 pkts/sec. We compare the hybrid wired/wireless



(a) Wireless jamming on the ZigBee link.

(b) Wireless jamming on the hybrid network.

Fig. 11: Setup of jamming attacks on the wireless link.



(a) Delivery rate of the legitimate node versus its packet generation rate.

(b) Throughput of the legitimate node versus its packet generation rate.

Fig. 12: Performance of the hybrid wired/wireless protocol with a wireless jammer attacking at the rate of 100 pkts/sec.

protocol with the *native ZigBee protocol*, which simply periodically generates data packets and sends them on the wireless link to the sink. The native ZigBee protocol is tested in the network of Fig. 11(a) and the hybrid wired/wireless protocol is tested in network of Fig. 11(b).

Comparisons between the delivery rate and throughput of the native ZigBee protocol and those of the hybrid wired/wireless protocol are shown in Fig. 12(a) and Fig. 12(b). The results show that under wireless jamming, the delivery rate of the ZigBee protocol is at most 54.90% at a packet generation rate of 50 pkts/sec, while the hybrid wired/wireless protocol achieves a delivery rate of 99.95%. The results show that the hybrid wired/wireless protocol is resilient to wireless jamming and schedules packet transmissions on the wired link when the wireless link experiences congestion.

VI. Conclusion

In this paper, we demonstrated via real testbed experiments that properly crafted protocol-compliant DoS attacks can consume over 98% of the CAN bus bandwidth. To address such attacks, we proposed a new hybrid wired/wireless protocol, which monitors link qualities and accordingly schedules packet transmissions. We showed that the hybrid wired/wireless protocol is robust to DoS attacks both on the wired and wireless links. Our solution does not only improve the security but also increase the capacity of the communication system under normal operations.

The throughput improvement brought by the hybrid wired/wireless protocol would be even more apparent if the ZigBee link were replaced with a wireless technology, such as Wi-Fi, that supports transmissions at higher data rates. We leave tests of a hybrid CAN/Wi-Fi network as future work.

With the introduction of wireless communication into an intra-vehicular network, security issues such as privacy and packet spoofing could emerge [15]. One should take these issues into consideration while further improving the hybrid wired/wireless protocol.

Acknowledgements

This work was supported by the National Science Foundation under grants CNS-1012910 and CNS-1409053.

References

[1] Robert Bosch GmbH, "CAN specification 2.0," 1991.

[2] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, and S. Savage, "Experimental security analysis of a modern automobile," in *Security and Privacy (SP), 2010 IEEE Symposium on*, May 2010, pp. 447–462.

[3] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno *et al.*, "Comprehensive experimental analyses of automotive attack surfaces." in *USENIX Security Symposium*. San Francisco, 2011.

[4] M. Hashemi, W. Si, M. Laifenfeld, D. Starobinski, and A. Trachtenberg, "Intra-car multihop wireless sensor networking: a case study," *Communications Magazine, IEEE*, vol. 52, no. 12, pp. 183–191, December 2014.

[5] "Hackers remotely kill a jeep on the highway - with me in it," http://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/.

[6] C. Miller and C. Valasek, "A survey of remote automotive attack surfaces," *Black Hat USA*, 2014.

[7] A. Van Herrewege, D. Singelee, and I. Verbauwhede, "CANAuth-a simple, backward compatible broadcast authentication protocol for can bus," in *ECRYPT Workshop on Lightweight Cryptography 2011*, 2011.

[8] O. Hartkopp and R. M. Schilling, "Message authenticated can," in *Escar Conference, Berlin, Germany*, 2012.

[9] B. Groza, S. Murvay, A. van Herrewege, and I. Verbauwhede, "Libracan: A lightweight broadcast authentication protocol for controller area networks," in *Cryptology and Network Security*, ser. Lecture Notes in Computer Science, J. Pieprzyk, A.-R. Sadeghi, and M. Manulis, Eds. Springer Berlin Heidelberg, 2012, vol. 7712, pp. 185–200.

[10] A. Hazem and H. A. Fahmy, "Lcap-a lightweight can authentication protocol for securing in-vehicle networks," in *10th escar Embedded Security in Cars Conference, Berlin, Germany*, vol. 6, 2012.

[11] K. Han, S. D. Potluri, and K. G. Shin, "On authentication in a connected vehicle: Secure integration of mobile devices with vehicular networks," in *Proceedings of the ACM/IEEE 4th International Conference on Cyber-Physical Systems*, ser. ICCPS '13. New York, NY, USA: ACM, 2013, pp. 160–169.

[12] Vector, "VN1600 Network Interface for CAN, LIN, K-Line, J1708 and IO," http://vector.com/vi_vn1600_en.html.

[13] S. Moeller, A. Sridharan, B. Krishnamachari, and O. Gnawali, "Routing without routes: the backpressure collection protocol," in *IPSN*, 2010.

[14] W. Si and D. Starobinski, "On the channel-sensitive delay behavior of LIFO-backpressure," in *Communication, Control, and Computing (Allerton), 2013 51st Annual Allerton Conference on*.

[15] R. M. Ishtiaq Roufa, H. Mustafaa, S. O. Travis Taylora, W. Xua, M. Gruteserb, W. Trappeb, and I. Seskarb, "Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study," in *19th USENIX Security Symposium, Washington DC*, 2010, pp. 11–13.