Probing Attacks on Physical Layer Key Agreement for Automotive Controller Area Networks

Shalabh Jain, Qian Wang[1], Md Tanvir Arafin[1] and Jorge Guajardo
Robert Bosch LLC, Research and Technology Center, Pittsburgh, PA 15222, USA
Email: {shalabh.jain, jorge.guajardomerchan}@us.bosch.com
[1] Department of Electrical Engineering, University of Maryland, College Park, MD 20742, USA
Email: {marafin,qwang126}@umd.edu

***Abstract*—Efficient key management for automotive networks (CAN) is critical, governing the adoption of security in the next generation of vehicles. A recent promising approach for dynamic key agreement between groups of nodes, *Plug-and-Secure* for CAN, has been demonstrated to be information theoretically secure based on the physical properties of the CAN bus. In this paper, we illustrate side-channel attacks on the scheme, leading to nearly-complete leakage of the derived secret key bits to an adversary that is capable of probing the CAN bus. We identify the fundamental network properties that lead to such attacks and propose ideas to minimize the information leakage at the hardware, controller and system levels.**

I. Introduction

Over the past few years, an increase in the external network interfaces on a traditional automobile has drawn the attention of the security community. Several high profile attacks have been demonstrated on the modern car by researchers, e.g. [1]–[3]. These attacks are primarily facilitated by the lack of security (authentication, encryption) in the existing architecture of the controller area network (CAN). Concurrently, several techniques, e.g. [4]–[6], have been proposed for integrating security into the current architecture. However, provisioning, maintenance, and update of cryptographic keys required for the proposed systems is difficult within the automotive supply chain, and may require changes to the manufacturing and servicing facilities. Several commercial systems to enable such a process have been proposed, e.g. in [7].

Dynamic generation and distribution of keys in a secure manner can simplify and reduce the functional requirements of these provisioning systems. One promising approach, *Plug-and-Secure* for CAN (PnS-CAN), has been proposed recently towards this goal, in [8], [9]. A key advantage of the *Plug-and-Secure* scheme is the utilization of inherent physical layer properties of the CAN bus, in lieu of complex mathematical operations, to provide security guarantees for key agreement between groups of nodes.

Cryptographic systems that are provably secure in the computational model have often been compromised by exploiting characteristics of their physical implementation. Physical characteristics such as timing differences, power leakage, and other features or radiation can provide a covert communication medium, *side-channels*, leaking system information to an adversarial observer. Several attacks have been demonstrated on traditional systems using side channels, e.g. in [10]–[12]. Comprehensive analysis of a system to identify and exploit such side-channels can be difficult. It has been observed from traditional systems that prevention of such attacks can add significant overhead to system design and negatively impact performance.

In this paper, we demonstrate that such techniques can be utilized to violate the security of the PnS-CAN system by an adversary capable of probing the CAN bus signals. Here we discuss several *voltage*, *timing* and *transient characteristics* based side-channels that can be used to attack the system and partially extract the secret keys. We then propose countermeasures that can be implemented at different levels of the system, namely low-level hardware changes, controller modifications, and system level changes. Our methods align well with existing system configuration and result in little overhead.

A. Our Contributions

We investigate the side-channels for the two-party PnS-CAN protocol proposed in [8]. Our contributions are as follows,

- We identify characteristics of the CAN bus that can be utilized to extract the secret key during execution of the PnS-CAN protocol.
- We outline a general strategy to use fingerprinting techniques to attack the scheme. We demonstrate the attack using simple timing features.
- We discuss principles that can be used to design effective countermeasures against these attacks.

B. Related Work

Our results extend the literature of traditional side-channels, e.g. [10], [11], [13], to the CAN bus, using differences in the physical node properties to extract the secret key. To the best of our knowledge, there is no work in literature demonstrating attacks on the PnS-CAN scheme. Another difference of our work from existing literature is that traditionally, side-channel attacks extract a secret key based on usage of the key, while we extract the secret key during its *derivation* phase, i.e. prior to storage. Thus, existing methods can be applied in conjunction with our work to extract any secret bits that remain.

[1]Equally contributing authors. Work performed while the authors were at Bosch Research and Technology Center, Pittsburgh, PA

978-1-5386-7471-0/18/$31.00

In another direction of work on the CAN bus, researchers utilize physical signal characteristics to fingerprint the network nodes, primarily for the design of intrusion detection systems (IDS). These results, e.g. in [14], highlight the existence of subtle, yet identifiable differences between the transmissions from different nodes on the CAN bus. Since the PnS-CAN system relies on the inability of an adversary to identify the transmitter, these transmitter fingerprints can assist in attacking PnS-CAN. There are several challenges in applying these for the PnS system. For example, existing systems rely on the classification of a message frame consisting of a single transmitter and sync source. For the PnS-CAN system however, this assumption does not hold. Typically, these differences used to increase system security (IDS based on the identified features), whereas we use them to attack the system.

II. Preliminaries

A. Notation

We utilize the following notation for the paper. For two nodes nodeN_1, nodeN_2 executing the protocol, we denote by $x - y$, the simultaneous transmission of x by the primary node, i.e. nodeN_1 and y by nodeN_2. Here, $x, y \in \{0, 1\}$ are logical bits. We denote a gateway by GW, the central entity that controls or initiates the execution of the key agreement protocol (typically the central gateway).

B. CAN Bus Physical Layer

CAN bus, the primary communication network for modern day cars, is a broadcast medium consisting of a series of nodes connected via a twisted-pair cable with termination impedance at either end. It has two logical states, the dominant '0' state, where the bus is driven by a voltage, and the recessive '1' state, where the bus is grounded. If two nodes transmit a bit simultaneously, the effective state of the bus is dominant '0' if any of the nodes transmits the dominant bit. Thus, the bus acts as a logical AND gate between inputs from the nodes.

The CAN bus utilizes differential signaling to transmit data. In the CAN standard, when transmitting the dominant bit 0 on the bus, the output pins of the nodes, CANH and CANL, are driven to different voltage levels, and the difference from CANH to CANL is the output of the CAN bus. Similarly, transmission of a recessive bit 1 occurs when CANH and CANL are not driven and will have similar voltage levels.

C. System Model

Aside from the requirements of the system under attack, [9], we assume the typical CAN bus to be comprised of heterogeneous nodes, i.e. nodes from different manufacturers or families. Since the PnS-CAN protocols are based on simultaneous transmission by two nodes, all write operations on the bus during the key agreement phase use ECU pairs. However, during regular operation, the network operates in standard CAN manner, i.e. a single transmitter.

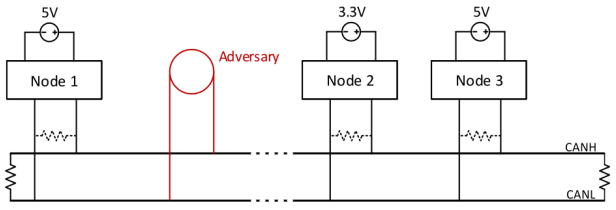


Fig. 1: A typical CAN bus with 3 nodes and a passive (eavesdropping) adversary

D. Adversarial Model

We consider a simple passive adversary that is capable of observing the variation of CAN bus signals with high voltage precision and timing resolution. Such an adversary can simply be realized by a regular ECU connected to the CAN bus with a high precision analog-to-digital converter (ADC) at the front end and a modified CAN controller capable of sampling the bus at a high frequency. An alternate means could by directly accessing the physical wires with measurement equipment such as an oscilloscope. In a car, such nodes can be connected to the OBD-II diagnostics port. A representation of the CAN bus with an attacker is illustrated in Fig. 1. For this paper, we limit the adversarial access to a single point on the network. Intuitively, multiple points of observation can increase the leakage. We defer that aspect for future analysis.

E. Experimental Setup

Our experimental setup, illustrated in Fig. 3a utilizes a modified CAN controller to instantiate the PnS-TwoParty scheme described in Protocol 1, implemented using Altera CycloneV FPGAs. The controller generates fully compliant CAN 2.0 frames that are accepted by any traditional CAN controller. The test network consists of 16 nodes, implemented using commercial CAN transceivers that utilize the FPGA CAN controller digital outputs, and connect together via the standard CAN twisted pair cable.

The typical CAN architecture consists of several subnetworks of ECUs connected by one or more powerful nodes that act as gateway nodes. However, for our system, the nodes are connected in a single chain with the GW at one end. This setup closely emulates one subnet of the network found in most modern cars. We use a commercial USB-CAN module to monitor the bus traffic. We use an Agilent 6012A oscilloscope to probe the bus and record the samples for offline processing.

F. PnS-CAN Scheme

The PnS-CAN scheme described in [8], [9], enables key agreement between multiple nodes connected to the CAN bus. For completeness, we present a brief overview of the two-party scheme from [9]. For details about the security aspects of the original scheme, the reader is referred to [9].

The PnS-CAN scheme between two nodes utilizes the wired AND property of the CAN bus to mask the bits simultaneously transmitted by the nodes. The security of this scheme is based on the inability of an eavesdropper to differentiate between transmissions that result in the same logical output on the bus, i.e. combinations that result in the dominant (0) output. The operation of the basic two-party protocol between nodeN_1 and nodeN_2, using random seeds r_1, r_2 can be described as follows in Protocol 1.

Protocol 1: **PnS-TwoParty (nodeN$_1$, nodeN$_2$, r_1, r_2)**

1) nodeN$_1$ generates a sequence of psueodrandom bits, s_1 using seed r_1. The sequence is split into chunks ($c_1(i)$) of half the size of the maximum CAN 2.0 payload. i.e. $s_1 = \{c_1(1)||c_1(2)||\ldots\}$. Similarly, nodeN$_2$ generates a sequence of bits, $s_2 = \{c_2(1)||c_2(2)||\ldots\}$, using seed r_2.
2) The bits in each chunk are interleaved with their complements to produce packets, namely $\{p_1(1), p_1(2), \ldots\}$ and $\{p_2(1), p_2(2), \ldots\}$. Note for interleaving, each 0 bit is replaced by the pair 01 and 1 is replaced by 10.
3) nodeN$_1$ initiates the transmission by using a dedicated PnS-CAN header and $p_1(1)$ as payload. nodeN$_2$ synchronizes to the header and simultaneously transmits packet $p_2(1)$ during the payload phase. For compliance with regular CAN frames, each node dynamically performs bit stuffing and CRC computation based on the bus outputs.
4) nodeN$_1$ and nodeN$_2$ sample the first resulting packet payload as b(1). The bits are de-interleaved to identify the result of the original and complement transmission. Location within the transmitted chunk, where both the non-inverted and inverted bits produced a 0 in the bus output, are hidden from an ideal eavesdropper. These secret bits are retained and the remaining bits are discarded.
5) The resulting sequence at nodeN$_1$(remaining bits in the chunk) are always inverse of the sequence at nodeN$_2$. Thus nodeN$_2$ inverts its sequence to obtain a sequence of secret bits identical to nodeN$_1$. The process is repeated from 3 until the desired number of secret bits are generated.

In this protocol, a secret bit is generated between nodeN$_1$ and nodeN$_2$ when one of the nodes transmits a logical 0 (dominant bit) while the other transmits a logical 1 (recessive bit). Note that an ideal adversary is unable to identify which of the two nodes, nodeN$_1$ or nodeN$_2$ transmitted the dominant bit. The group key scheme described in [9] utilizes the PnS-two-party protocol between successive nodes in a group to form a PnS chain. It was demonstrated that pairwise interactions between consecutive ECUs, where the result of a stage is used in the next stage, are sufficient to obtain a secure group key. The security of the group key protocol depends on the security of the pairwise PnS-TwoParty scheme.

In systems that only allow high-level (software) access to the nodes, the adversaries can only observe the logical output of the bus, as determined by a single (or three) transceiver samples. Thus, perfect secrecy of the bit values in PnS-TwoParty is theoretically possible. However, as we will show, it cannot be guaranteed for practical systems, as adversaries may observe multiple high-resolution bus samples.

III. Attacks on PnS-CAN

As described in Section II-F, in the PnS-TwoParty system, a secret bit results when one of the nodes transmits the dominant bit 0, i.e. drives the bus, and the other node transmits the recessive bit 1, i.e. performs no action. Though in the PnS-CAN system, nodes transmit messages as full frames, for each bit of significance (secret bit), only a *single* node is driving the bus. Thus identification of the transmitting node effectively leaks the bit, as the bit is 0 if the driving node is the primary participant and 1 otherwise.

For example, consider a PnS interaction between nodeN$_1$ and nodeN$_2$ using the *random* sequences $\{0, 1, 1, 0\}$ and $\{1, 0, 0, 1\}$ respectively. This results in 4 shared secret bits, i.e. key = 0110. The bit observations on the bus corresponding to these would comprise of 8 bits (random bits interleaved with their complements). $\{(b_1, b_2), \ldots, (b_7, b_8),\ b_i = 0, i \leq 8$. Here, b_1 results from nodeN$_1$ transmitting a dominant bit and nodeN$_2$ a recessive bit. If the adversary can identify nodeN$_1$ as the active node during the transmission b_1, it can learn that the first secret bit is 0. This can be similarly extended to all bits. We now describe the physical characteristics that can be used to identify the transmitter.

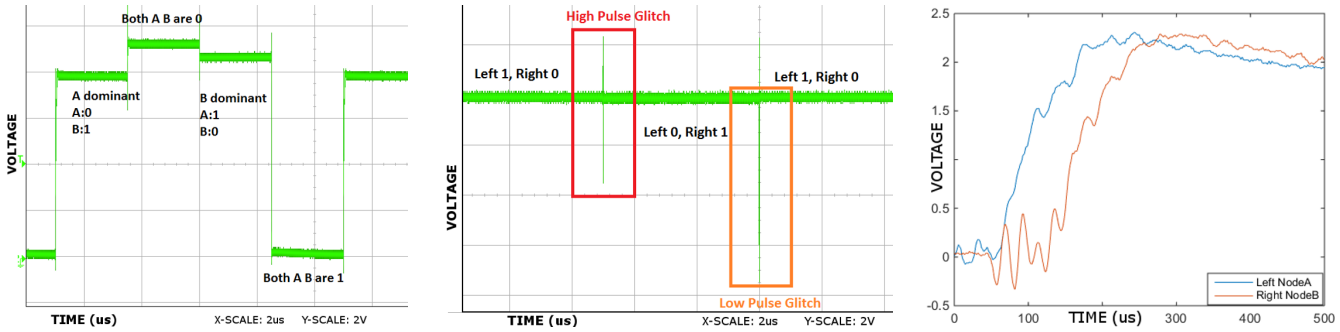
A. Physical Characteristics

Similar to other electrical systems, an automotive network has differences in characteristics of the driving nodes and network between an observer and the transmitter. We illustrate three phenomena that can be utilized to differentiate between bits transmitted by different nodes.

Steady state characteristics - CAN differential signaling enables devices with varying electrical characteristics to be utilized on the same bus without any additional compensation circuitry. While this improves the design and robustness of the bus, it can enable identification of the transmitter due to,

1) Driver circuits - Transceivers from different manufacturers (or different models of the same manufacturer) can have different drive characteristics and output voltage range of the CANH and CANL pins. This can be due to different circuits, components or load impedance. Thus an adversary measuring the absolute voltage on CANH and CANL lines with respect to a common ground reference can distinguish between the dominant transmissions from different nodes.
   For example, consider the network in Fig. 1 using Microchip (MCP2551) for nodeN$_1$ and NXP (TJA1040) for nodeN$_2$. The specified range for the CANH pin for MCP2551 is between $2.75V$ and $4.5V$, while the same range for TJA1040 is between $3V$ and $4.25V$. Fig. 2a illustrates the voltage observations of the adversary for a sequence of transmissions to generate a secret bit, i.e. transition from a $0-1$ scenario to a $1-0$ scenario. The adversary can clearly distinguish between the dominant transmission by nodeN$_1$ and nodeN$_2$.
2) Physical location - Even nodes with identical drivers and operating voltages can seem different from the view of an observer in the network due to the differences in the effective impedance of the network segment between the two transceivers and the observer. Several factors can contribute to these differences, e.g. different length of wires between the nodes and the observer, or a different number of intermediate nodes. In a typical CAN network, the difference in distance can be over $30m$, leading to significant variation. Though this difference appears small in comparison to other phenomena, it can be useful in many scenarios.

Transient characteristics - The CAN physical medium has a non-negligible capacitance and inductance that influences the signal as it propagates. Coupled with inductive-loading of intermediate transceivers between an observer and the nodes, these can have an observable impact on the signals. Thus, as the state of a node transition from recessive to dominant



(a) Signal (voltage) observations for scenario of 2 nodes, A:MCP2551, B:TJA1040

(b) Signal (voltage) observations of an adversary centrally located between two transmitters

(c) Voltage obs. of signal transients for an adversary, centrally located b/w two identical txs.

Fig. 2: Signal observation from CAN transceivers

(or vice versa), it exhibits different transient characteristics. The sample point of a typical CAN bit is sufficiently delayed to ensure robustness against such transients. However, an adversary observing with a higher sampling rate can use these differences to identify the transmitter.

Such transients can vary widely even for similar nodes placed symmetrically with respect to an observer. As illustrated in Fig. 2c, observations by an adversary symmetrically placed in Fig. 1 relative to two identical transmitters can be clearly distinguished, based on time-domain characteristics. However, a disadvantage of the transients is that the performance is highly dependent on the features that are derived for identification. Several such features have been enumerated in [14].

Timing characteristics - The typical propagation delay for the twisted pair cable used as the physical medium for the CAN bus is $5ns/m$. Thus for a traditional network of length up to $50m$, the difference in the time the transmitter drives (or releases) the bus and an observer observes a signal transition can be up to $250ns$. Though such delays are accommodated within the CAN bit timing specification for a correct sampling of the bit value, they can be exploited by an adversary to identify the transmitter.

The relative bit timing observed by an adversary for two transmitters can be influenced by two factors, the difference in propagation delay between the observer and the transmitters and the synchronization offset between the transmitters. Fig. 2b, illustrates the observations of an adversary symmetrically placed relative to two identical transmitters. It can be seen, a propagation difference of 0.02% of the bit timing is sufficient to distinguish between the dominant transmission from the left node and the right node, i.e. the scenario of secret bit generation, $1-0$ followed by $0-1$ transmission in PnS-CAN.

B. Generalized Attack Strategy

Each PnS interaction occurs between two nodes in the network. As discussed earlier, the attack on the PnS-CAN system simply reduces to the identification of the transmitting node for each individual bit of the frame. Since there are just two nodes in each PnS-CAN round, the adversary simply needs to *distinguish* between the signals transmitted from the two nodes (binary hypothesis testing). Here, we present a general outline of the attack methodology that an attacker may follow.

(Data Collection) The adversary can collect data from different transmitters on the bus. Similar to any CAN-compatible node, it can synchronize to the transmitter, then sample and save the bits in a transmitted frame. For traditional CAN networks, such data is not labeled as the packets do not contain identifying information. However, an adversary can generate labels for the data by activating known ECUs (sending a query) and observing their output.

(Classifier Training) The adversary can utilize a variety of supervised or unsupervised techniques, e.g. binary support vector machines (SVM), LSTM, Convolutional Neural Networks (CNN), to train classifiers for each node based on signal observations.

(PnS-CAN compromise) During PnS-CAN operation, an adversary does not require to perfectly identify the node to extract the key. Since the PnS system involves just two ECUs at any time, its task is to simply distinguish between the transmissions from the two participating ECUs, which leads to the key compromise within 1-bit entropy (i.e. key vs. $\overline{key}$).

We note that methods from CAN identification literature, e.g. from [14], [15], can be directly mapped to this generalized outline to attack the PnS-CAN scheme. However, the accuracy may be lower since the identification is for a single bit rather than a group of bits (frame).

C. Timing Based Attack Evaluation

We demonstrate an attack on the PnS-CAN scheme using the timing characteristics described in Section III-A. Our choice of the timing parameter is governed by several factors

1) Timing differences are independent of the node characteristics. In fact, they depend only on synchronization and propagation aspects. Thus they can be applied to all networks.
2) Timing attacks demonstrate the vulnerability of the unprotected system against a very simple adversary.
3) The timing component represents a feature that has not been specifically evaluated in [15]. Other features from [15] can be directly applied to attack PnS-CAN via our framework described in Section III-B.

We sample the differential bus voltage at $125Msamp/s$. The transitions are identified as points of large change in bus

Node ID	Delay (ns) Min	Max	Mean	Std	Node ID	Delay (ns) Min	Max	Mean	Std
1	138	166	151.8	12.6	9	118	154	135.2	15.7
2	140	168	153.4	12.5	10	118	154	135.0	15.3
3	140	168	153.8	12.6	11	122	156	139.1	14.7
4	140	172	156.2	12.9	12	124	158	140.9	14.6
5	130	162	144.4	14.2	13	116	146	130.6	12.6
6	130	160	144.8	14.0	14	118	146	131.2	12.3
7	132	164	147.1	13.7	15	118	152	135.4	14.8
8	136	164	148.7	12.4	16	122	154	137.2	13.3

TABLE I: Signal delays between nodes and observer

Interval overlap	N_1	N_2
6.00	2	13
6.00	2	14
6.00	3	13
32.00	11	12
34.00	9	15
34.00	10	15
36.00	9	10

TABLE II: Maximum and minimum overlaps

voltage (greater than the CAN trigger) followed by a steady state over at least half the bit width. We utilize the 50% point of the transition to compute the rise time, fall time and latency.

First, we investigate the separability of the nodes by measuring the propagation delays of the nodes synced with respect to the observation point. In Table I, we enumerate the propagation delays from each node. Intuitively, nodes that have similar propagation delay would be difficult to differentiate (in the perfectly synchronized scenario). Further, in Table II, we enumerate combinations of nodes that have the least (and maximum) overlap. Such nodes pairs correspond to the nodes with the largest (and smallest) adversarial advantage.

Fig. 3b illustrates a snapshot of the CAN protocol between two nodes as observed by an adversary. Intuitively, the attack exploits the difference in propagation delays by synchronizing to one of the transceivers and estimating the transmitter based on the rise/fall time offset. There are two key features of the PnS-CAN implementation that aid our attack.

1) The initiating node transmits the PnS header and the secondary node synchronizes to the initiating node. Since only a single node is transmitting during the header phase, it is easy to estimate the timing variation for the synchronous bits.
2) As described in Section II-F, the random bits are interleaved with the inverted bits. This introduces dependence in successive transmissions as it enables only certain transitions during the PnS phase. This can be utilized to estimate the bits in some cases.

The detailed attack is described as *Attack PnS-TwoParty*. We utilized this to identify the secret key for 12 pairs of nodes in our setup. With a minor modification of the threshold parameters between different iterations, we were able to successfully identify all the exchanged secret bits.

Attack: PnS-TwoParty

1) The adversary synchronizes to the first 1 to 0 transition, i.e. start of frame (SOF) bit. The transmitting node is the referred to as the primary node.
2) The adversary utilizes the first 19 bits of the header to estimate the expected variation parameters of transition times (μ_p, σ_p).

Start of the PnS data frame:

3) If the bit value has changed, compute the transition time. Compute the bit triggering the transition by comparing the transition time to a threshold τ, a function of (μ_p, σ_p).
4) If bit value has not changed,

a) If this is corresponding to the first bit (non-inverted), then the possible transitions are $0-0 \rightarrow 0-1$, $0-1 \rightarrow 1-0$, $0-0 \rightarrow 1-0$, and vice versa. If the current voltage level is higher than the previous bit ((b) in Fig. 3b), both nodes transmitted a dominant value for the current bit. Otherwise, if the level decreases, nodes transitioned to a $0-1$ or $1-0$ configuration. Utilize the next bit to compute the current value.

b) If this is corresponding to the second bit, it could have only resulted from a $0-1 \rightarrow 1-0$ transition or vice versa. If a dip is detected at the start of the frame ((a) in Fig. 3b), the primary node was transmitting a dominant bit in the previous frame. Otherwise, if an increase is detected ((c) in 3b), the secondary node was transmitting the 0.

Soft synchronization:

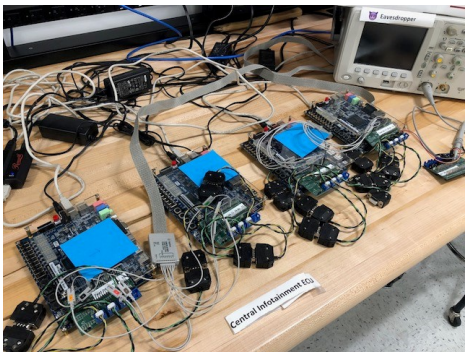
5) If the secondary node ever triggers a recessive to dominant transition, resync to the secondary node and switch the roles of the primary and secondary nodes. This is depicted by (r) in Fig. 3b.

IV. Possible Countermeasures

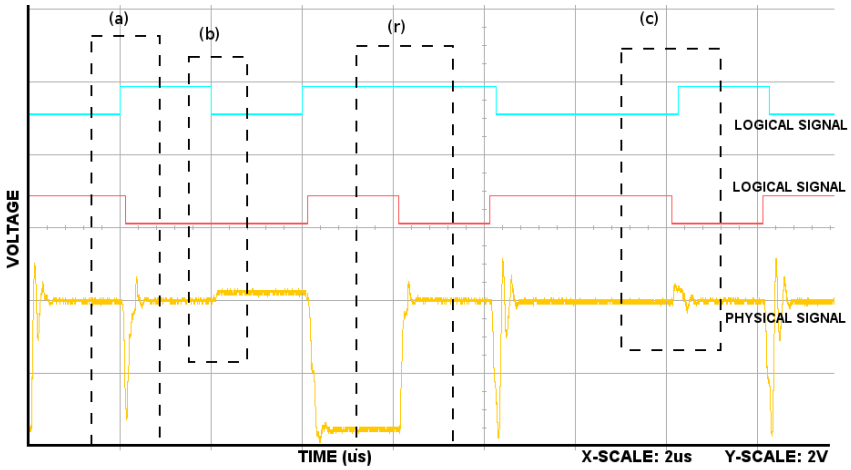
As discussed, the PnS-CAN module is susceptible to a range of attacks due to the consistency of the features of the physical signals from individual nodes. However, the CAN standard allows for significant variation of these properties due to the range of operating conditions. Thus, countermeasures can be designed by adding controlled noise to the physical signals, such that signals from different nodes are indistinguishable.

Based on this principle, we have identified a variety of countermeasures at different levels of abstractions, namely the transceiver hardware level, CAN controller level and the system level. However, due to space constraints, here we only provide a high-level intuition. We refer the reader to the full version of this paper for a comprehensive discussion [16]. At the hardware/controller level, one can minimize the adversarial advantage by adding noise through impedance variation, addition of timing jitter, or improving cooperation between the nodes. Operations at the system level can further reduce it by restricting interaction between highly differentiable nodes.

For example, consider the voltage observed by an adversary for a dominant transmission, which is a function of the strength of the current source, and impedance of the network between the observer and the source. For a non-uniform network, this directly relates to the physical positions of the source, the observer, and any intermediate sinks. To prevent identification, the voltage can be varied for each transmission of the dominant bit by using a random number of current sources and sinks located at different points in the network. An example of varying observations of an adversary due to multiple nodes



(a) Experiment setup



(b) Observations of the logical and physical signals for a snapshot of the PnS protocol

Fig. 3: Timing attack on PnS protocol - Setup and Results

TABLE III: V_{out} for dominant tx by multiple nodes with isolation

N_1	N_2	N_3	V_{out}	N_1	N_2	N_3	V_{out}
0	0	0	2.4230	X	0	0	2.5842
0	0	1	2.1281	X	0	1	2.1174
0	1	0	2.1197	X	1	0	2.0923
0	1	1	1.8208	0	0	X	2.3159
1	0	0	2.3400	0	1	X	1.9647
1	0	1	1.7710	1	0	X	2.1493
1	1	0	1.7629	0	X	0	2.2957
1	1	1	0.0000	0	X	1	1.9599

used to transmit the same bit is illustrated in Table III. The X in the table denotes a node that can be disconnected from the network by using an isolation circuit to vary the number of sinks.

Such a variation can be achieved in multiple ways. A simple, albeit expensive, hardware modification would be to equip each CAN controller with multiple transceivers. If each node in the PnS-CAN protocol is equipped with N transceivers, the system can cycle between $(3^N - 2^N)(2^N - 1)$ voltage levels for each secret bit. Thus, even a system with 2 transceivers per controller can produce 15 different voltage levels. Alternatively, such a variation can be achieved by cooperation between controllers (particularly simple for group keys).

V. Conclusions

In this paper, we discussed several sources of physical identifiers that can be used to attack the PnS-CAN scheme. The goal of this discussion is twofold. Firstly, to demonstrate that even simple physical features, such as timing, can be utilized to compromise the PnS-CAN scheme, which has promising theoretical guarantees. Secondly to illustrate that there are several simple fixes that can be designed to significantly reduce the leakage to the adversary.

We emphasize that neither of these investigations is intended to represent the comprehensive set of attack vectors or defense mechanisms. This work serves as a proof-of-concept for the existence of attacks on the PnS-CAN schemes and the ability of the system designer to prevent them. Further, since the performance of countermeasures based on noise addition is highly dependent on the CAN properties, they can help in minimizing the leakage, rather than provably eliminating it. The design of countermeasures that can formally be shown to be effective against the adversaries treated here remains an open problem.

References

[1] T. Hoppe, S. Kiltz, and J. Dittmann, *Security Threats to Automotive CAN Networks – Practical Examples and Selected Short-Term Countermeasures*, 2008, pp. 235–248.

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

[3] C. Valasek and C. Miller, "Remote exploitation of an unaltered passenger vehicle," www.ioactive.com/pdfs/IOActive_Remote_Car_Hacking.pdf, IOActive Inc., Tech. Rep., [Online: Accessed 2016-02-09].

[4] B. Glas, J. Guajardo, H. Hacioglu, M. Ihle, K. Wehefritz, and A. Yavuz, "Signal-based automotive comm. security and its interplay with safety req." *Embedded Security in Cars (ESCAR) Europe*, Nov 2012.

[5] A. V. Herrewege and I. Verbauwhede, "CANAuth - A Simple, Backward Compatible Broadcast Authentication Protocol for CAN bus," in *ECRYPT Workshop on Lightweight Cryptography 2011*, Louvain-La-Neuve,BE, 2011, pp. 229–235.

[6] D. Brown, G. Cooper, I. Gilvarry, A. Rajan, A. Tatourian, R. Venugopalan, D. Wheeler, and M. Zhao, "Automotive security best practices," *Intel White Paper*, pp. 1–17, 2015, [Accessed Mar-15-2017].

[7] N. Bißmeyer, "Security in ecu production," *ETAS White Paper*, 2016, [Accessed Mar-15-2017].

[8] A. Müller and T. Lothspeich, "Plug-and-secure communication for CAN," *CAN Newsletter*, pp. 10–14, Dec 2015.

[9] S. Jain and J. Guajardo, "Physical layer group key agreement for automotive controller area networks," in *Proc. of Crypto. Hardware and Embedded Systems (CHES 2016)*. Springer Berlin Heidelberg, 2016.

[10] P. C. Kocher, *Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems*. Springer, 1996, pp. 104–113.

[11] P. Kocher, J. Jaffe, and B. Jun, *Differential Power Analysis*. Springer, 1999, pp. 388–397.

[12] D. J. Bernstein, "Cache-timing attacks on AES," Tech. Rep., 2005.

[13] O. Acıiçmez, Ç. K. Koç, and J.-P. Seifert, *Predicting Secret Keys Via Branch Prediction*. Springer Berlin Heidelberg, 2006, pp. 225–242.

[14] K.-T. Cho and K. G. Shin, "Fingerprinting electronic control units for vehicle intrusion detection," in *Proc. USENIX Security Sym.* USENIX Association, Aug. 2016, pp. 911–927.

[15] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PLOS ONE*, vol. 11, no. 6, pp. 1–17, 06 2016.

[16] S. Jain, Q. Wang, M. T. Arafin, and J. Guajardo, "Probing attacks on physical layer key agreement for automotive controller area networks (extended)," in *ESCAR Europe 2017 (Online on arXiv:1810 [cs.CR])*.