ZBCAN: A Zero-Byte CAN Defense System

Khaled Serag, Rohit Bhatia, Akram Faqih, and Muslum Ozgur Ozmen,
Purdue University; Vireshwar Kumar, Indian Institute of Technology, Delhi;
Z. Berkay Celik and Dongyan Xu, Purdue University
Published in : 32nd USENIX Security Symposium

Outline

1. Introduction
2. Background
3. Related Work
4. Threat Model

Outline

5 ZBCAN

6 Security Analysis

7 Evaluation

8 Discussion and Conclusions

1

Introduction

Introduction

- Modern vehicles contain hundreds of sensors and actuators, administered by Electronic Control Units (ECUs), including brake, engine, and steering control units.
- The most central communication channel among ECUs is CAN.
- Although reliable and robust against electromagnetic interference, CAN lacks any security measures.
- Researchers have demonstrated the feasibility of remotely compromising an ECU on the CAN bus.

Introduction

- To secure CAN traffic, two primary approaches have been proposed

1. cryptographic approach

    - The first is its impact on performance as cryptographic operations incur an unaffordable processing overhead for most commercial ECUs

    - Another issue is the lack of intrusion confinement. Since most of these solutions use group keys, if one ECU gets compromised, it can impersonate any node in the group.

Introduction

2. intrusion detection (IDS) approach

   - First, IDSs take no measure to stop or prevent attacks.
   - Second, most CAN IDSs do not achieve single-message detection. Instead, they retrospectively detect flows of injected messages. This allows intermittent or gradual intrusions to pass unnoticed and contributes to these IDSs' inability to translate their attack detection into prevention, for a flow of messages is composed of a stream of individual messages.

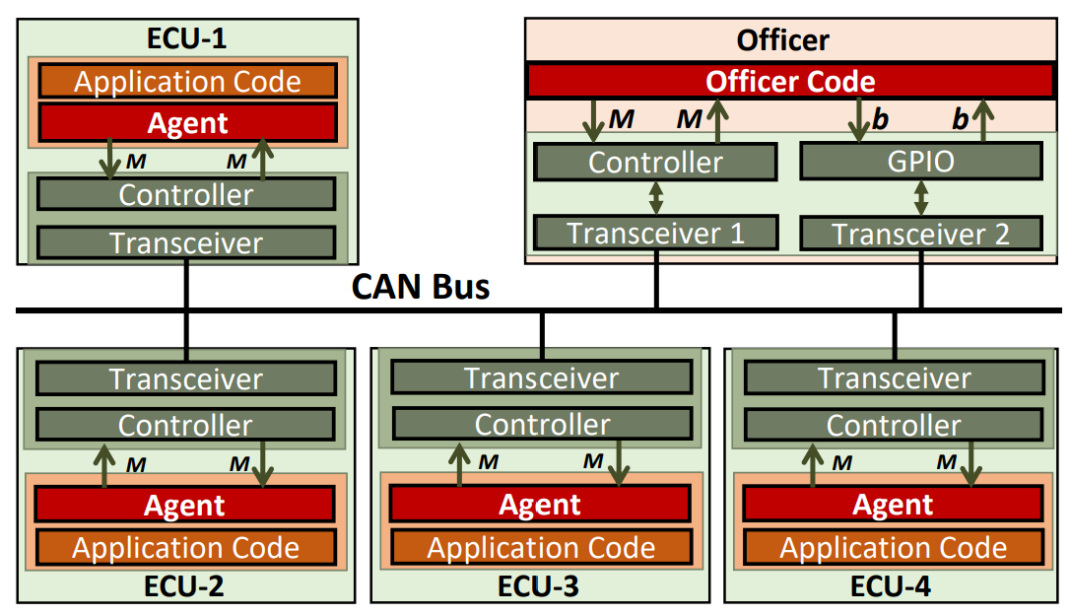
Introduction

- We present ZBCAN, a versatile defense system that uses inter-frame spaces to defend against the most common CAN attacks, offering both detection and prevention abilities.
- We introduce a new method to suspend any ECU as soon as it starts transmitting a frame called Instant Bus-Off. This method could be used to suspend intruding nodes.
- To show its applicability, we evaluate different aspects of our system on a CAN testbed, on a real vehicle's traffic, and directly on a real vehicle's CAN bus.

2

Background

ECU、CAN BUS、Arbitration and Priority

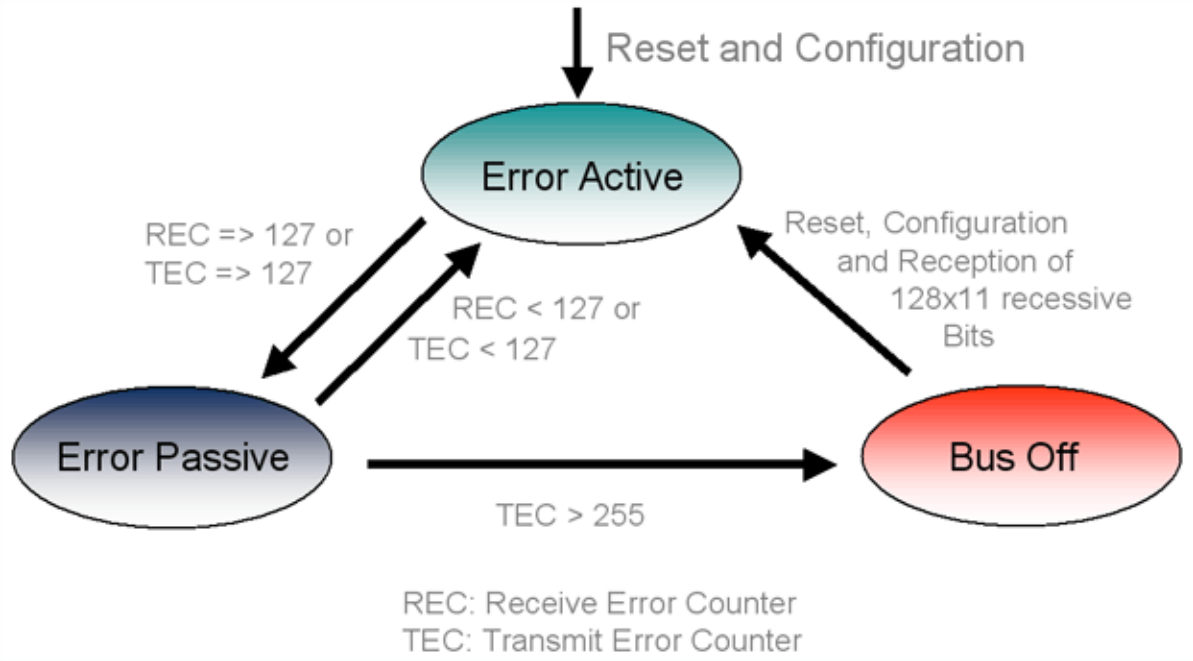


ECU、CAN BUS、Arbitration and Priority

Figure 2: CAN frame fields of two back-to-back frames.

CAN uses the ID as a priority field, with lower ID values having higher priorities.

Error States



Attacks

- Injection Attacks.
    1. Targeted Injection: Forging and injecting messages that look similar to those transmitted by a specific ECU in charge of certain functions in order to alter those functions.
    2. Replay Attacks: Replaying one or more messages transmitted by a different ECU.
    3. Random Injection: Forging IDs randomly or semi-randomly to cause damage or to discover hidden message semantics

Attacks

- Flooding Attacks.
    1. The attacker injects an endless stream of back-to-back high-priority messages to deny other ECUs access to the bus and cause them to drop messages.
- Error Handling Attacks.
    1. For instance by accumulating these errors, attackers could push ECUs to the error passive or bus-off states. These error states could then be exploited to launch persistent DoS attacks, evade voltage intrusion detection systems, or map the network.

3

Related Work

Intrusion Detection Approach.

- Some IDSs rely on traffic features such as message frequencies, lengths, payloads, or clock skews to detect anomalies.
- Others use physical features such as the unique electrical characteristics of each ECU, manifesting in their transmission voltage levels.
- Nevertheless, IDSs have their problems. Namely, many of these systems were shown to be evadable.

Timing-Based Approach.

- INCANTA proposed adding secret delays to the expected arrival times of periodic messages, with receivers inspecting the delay of every message.
- However, the accuracy of such delays degraded significantly for lower-priority IDs.
- CANTO suggested pre-scheduling bus traffic to avoid unexpected delays of lower priority messages.
- Unfortunately, both methods use up to 8 message bits and incur processing overhead on the receiving side.

4

Threat Model

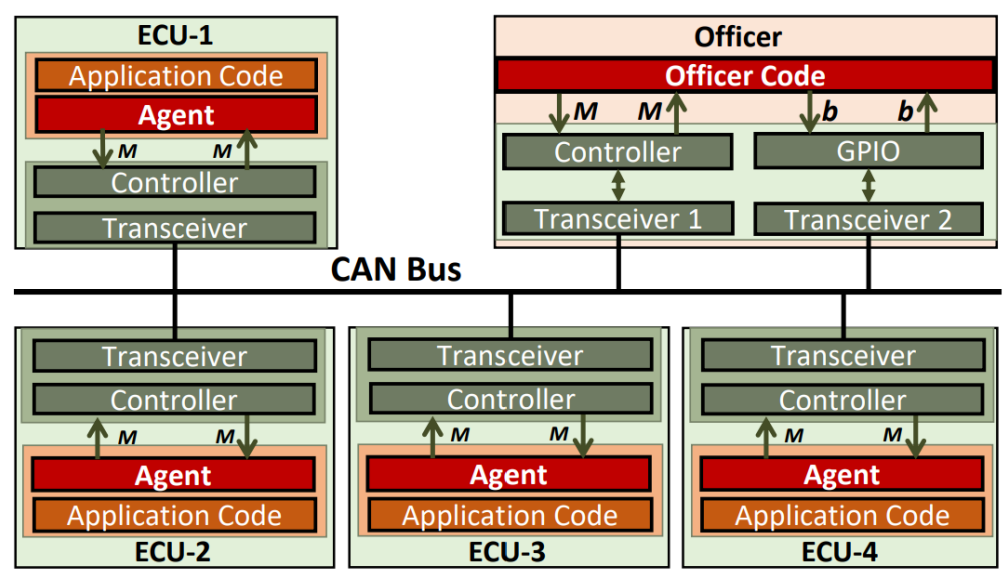
Threat Model

- We assume a remote attacker who has successfully compromised an ECU through Bluetooth, Internet, or any other remote means.
- The attacker can execute any code but has no control over the protocol controller and cannot alter protocol's rules.
- The attacker has no physical access to the bus and hence cannot attach devices with special hardware.

5

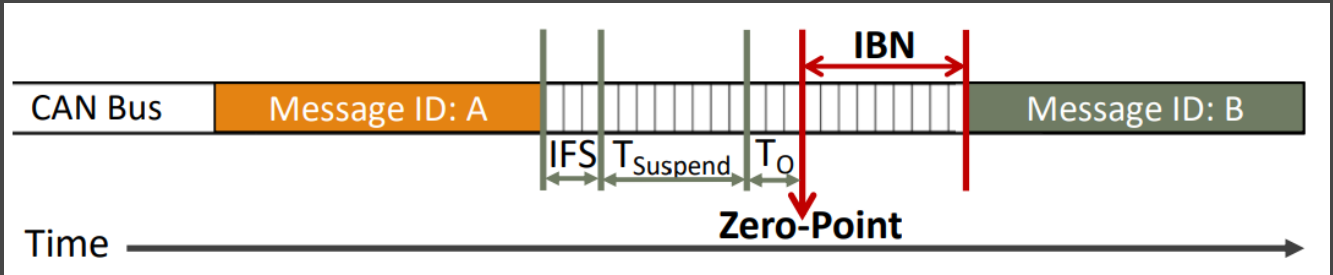
ZBCAN

Architecture and Operation Overview



Symbol (M) refers to messages. Symbol (b) refers to bits.

The In BetweeN (IBN)



First, nodes operating in the error-passive state have an additional 8-bit suspend-transmission penalty ($T_{suspend}$), enforced at the protocol controller's level.

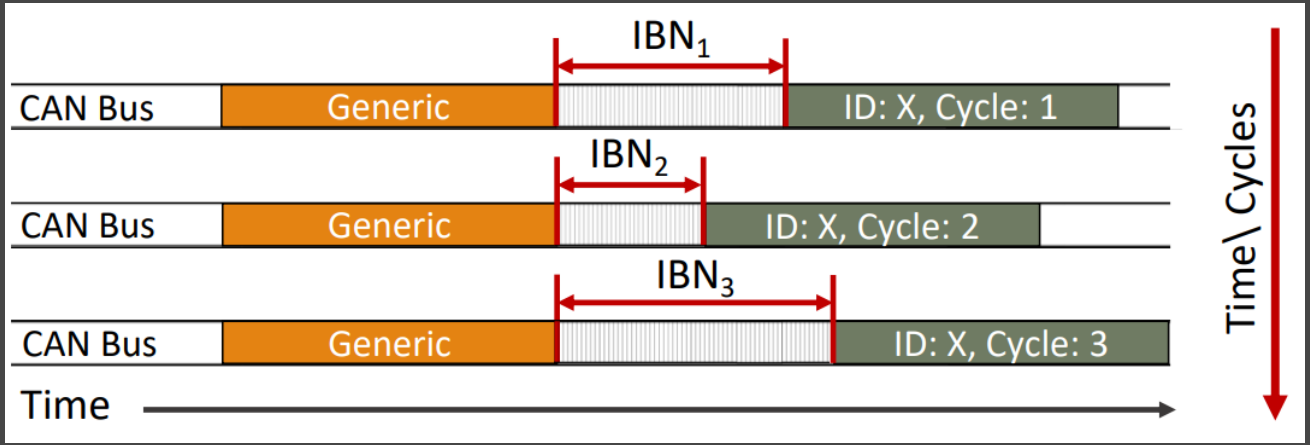
Second, if IBN_{sc} is too low, an ECU with low computational power may not have enough time to initiate transmission in time but after an overhead period (T_O).

T_O should be measured empirically for every system.

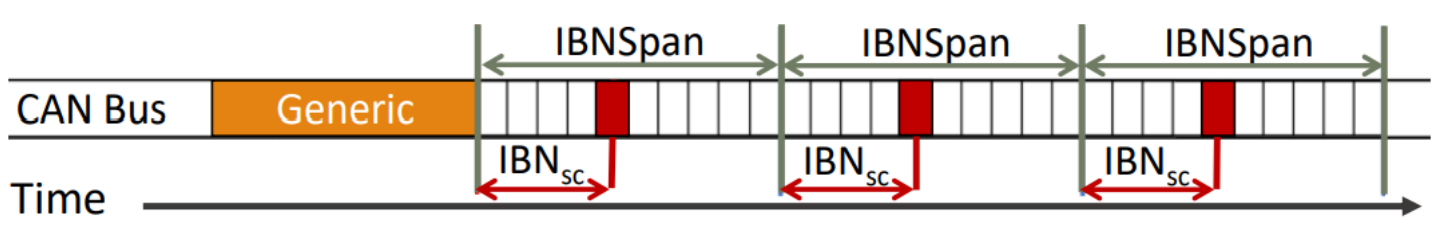
Officer and Agent

- Officer
  - GPIO channel serves three purposes
    1. accurately measuring the IBN of every message
    2. reading message IDs before their data is delivered
    3. allowing the officer to inject error frames on demand to stop any message.
- The agent's role is to apply the IBN sequence upon outgoing messages.

The In BetweeN (IBN)



IBN Implementation Details



If the bus is busy and IBN_{sc} is too long, the agent may never find the opportunity to transmit. To prevent this, all IBN values should be kept within a span ($IBNSpan$) so that any message with $IBN_{sc} \in IBNSpan$ is guaranteed to transmit within a window not exceeding its deadline.

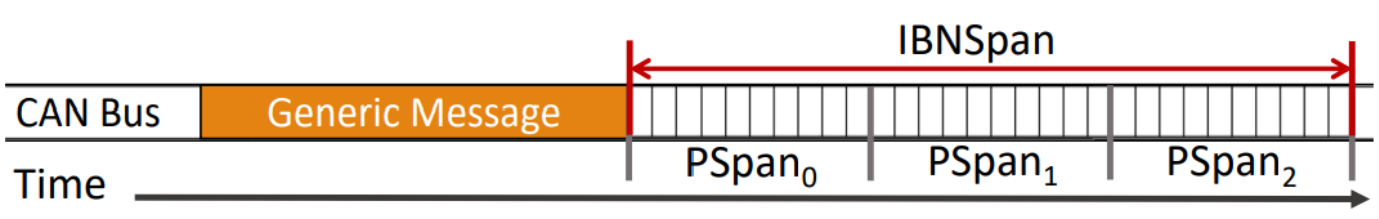
IBN Implementation Details

ID 0, IBN 10 vs. ID 10, IBN 0

ID 10 will transmit first, inverting the priority system.

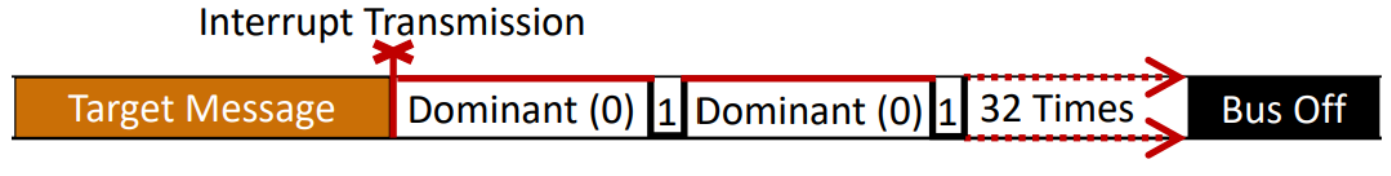
First, we divide $IBNSpan$ further into N_{pri} non-overlapping ranges called priority spans ($PSpans$), each representing one priority level as shown below.

Next, we arrange all message IDs in ascending order, based on their deadlines, then group them into $N_{pri} \geq 1$ priority groups (P_{groupS}).



Disabling Transmitter (Instant Bus-Off)

1. We pick a frame on the bus and wait until a one is being transmitted. Once that happens, we inject a zero.
2. After a single bit, the transmitter detects this error and attempts to send an error frame composed of 6 zeros (flag) and 8 ones (delimiter).
3. After the delimiter starts, we release the bus for a single bit, allowing the one to appear.
4. After the one, we re-inject a zero. Step (2) repeats



6

Security Analysis

Injection and Detection Window

- ZBCAN offers probabilistic security guarantees against injection attacks
- For a periodic message (m) with period (T) and scheduled IBN = IBN_{sc}
- Probability of guessing IBN_{sc} is 1/n (n is | P_{span} |)
- Legitimate ECU sends a message within time period ≤ T with IBN = IBN_{sc}
- Injection detected within a time window ≤ T, except if IBN_{sc} +1 is randomly = IBN_{sc}

$$P_{prevent} = 1 - \frac{1}{|PSpan|}$$

Error Handling Attacks

- With ZBCAN, the attacker cannot randomly inject a high priority message for synchronization or it will be stopped by the officer.
- Further, the attacker has to accurately guess the scheduled IBN for the victim's message.
- Assuming that the attacker only has to guess IBN, Equation could be applied to estimate a probabilistic lower bound for the prevention rate

$$P_{prevent} \geq 1 - (1/|PSpan|^{16}).$$

$$P_{prevent} \geq 1 - (1/|PSpan|^{32}).$$

Flooding Attacks

- ZBCAN prevents flooding attacks by employing the **instant bus-off technique**
- The success of flooding attacks is measured by the drop rate ($rate_{drop}$) of messages
- The prevention rate of flooding attacks is defined as $rate_{prevent} = 1 - rate_{drop}$
- $rate_{prevent}$ varies across different systems based on factors such as busload and network ID allocation

7

Evaluation

Evaluation

- Trusted Officer Platform: A Renesas RA6M5 MCU board was selected as the officer platform, featuring ARM Cortex M-33 architecture and TrustZone technology.

- Pseudo Random Function (PRF): Chaskey, an open-source PRF, was employed. It takes ≤ 0.5 milliseconds to generate computations for Seqlength = 128 b on an Arduino Uno board and approximately 1.9 microseconds on the RA6M5.

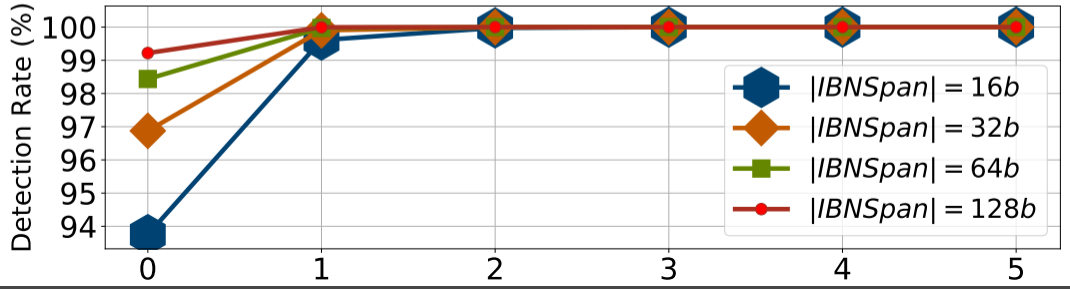
- Zero-point Calculation: The value of T0 was measured on an Arduino Uno, determined to be 7 b. The zero-point is defined as T0 + TSuspend = 15 b after the Interframe Space (IFS).

ZBCAN Security Evaluation on a Testbed

- **System Configuration:**
  1. 500 kbps CAN bus.
  2. Connected components include the officer, 5 ECUs, and a dummy message generator.
  3. ECU composition: Arduino Uno boards, mcp2515 CAN controllers, and mcp2551 transceivers.
  4. One node designated as the attacker.
- **Attack Scenario:**
  1. Smart attacker with knowledge of the system's IBNSpans.
  2. Attacker provided with a modified agent on all nodes to launch attacks.

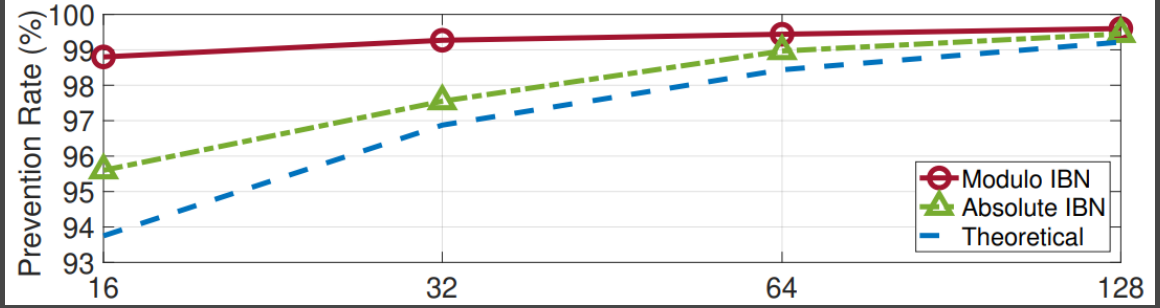
Injection on a Testbed

Attack	Detection Rate	Prevention Rate Per |IBNSpan|			
		16 b	32 b	64 b	128 b
Random Injection	100%	100%	100%	100%	100%
Targeted Injection	100%	93.6%	96.9%	98.5%	99.1%
Replay	100%	93.8%	96.8%	98.4%	99.3%



Error Handling on a Testbed

Attack	Prevention Rate Per |IBNSpan|			
	16 b	32 b	64 b	128 b
Collision Injection	98.8%	99.3%	99.4%	99.6%
Error-Passive	100%	100%	100%	100%
Bus-Off	100%	100%	100%	100%

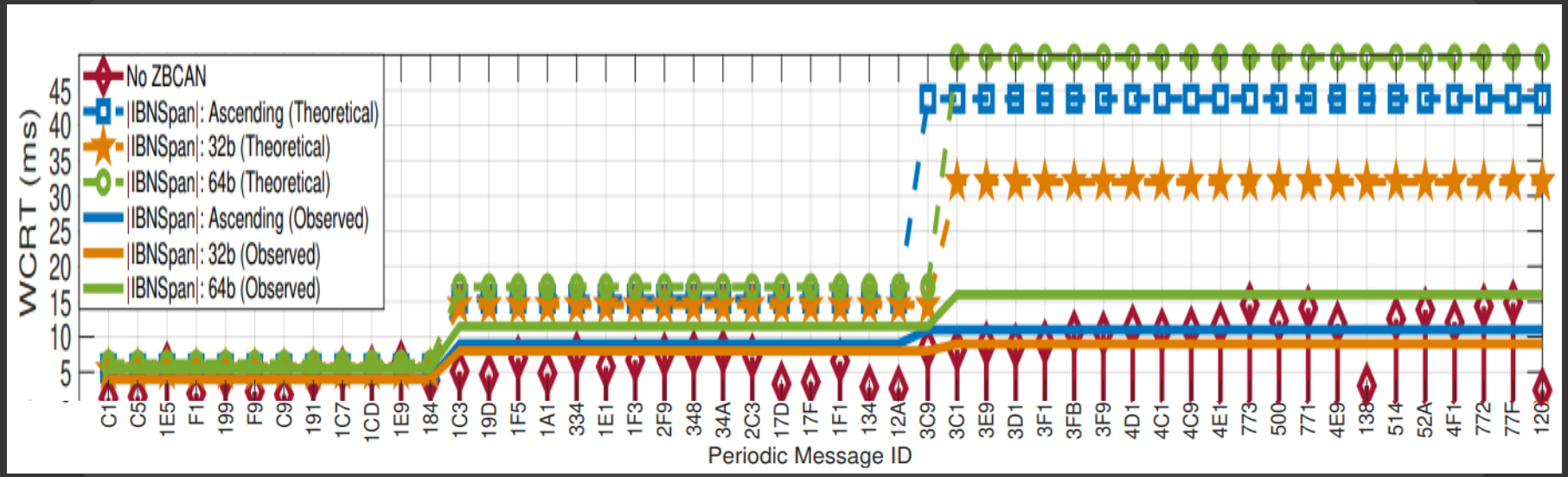


ZBCAN on a Real Vehicle

- Incremental Deployment: Adding a protected ECU to an unprotected bus differs from adding it to an already protected one.
- Message Delays: Messages from the protected ECU experience IBN delays, leading to higher delays compared to an already protected bus.
- Experimental Setup: The officer and a ZBCAN-equipped ECU were connected to the CAN bus of a 2011-Chevrolet Impala.

Setting	Testbed				Chevy-Impala
	10%	20%	30%	40%	
Prevention Rate	100%	100%	100%	99.33%	100%

Performance with Real Vehicle Data



Compared to Other Solutions.

CANARY is one of the few defense systems that addressed error handling and flooding attacks. In addition to the expensive costs of wiring and adding relays, relays work by isola… …s, toge… atta…

Defense System	Response Time	Attacker Isolation	Hardware Changes
CANARY [23]	5ms-100ms	Partial	1 Guardian Node + 8 Relays + Wiring
ZBCAN	22-72 us	Full	1 Officer Node

8

Discussion and Conclusions

Discussion

- Intrusion Confinement in ZBCAN is implemented through two primary mechanisms. Firstly, due to the absence of shared keys or sequences among agents, a compromised ECU agent is unable to predict the IBN sequences of other nodes.
- ZBCAN Controller. CAN controllers have all the hardware required to monitor and change message spacing (e.g., suspend transmission period).
- Content Authentication: In aiming for lightweight design, ZBCAN focuses on transmitter authentication. However, it can be easily extended for content authentication by calculating a hash or MAC for each message and XORing the result with IBNsc.

Conclusions

- We introduced ZBCAN, a novel defense system utilizing inter-frame spacing to safeguard against common CAN attacks without using any message fields or computationally expensive operations such as encryption.
- We introduced a novel and instant way to suspend nodes called the instant bus-off technique, which we used for defense purposes against intruding nodes.
- we proved the applicability of our system by evaluating different aspects of it on a testbed using artificial data, then a testbed using a real vehicle's data, and finally on a real vehicle's CAN bus

Thanks for listening

Q & A