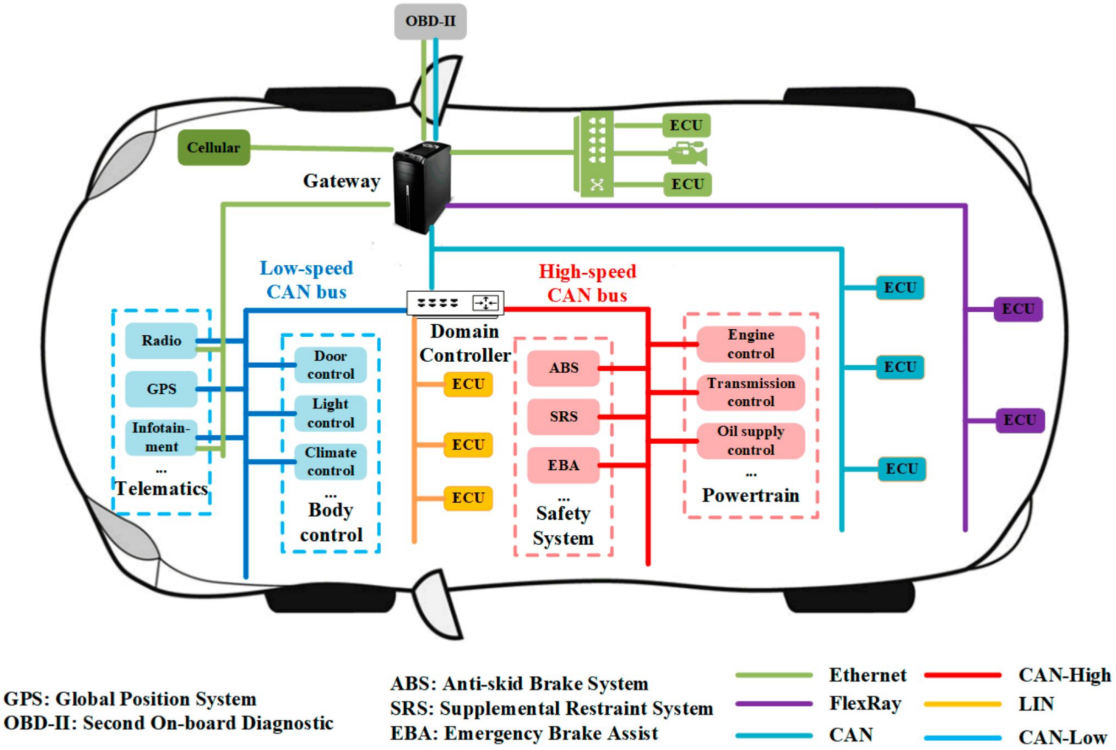
# Intrusion Attacks on Automotive CAN and their Detection

Halley Paulson

# Motivation

**Controller Area Network (CAN)** : controller communication standards and protocols

**Intrusion Detection System (IDS)** : system that monitors host or network traffic and alerts when there’s malicious activity




https://www.mdpi.com/1424-8220/20/17/4900

# Outline

Attacking The CAN

- ➔ Background
- ➔ Exposing the Vulnerability

Securing The CAN

- ➔ Background
- ➔ Identifying Threats

Conclusions

Acknowledgments

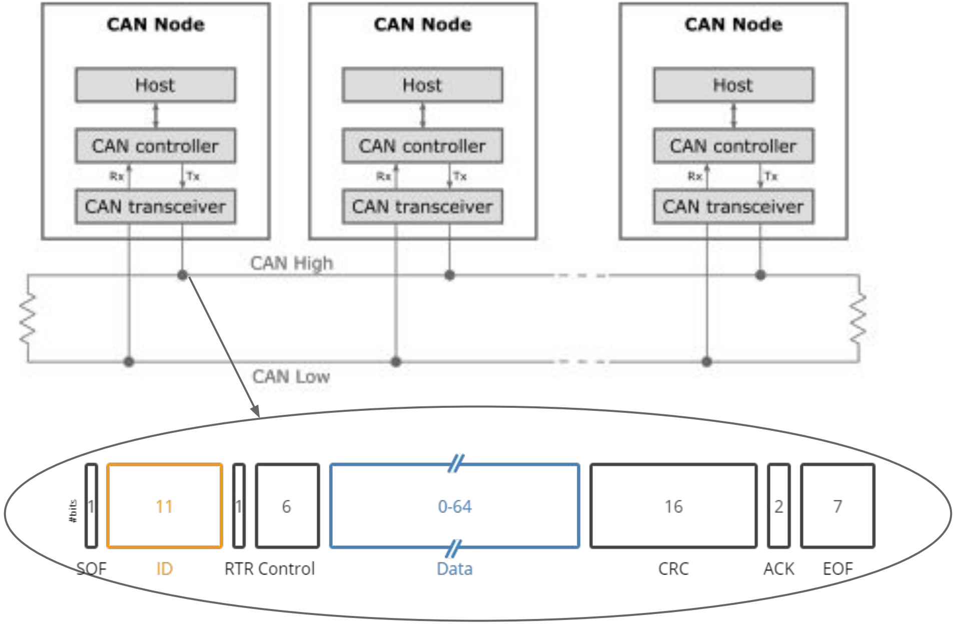
Questions

# Attacking the CAN - Background

# Background

CAN

Developed in 1985!



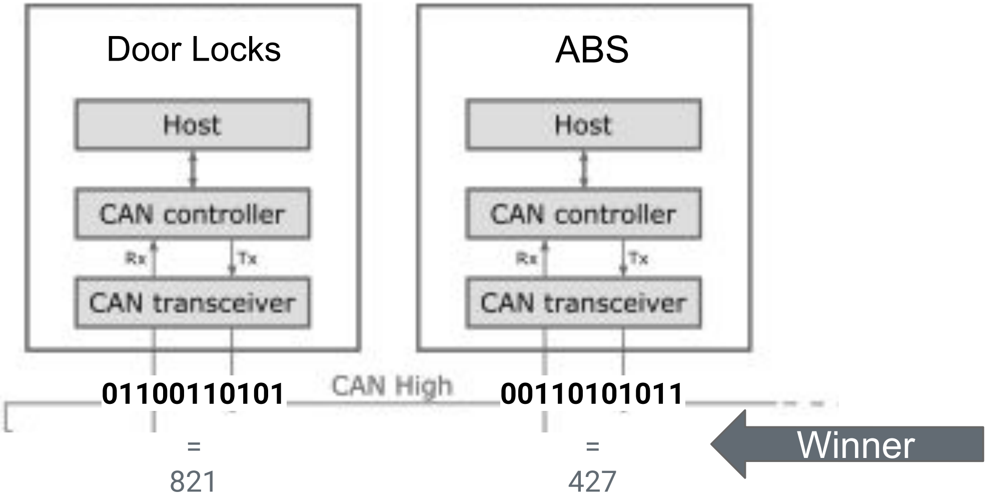

- ➔ CAN nodes send data through frames
- ➔ All CAN nodes broadcast frames in real time
  - ◆ CAN nodes compete for control of CAN bus

# Background

CAN

**Arbitration** : organization of nodes taking and releasing control of CAN bus

- ➔ Compares Message Priorities
  - ◆ Initial bits of a frame
- ➔ Lower Message Priority wins



# Background

CAN

Error states manage CAN nodes to reduce errors in network

➔ Transmission Error Counter (TEC)
➔ Received Error Counter (REC)

Possible error states

➔ Error Active [ $TEC \leq 127$ ]
➔ Error Passive [ $127 < TEC < 256$ ]
➔ Bus Off [ $TEC \geq 256$ ]

# Background

## Fault Injection

**Fault Injection** : verification technique to test system response to faults by inducing them

- ➔ Bad actors employ technique maliciously
- ➔ Software and hardware injectors
  - ◆ Software tests code and protocols
  - ◆ Hardware tests behavior of physical parts

# Attacking the CAN - Exposing the Vulnerability

# Exposing the Vulnerability

Critical Issues contributing to vulnerability

- ➔ Too low-resource for encryption
- ➔ Previously assumed to be impenetrable
  - ◆ Fault injectors can communicate with CAN
  - ◆ Vehicles have wifi

CAN mechanisms can be used against the system

- ➔ Frames are manipulated to
  - ◆ Abuse arbitration
  - ◆ Forcibly change CAN node error states

# Exposing the Vulnerability

Artificial faults are bits injected into frames strategically

Full Bus DoS

- ➔ Continuously send 0's on CAN bus
  - ◆ CAN bus always active
  - ◆ Prevented CAN nodes from sending frames

Directed Bus DoS

- ➔ Injected 0's into frame's data segment until Bus Off state
- ➔ Injecting into Message Priority segment blocks CAN node

# Securing the CAN – Background

# Background

IDS

Basic IDS types

**Online** vs Offline

- ➔ Immediately notify

**Network** vs Host

- ➔ Monitor traffic for entire network

**Anomaly-based** vs. Signature-based

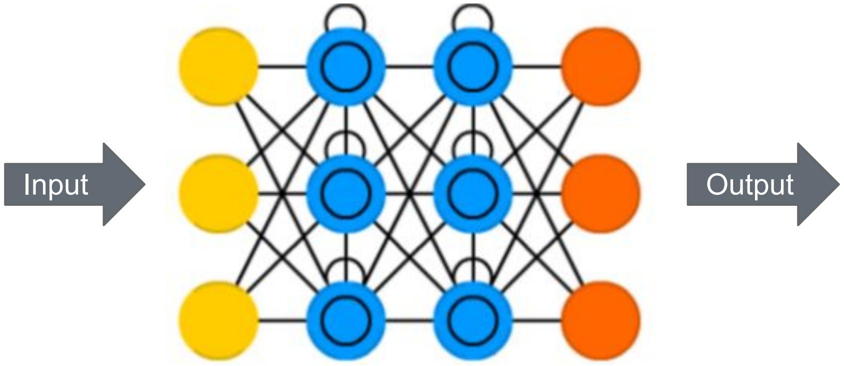
- ➔ Learn and predict based on normal network behavior
- ➔ Great for new anomalies
- ➔ High False Positive Rates (FPR)

# Background

## Recurrent Neural Networks

**Recurrent Neural Networks (RNNs)** : type of machine learning model

- Formed by layers of cells
  - Weights connect cells
  - Activation functions in each cell introduce non-linearity
- Trained to predict sequential data
  - Maps inputs to predetermined outputs
  - Weights are adjusted
  - Loops previous inputs in cell's hidden state

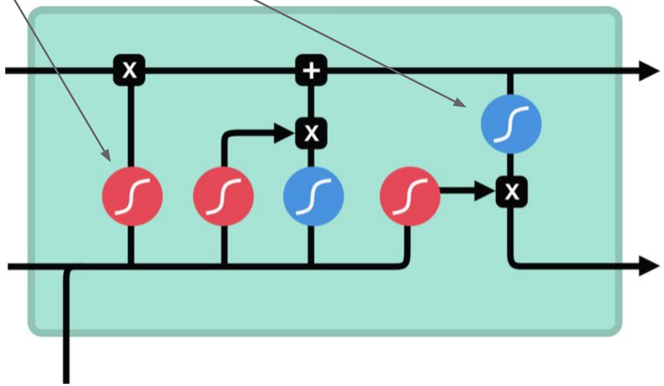


# Background

## Long Short-Term Memory Networks

**Long Short-Term Memory Networks (LSTMs) :** type of RNN

- ➔ Cell state and output in each cell controlled by gates
  - Forget gate
  - Input gate
  - Output gate
- ➔ Gates go through activation functions with inputs
  - tanh()
  - sigmoid()



https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21?gi=78610a07adca

# Securing the CAN – Identifying Threats

# Identifying Threats

IDS comprised of two main engines

- Anomaly Detection Engine
  - **LSTM Prediction Algorithm**
    - Given a stream of network data, predict anomalies
  - Flagged anomalies go to Decision Engine
- Decision Engine
  - Consumes anomaly patterns
  - Alerts network

# Identifying Threats

Formulas created to train LSTM on multiple CAN measurements

- Data came from 10 cars all driving same route for 35-45 minutes
  - Each measurement represents a CAN node
  - Measurements were recorded frames

**Formula A**

$$\frac{EngineSpeed}{AcceleratorPedalPosition}$$

**Formula B**

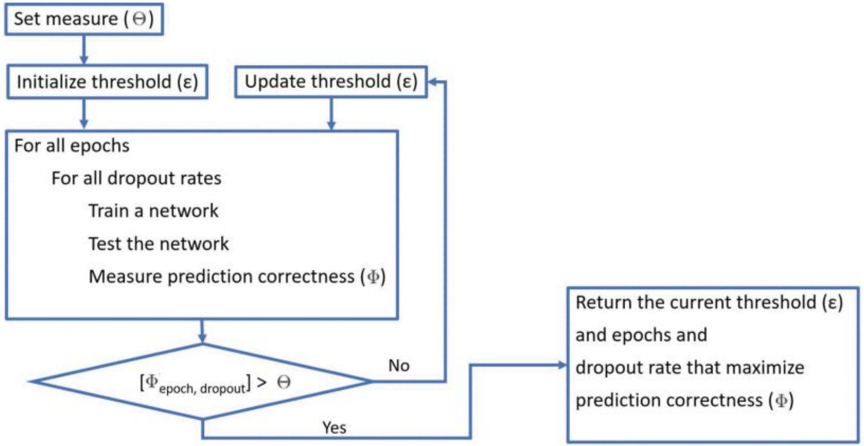
- Pearson Correlation Coefficient : *corr(a,b)*

$$+\frac{x_3}{x_4}+corr(x_5,x_6)+$$

$x_3$ = wheel speed  $x_4$ = current gear  $x_5$ = lateral acceleration  $x_6$ = steering angle

# Identifying Threats

**Hyperparameter** : parameter used to control the training process




Best Values

- ➔ Epochs = 100
- ➔ Dropout Rate = 20%
- ➔ Threshold ($\epsilon$) = 0.3

# Identifying Threats

Using Formula A dataset

- ➔ How accurately can LSTM detect anomalies?
- ➔ Does the % of anomalies in dataset effect accuracy?
- ➔ How does performance vary from car to car?

Summary of Results

- ➔ Test had 1% of Formula A dataset made anomalous
  - ◆ Anomalies created by tripling Engine Speed

| | Accuracy | FPR |
|---|---|---|
| Car 1 | 0.9855 | 0.0140 |
| Car 2 | 0.9864 | 0.0126 |
| Car 3 | 0.9780 | 0.0209 |
| Car 4 | 0.9789 | 0.0202 |
| Car 5 | 0.9816 | 0.0168 |
| Car 6 | 0.9864 | 0.0124 |
| Car 7 | 0.9807 | 0.0189 |
| Car 8 | 0.9868 | 0.0118 |
| Car 9 | 0.9802 | 0.0188 |
| Car 10 | 0.9803 | 0.0187 |

| | | Actual | |
|---|---|---|---|
| | | Malicious | Normal |
| Predicted | Malicious | 14 | 28 |
| | Normal | 3 | 1642 |

Confusion Matrix for Car 5 results

# Identifying Threats

Using Formula B dataset

- ➔ How does LSTM perform when three CAN measurements are changed at the same time?
- ➔ Does anomaly placement in dataset alter performance?
- ➔ Does LSTM perform better with Formula B compared to Formula A?

Summary of Results

- ➔ Performed better with Formula A in similar tests
  - ◆ Formula B represents realistic attack
- ➔ Anomaly placement boosts performance
  - ◆ With adjacent placement, Formula B performed better

# Conclusions

# Conclusions

CAN vulnerability

- ➔ CAN protocols aren't designed to handle cyberthreats
  - Protocols can't tell difference between faulty behavior or malicious activity
- ➔ Newer vehicles at higher risk
  - Poses a data risk but also a health risk
  - Could be worse than DoS attacks

Proposed IDS

- ➔ Can help CAN identify attacks
  - Adjustments to CAN could be made
- ➔ Accurate predictions and acceptable FPR in most tests
  - Formula B more resilient to realistic attacks
- ➔ Still in development but a promising solution

# Acknowledgments

A huge thank you to Elena Machkasova for her guidance and friends and family for their support.

# Questions?

# References


S. Hajj, R. El Sibai, J. Bou Abdo, J. Demerjian, A. Makhoul, and C. Guyeux. Anomaly-based Intrusion Detection Systems: The requirements, methods, measurements, and datasets. 2021.

P.-S. Murvay and B. Groza. DoS attacks on Controller Area Networks by fault injections from the software layer. 2017.

M. Phi. Illustrated guide to LSTMs and GRUs: A step by step explanation, Sep 2018.

M. Phi. Illustrated guide to Recurrent Neural Networks, Jun 2020.

G. Rodriguez-Navas, J. Jimenez, and J. Proenza. An architecture for physical injection of complex fault scenarios in CAN networks. 2003.

V. Tanksale. Anomaly detection for Controller Area Networks using Long Short-Term Memory. 2020.

C. Watterson. Controller Area Network (CAN) Implementation Guide. 2017.