Towards CANLay: An X-in-the-loop Virtual Testbench for In-Vehicle Security Testing with Real-time Network Performance Monitoring

Abstract

Your abstract text goes here. Just a few facts. Whet our appetites. Not more than 200 words, if possible, and preferably closer to 150.

1 Introduction

In recent years automotive security has been a much talked about topic of research. Researchers [2] have shown that remote interfaces on modern passenger vehicles can be used to intrude into the in-vehicle network of embedded controllers, also referred to as Electronic Control Units (ECU). MHD vehicles expose similar interfaces that can be used to control/disrupt critical functions [1, 4] typically operated by ECUs using sensors and actuators. To evaluate the effectiveness of their approaches, researchers have traditionally experimented on real-vehicles or homegrown testbed setups that mimic real vehicles. While most households in the United States have at least one passenger car[1], this is not the same for medium and heavy-duty (MHD) vehicles. Moreover, creating homegrown testbeds is both logistically and economically challenging. To that end, the need for a publicly accessible testbench is imminent.

There are two important criteria that reseach in this area has established for this type of testbench. The first is fidelity, i.e. the ability of the setup to replicate a real-world in-vehicle networking infrastructure. The second is adapability i.e. the ability to be reconfigured to suit different needs. It may be difficult to maximixe the extent to which both these criteria is achieved. The most adaptible testbed is the one in which ECUs as well as the network configuration can be programmed on the fly. Existing solutions have enabled ECU virtualization but not network virtualization for real-world ECUs. In those setups, ECUs from different physical locations cannot be used in the same testbed, neither can networks be configured on demand, unless the ECU is virtualized. We believe that having real-world ECUs in the testing setup is critical. This not only provides greater fidelity but also alleviates any concerns with intellectual privacy and availability on the ECUs. Another aspect to fidelity and adaptibility is the realization of vehicular conquality of the network Existing solutions

In this

2 Related Work

3 Design Goals

The overarching design goal of this project is elucidated through figure 2. To design a platform that simulates a CAN internetwork within a running vehicle using ECUs from separate subnets within the internet a. la. the software-defined truck [3]. Clearly, this has to be achieved over in-vehicle communication overlays on top of protocols of the internet.

[1] https://www.statista.com/statistics/551403/number-of-vehicles-per-household-in-the-united-states/

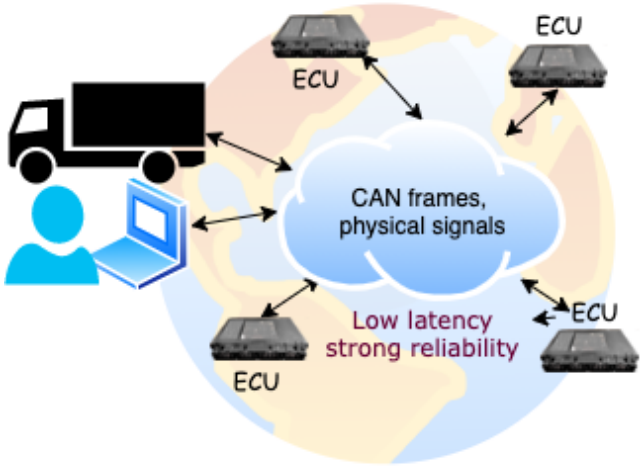

Figure 1: Design Goal of CANLay

The distributed overlay nature of the project introduces a set of fidelity-related challenges when emulating the tightly integrated infrastructure within the vehicle. Firstly, CAN provides a low-latency, strong reliability communication media that may be difficult to emulate over long-range communication channels. This requirement is even stronger for the physical signals that are usually transmitted over direct wiring in a real-world setting. Secondly, the broadcast nature of CAN may be difficult to realize in a distributed manner over the (typically) unicast internet protocols. Using unicast packets for multi/broadcast may require duplication in linear time. This is both space and time-intensive, especially if performed in a smaller scale local network that has limited bandwidth. The final challenge is to realize the complexity of interactions between the different components of the vehicle that normally operate in the same physical setting.

Albeit, all of these challenges can be difficult to realize in full in an adaptability-centric setting like ours. Even so, a major goal of this project is to optimize and report the quality of experimentation, thereby creating a transparent and usable environment for the experimenter.

4 Design

ECUs in our system are required to interface with Smart Sensors Simulator (SSSF) devices. These devices act as gateways to the CANLay system. At startup they communicate with a central server thereby registering themselves. This communication happens over HTTP(s).

4.1 Current Status of Development

4.2 Component Descriptions

4.2.1 Smart Sensor Simulator and Forwarder (SSSF)

4.2.2 Controller

4.2.3 Server

4.2.4 Vehicle Simulator

4.2.5 Publish/Subscribe Endpoints

4.2.6 User Interface to CAN

4.2.7 Network health monitor

4.3 Behavior Descriptions

4.3.1 Overlay Setup

4.3.2 X-in-the-loop Simulation

4.3.3 CAN communication

4.3.4 Network health monitoring

A health

5 Analysis

References

[1] Yelizaveta Burakova, Bill Hass, Leif Millar, and Andre Weimerskirch. Truck Hacking: An Experimental Analysis of the SAE J1939 Standard. In *Proceedings of the 10th USENIX Conference on Offensive Technologies*, pages 211–

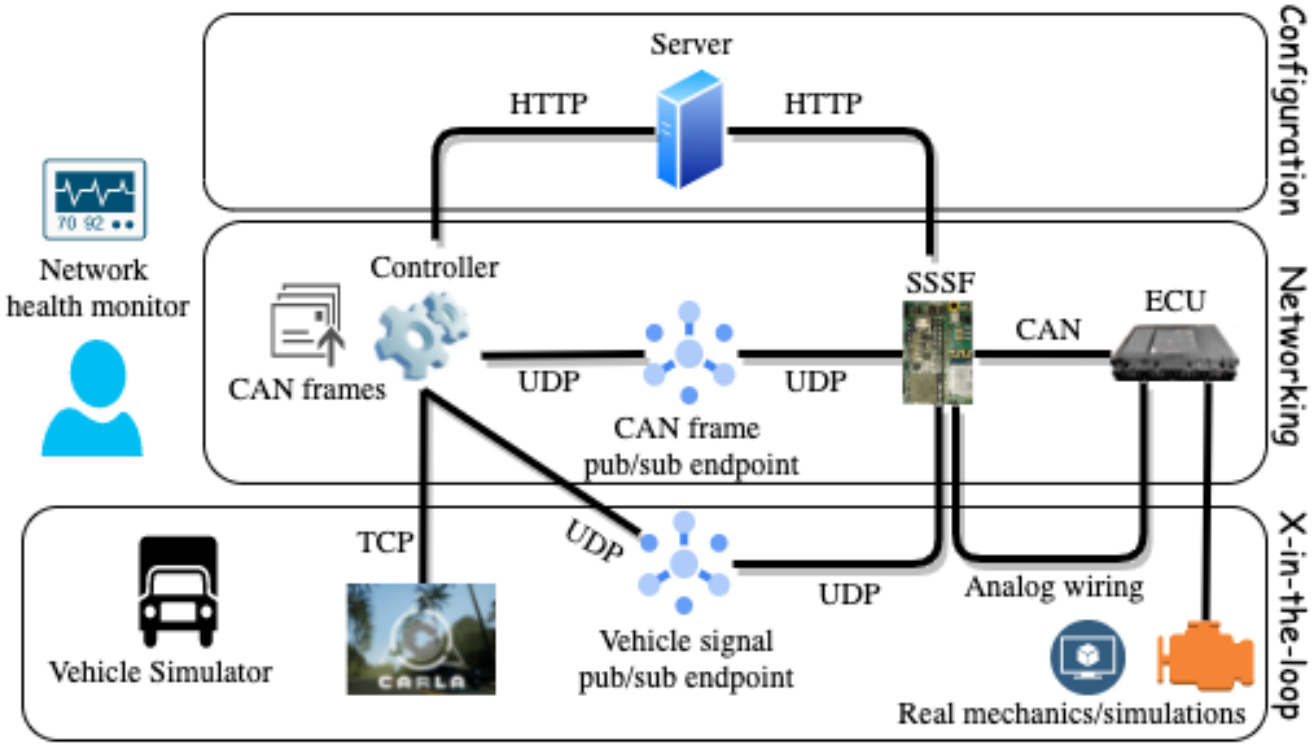

Figure 2: Proposed System

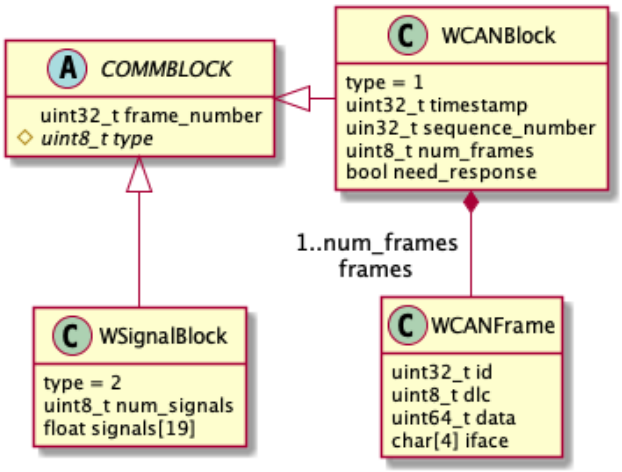

Figure 3:

220, Austin, TX, USA, 2016. USENIX Association.

[2] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and Tadayoshi Kohno. Comprehensive Experimental Analyses of Automotive Attack Surfaces. In *USENIX Security Symposium*, volume 4, pages 447–462, San Francisco, CA, USA, 2011. USENIX Association.

[3] Subhojeet Mukherjee and Jeremy Daily. Towards a Software Defined Truck. In *Proceedings of the 31st Annual INCOSE International Symposium*, page 16, Online, 2021. INCOSE.

[4] Subhojeet Mukherjee, Hossein Shirazi, Indrakshi Ray, Jeremy Daily, and Rose Gamble. Practical DoS Attacks on Embedded Networks in Commercial Vehicles. In *International Conference on Information Systems Security*, pages 23–42, Jaipur, Rajasthan, India, 2016. Springer.

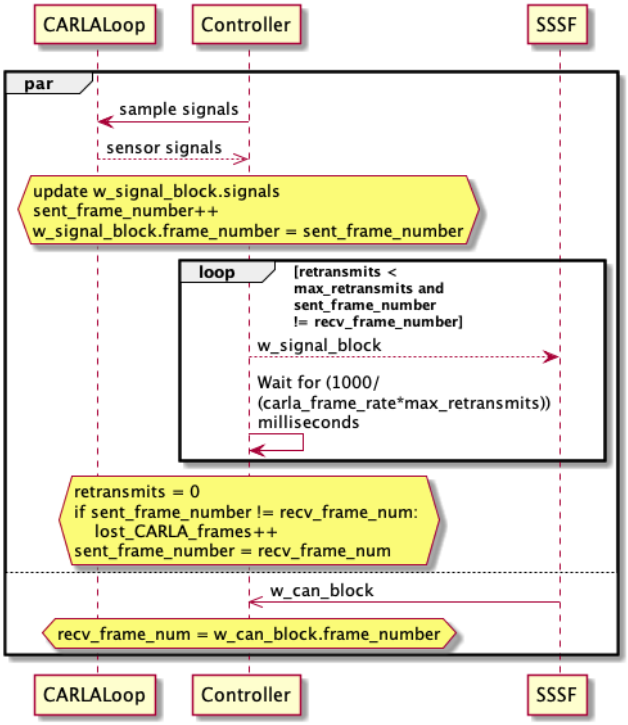

Figure 4:

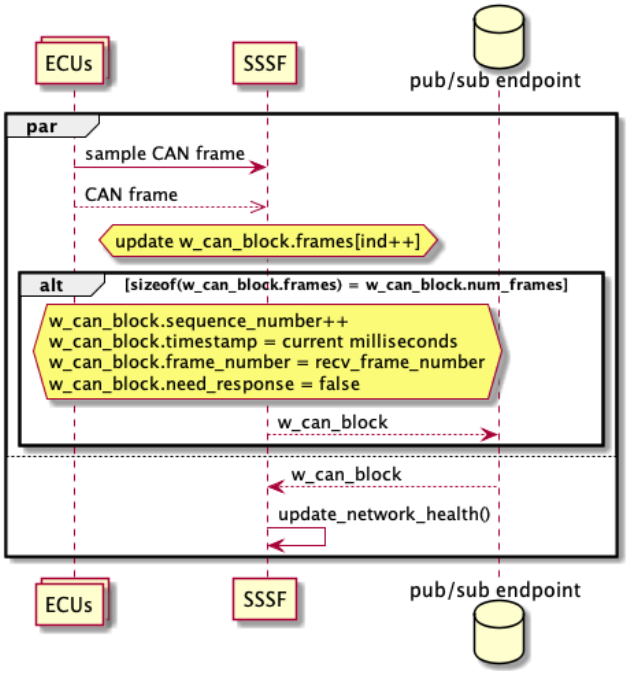

Figure 5:

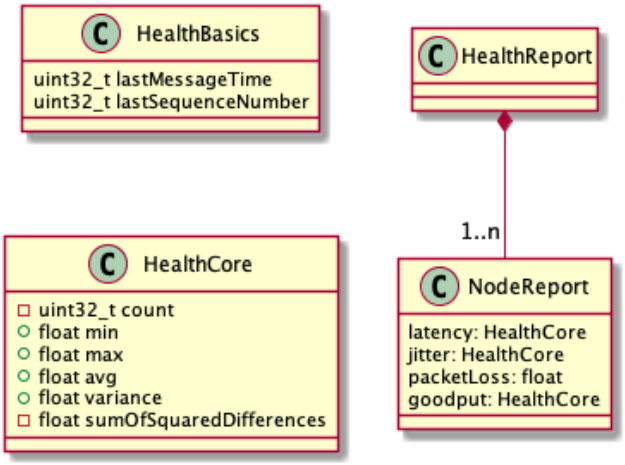

Figure 6: