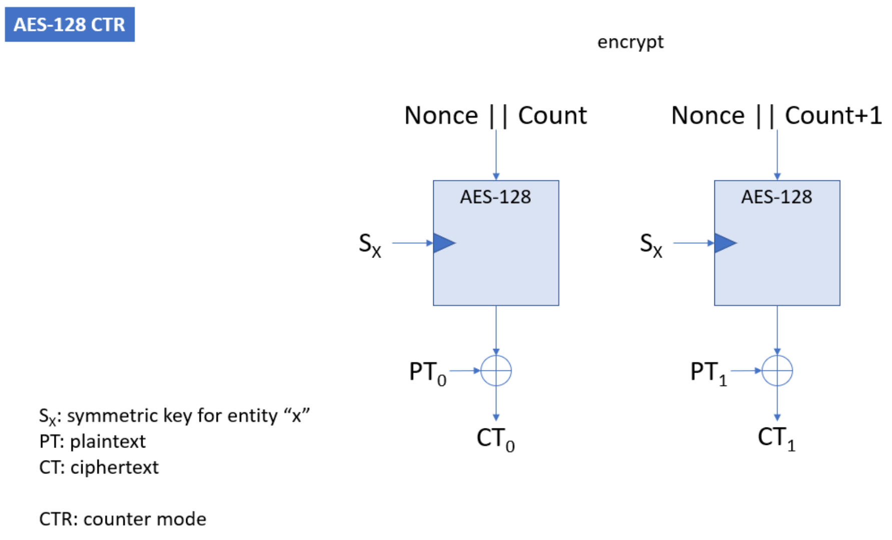
4.3 Proposed Design

4.3.1 Overview

Our current design is to improve the security of vehicles by protecting the data they use to communicate. The vehicle data in question includes things like breaks, transmission, etc. This will be accomplished by adding a small device which reads data traveling through the vehicle, and validates it accordingly. Any malicious or incorrect messages will be caught and thrown out with software that we write for this device.

4.3.2 Detailed Design and Visual(s)

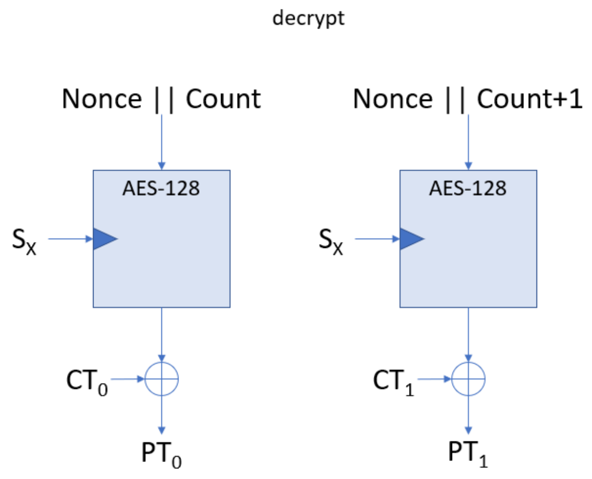
Our current design plan is developing a program for encrypting and decrypting a single CAN frame. Once we have it working for a single CAN frame, we will look to test it on larger inputs of multiple CAN frames and make sure the code consistently performs as expected. Here is a diagram of how AES-128 CTR encryption will work:



The encryption process works by sending in the Nonce, Count, and Symmetric Key to the program to encrypt using AES-128. Then you take the output of that and XOR it with the Plaintext to produce the Ciphertext.

The decryption portion of the program will work very similarly, except for the ciphertext being XOR'ed where the plaintext was in the encryption program to then output the plaintext:

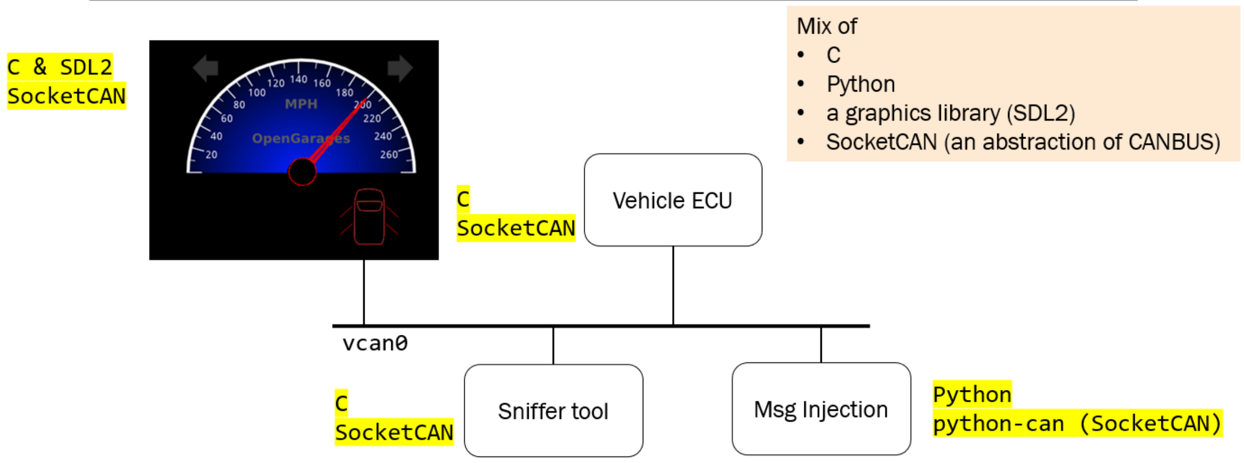
AES-128 CTR



S_X: symmetric key for entity “x”
PT: plaintext
CT: ciphertext

CTR: counter mode

This is a visual of the SocketCAN network and how it communicates with the vehicle. The messages sent can be related to any variety of things like vehicle acceleration, steering, and other functions of a vehicle. The main reason behind our project is that the current network is not very secure and leaves vehicles vulnerable to being hacked, so we are trying to increase the safety of the system.



Mix of
- C
- Python
- a graphics library (SDL2)
- SocketCAN (an abstraction of CANBUS)

4.3.3 Functionality

Our design is intended to operate in the manner that a bad actor won't be able to send repeat data on the CAN bus network, nor their own fabricated instructions. For example, the user will be driving their car, controlling the MPH, steering, brakes, radio. All of these generate signals for the CAN bus to process and execute, but it should do so in a manner that consists of integrity and availability. This means that a bad actor can't send their own instructions to the vehicle CAN network via intrusion because they won't have the "rights" to send data, so instead, it will be rejected by the network and continue operations as usual. Additionally, the CAN network will also not accept repeat data. For example, a person could be snooping on the network traffic being sent on the internal CAN network of a car and grabbing CAN data frames and repeating them into the system. How our device will prevent this is by using a mixture of a counter and freshness value in each data frame packet that is sent to ensure that old packets can't be sent again.

4.3.4 Areas of Concern and Development

The current design that we have implemented is a good starting point for what our finished product will represent. Currently, we are using the AES-128 encryption standard. We believe that this is exactly what we need to satisfy the requirements and the user needs. Our client specifically told us that this is the encryption standard we will be using as the lower bit size allows us to ensure the availability of the data, as this is a much faster encryption speed than 192, 256.. etc. What we are unsure of currently is how we will prevent repeat data right now. We are figuring that we can use the expanded CAN FD frame to allocate certain bits to be "time stamps" or "freshness values" that will help prevent repeated instructions.

The immediate plans for developing the solution to this concern is researching more into the exact bit fields of the CAN FD frame and verifying this, or identifying if some other solution will arise out of the provided documentation for the bit fields.

We don't have any questions as of right now.

4.4 Technology Considerations

The technologies that we are currently using include, the C programming language, the emacs text editor, and a CAN BUS simulation program. The advantage of using C is that there are a vast number of libraries for our use case. For example, because we are using AES-128 bit encryption, we found a library in which to build this program. Also, being that C is a lower-level language, we have more control over the lower-level details for our program. Next is the emacs text editor, which has the advantage of being lightweight, and easy to use. Finally is the CAN BUS simulation program, which we are using in order to delve into how an actual attack into a CAN BUS network might happen.

One of the outstanding disadvantages to using C is the learning curve associated with it. Being that it is a lower-level language means it is harder to adjust the mindset needed when using it. As for the text editor, although it is lightweight, that means there aren't as many features associated with it. One of the more noticeable downfalls is the lack of a debugging tool. Even though these weaknesses exist, the advantages outweigh said disadvantages.

4.5 Design Analysis

Currently we have found a potential candidate for our cryptography. The library is called OpenSSL and will allow us to use AES-128 with C. It's a very vast tool kit with a lot of documentation online which will hopefully mean that many questions that may come up while we get a closer look at its uses will already be answered. As of now one member of our team has experimented with the toolkit and has reported its uses.

We have also individually set up our virtual machines that we intend to use for virtual testing of our code. We intend to begin experimenting with code on these virtual machines soon, but as of now there is nothing to report in terms of issues with getting set up and ready to go.

We are still keeping in touch with our advisor, John Potter, and receiving helpful information from him regarding our project about every week. Currently our proposed design from 4.3 is going smoothly and there are no glaring problems that we have had trouble solving.