2 Project Plan

2.1 Project Management/Tracking Procedures

For this project, our team is planning on adopting the waterfall model rather than the agile method since this way we can help each other with implementation processes, rather than having a single person or two on various things going at once. The reason we are going with this approach as compared to agile, is that this group is not cyber oriented, so working together on new concepts will work best.

How we will track our progress as a team is by using Git, Github, and Git Milestones, so we can track who is contributing what, when, and what people are currently working on. And then, we have our weekly reports due on Fridays.

2.2 Task Decomposition

Lists of tasks may include but not be limited to:

- Choose programing language
- Find a crypto library that works best with AES-128 and Python.
- Setup virtual testing environment
- Figure out how to simulate CAN data for testing.
- Distribute workload amongst team members, finding strengths to ensure everyone is comfortable with their own goals
- Figure out how to upgrade a CAN frame into CAN FD for an expanded byte-size frame for security implementations.
- Individually familiarize with chosen language syntax and development practices.
- Understand how to encrypt and decrypt CAN frames using chosen language
- Get hands on physical box
- Run tests on physical vehicle

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

All tools have been researched and decided (crypto library, programming language, etc.)

All members of the team feel comfortable developing functions and logic in chosen programming language.

All members of the team have a simulated environment of the CAN bus system up and running on their individual machines.

Method for upgrading CAN frames to implement our improved security solution has been discovered.

CAN frame encryption and decryption 10% solved, code structured and rough understanding.

CAN frame encryption and decryption 25% solved, able to encrypt data with some implementation

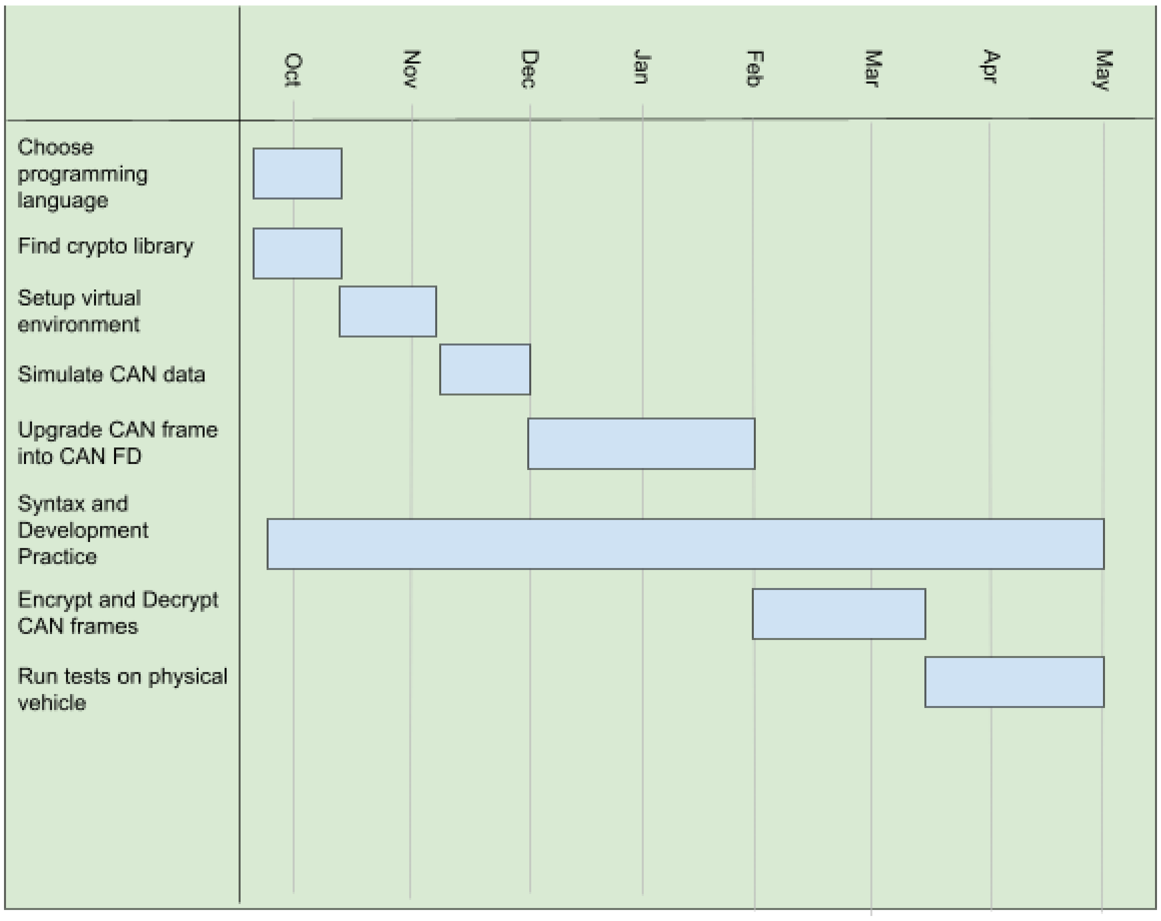
CAN frame encryption and decryption 50% solved, data is running through encryption and decryption cycle with hiccups.

CAN frame encryption and decryption 100% solved, all CAN frames are running through the encryption and being read correctly on the other end

Software has been loaded into device and is functioning properly

Physical device connected to CAN bus of a vehicle and is functioning properly

2.4 Project Timeline/Schedule



2.5 Risks And Risk Management/Mitigation

Task	Risk
Choose programing language	● Not everyone may know or be efficient with the language we decide on as a group. ● We may find out later that X language isn't the most efficient with what our project is.
Find a crypto library that works best with AES-128 and Python.	● Make sure nobody gets left behind in the learning phase so we all understand how the encryption libraries work. We will need to thoroughly comment on the code we write.
Figure out how to simulate CAN data for testing.	● No risk involved, this is a much needed step. A risk would be not doing this early enough.
Distribute workload amongst team members, finding strengths to ensure everyone is comfortable with their own goals	● Burning out team members of what they are good at if they are doing the same repetitive thing.
Figure out how to upgrade a CAN frame into CAN FD for an expanded byte-size frame for security implementations.	● Misunderstanding the bit fields of CAN FD, or overwriting data into important bit fields.
Individually familiarize with chosen language syntax and development practices.	● No risk, this is a much needed step.
Understand how to encrypt and decrypt CAN frames using chosen language	● No risk, this is a much needed step.
Get hands on physical box	● Finding the right size of box with the right chip component that we can flash our embedded system program onto.
Run tests on physical vehicle	● We will have to be careful with testing the vehicles as they're multi hundred-thousand dollar machines.

	Note for teacher: Our client said he would love it if we could bring in a tractor and run this demo on.

Risk mitigation for “Run tests on physical vehicle”: We will need to be thorough with the testing of our simulation on the laptop before we sit down with a physical vehicle. Must cover edge cases as we don’t want to mess up a very expensive piece of equipment.

2.6 Personnel Effort Requirements

To estimate these numbers we took into consideration that we have 7 people on our team so a 1 hour long team meeting would equal 7 person-hours.

Task	Time
Choose programing language	7 hours
Find a crypto library that works best with AES-128 and Python.	14 hours
Figure out how to simulate CAN data for testing.	35 hours
Distribute workload amongst team members, finding strengths to ensure everyone is comfortable with their own goals	14 hours
Figure out how to upgrade a CAN frame into CAN FD for an expanded byte-size frame for security implementations.	35 hours
Individually familiarize with chosen language syntax and development practices.	21 hours
Understand how to encrypt and decrypt CAN frames using chosen language	28 hours

Get hands on physical box	42 hours
Run tests on physical vehicle	56 hours

2.7 Other Resource Requirements

Outside of financial resources, other resources would include our own computers (laptop or pc) to make the project from scratch on an IDE with various libraries, test, and simulate the project. We will also use the CAN network for our Security Bridge. We have our client who has provided us with plenty of information on our project. And finally, we will be testing everything using a farming tractor that our client has provided once the project is able to be tested.