# Report Outline

Architecture

* Our program is divided into 3 threads. The main thread, the timer thread, and the server thread.
  + Server Thread: This thread opens a UDP server and listens for packets. When a new packet comes in, it is pushed on to a queue that the main thread can pop from.
  + Main Thread: Runs a main infinite loop that constantly checks the packet buffer. If there is something in the buffer it takes it out and processes it. This includes forwarding it via Reliable Broadcast Algorithm rules, updating other nodes positions, and using that packet to determine if we need to speed up / change lanes / slow down / or join a platoon.
  + Timer thread. This thread waits for some small amount of time and then sets a Boolean to true. Our main thread will only transmit a status packet to all it’s neighbors if this Boolean is set to true. After status packets have been sent, it sets the Boolean to false. This allows us to constantly process packets while only sending a status packet every so often.

Design

* Here are some design details on how the program works
  + How is a simulation set up?
    - When you start the program it checks to see if a Config file already exists. If the file does not exist, this program knows it is the first node to join and is therefore the truck. If the file does exist, the node is set as a car and reads in the other node information from the Config file. Once the node reads the config file, it sends out an initialization UDP packet that tells the other programs a new node is joining. Then the joining node will wait and listen for status packets. Once it has received a status packet from every node, it can safely find a place to enter the highway.
  + How do we update the positions of every node?
    - Every node (node means car/truck) sends an “admin packet” to every other node regardless if it is in range. This packet is guaranteed to send. When a node receives one of these packets it updates it’s Data Structure that keeps track of every node’s position. This packet is flagged and is not forwarded through the RBA algorithm, it is simply used to update the “simulator’s data structure.”
  + How do we write output data?
    - Apart from the standard out text you see on the screen, the Truck node will write to an Output.txt file the current locations and connections of every node. This way we can view the current locations of every car at different times throughout the simulation by typing “cat Output.txt” in the terminal.

Implementation Issues

* We had a lot of issues with the file writing. Having more than one program reading and writing to a file causes all kinds of issues.
  + This was tolerable when dealing with a computer with one core. But once we tried it on a multi-core system, everything went to hell.
  + To fix this, only the truck writes to the output file. The cars only write to the config.txt file when they are first entering the simulation. So the chance of two cars writing at the same time is impossible as long as two car programs are not started at exactly the same time.
  + Instead of cars reading a config file all the time. They only read it when they first enter the simulation, and the only info they extract from the config file is the other node’s hostnames and port numbers.

Introduction / Motivation

* Basically talk about how cars will one day be mandated to have VANET systems built into them. This will make driving much safer. But this project explores some of the problems and challenges that come along with implementing a VANET system.

Problem Statement

* This can be the last sentence of the intro.

Algorithm Details

* How do you enter a platoon? What are the road rules?
  + See the platoon rules document. That could be cleaned up and used as appendix material
* RBA Details are in the report. Not sure we need to talk about that.

Experimental Performance Results (Will come later)

* Must include graphs of: (for different number of cars)
  + Average throughput
  + Packet loss
  + Latency

Conclusions

* The Reliable Broadcast Algorithm sends way too many copies of every packet to every other node.
* After about 5 cars our programs start to get bogged down and sometimes display unpredictable behavior. They are just too many packets being received/sent by each node due to the RBA algorithm.