Access Control in VANET
Minor Project 1

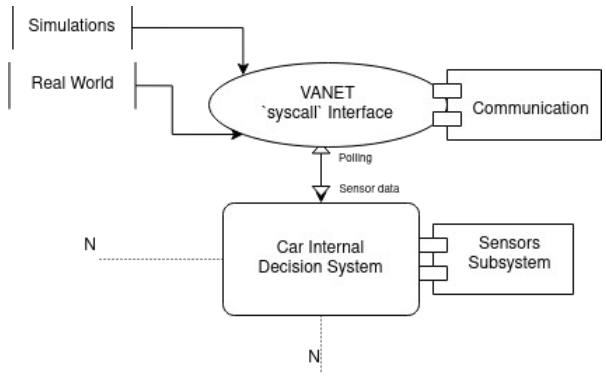
Abhay, Aditya, Rabin (CSE-1, 2019-23)

Mentor: Dr. Bhaskar Mondal, NIT Patna

This describes our initial idea till now, and maybe vague at points. Including claims that are not our goal, but included as interesting for future consideration

Minor Project topic is at section 4

1. Overview & Design



Basically, VANET interface will have some function calls, which I call 'syscall' (named after System calls in Operating Systems) interface.

It allows:

1. Modular Interface
2. Adding/Removing willl be easier
3. The design allows, for minimising some layers of abstraction, even custom Operating System should work as long as they follow the interface, thus increasing available processing power

2. Car Internal Decision System

At minimum, it will require minor changes atleast to incorporate VANET, such as providing sensor data to VANET system.

We also suggest using a Decision Tree based (prevents black-box problems) ML model to decide further action based on data from VANET, sensors and other subsystems.

To access VANET subsystem events, the steps are loosely:

1. Initialise VANET system (provide car data etc.)
2. ... Any other actions, then start broadcast using `start_broadcasting()` syscall.
3. Use polling to access VANET events (lesser interval between polling, may increase safety but costlier).

It should also provide a function for VANET to also get sensor data, for eg.

```
/**
* @returns array of {uint64, uint64, byte array}
* First uint64 represents sensor type
* Second uint64 represents sensor UUID or any other unique identifier
* Third is data bytes received from sensor (the internal system may modify it as long as VANET implementation understands what these random bytes mean)
**/

array(uint64, uint64, byte[]) get_sensor_data();
```

3. VANET syscalls

The VANET subsystem should provide functions (aka. syscalls) like:

```
void    set_payload(byte[]);
void    set_broadcast_rate(float);

/* should have vehicle information according to an agreed standard format */
void    set_broadcast_header(byte[]);
void    start_broadcast();
void    stop_broadcast();

/*This `event` type depends on the implentation, though we suggest it to be an integer, like in operating systems, where they return a file
```

```
descriptor instead of a platform specific file
object*/
event[] poll_events();
```

This way, broadcasting may internally be using Bluetooth Low Energy (BLE), WiFi, or any other tech, and will require minor or no change in car's system which is using this syscall interface.

> VANET subsystem should do internal state-keeping when required, for eg. an event queue, which will store events until he car system uses poll_events() to get them.

> We are using broadcasting to send data to others & polling to get data from VANET subsystem, these maybe costly if we see on processing power, though embedded systems in vehicles should be enough to do these.

> Payload may/may not include sensor data

The modular design allows for simple modifications for use in simulations.

3.1 VANET routing protocol

We are going to use already available protocols such as the reactive & topology drive AODV protocol [#2], or reactive & cluster-based LORA-CBF protocol [#3]

> Prefer cluster-based

When using cluster-based protocol, in a cluster, only vehicles moving in same direction should be considered, it also helps since relative speed being lesser, the connection can be better with vehicles moving in same direction.

Single vehicles will be their own clusters.

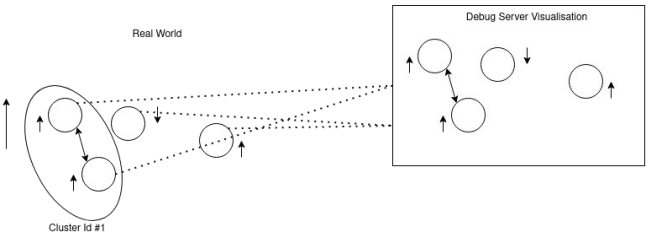
4. Security & Access Control

1. NOW [#4] and SEVECOM [#5] as a baseline for improved security
2. Public key signatures while communicating [IEEE 1609.2] (similar paper - [#6])
3. To help privacy, two nodes in a cluster may 'swap' their identifier/key and then re-register with their respective clusters (similar paper - [#55])

5. Debugging Interface

All nodes send their status to a server, data being something like:

```
{
  cluster_id,
  my_data     // may include GPS location, etc.
}
```



Each node sends those data to some remote server, and remote debug server can visualise it using, say graphs. With nodes with same cluster id the debug server can assign the node to a specific node.

References:

1. Vehicular Ad-Hoc Networks (VANETs) - An Overview and Challenges; Journal of Wireless Networking and Communications 2013
2. C. E. Perkins and E. M . Royer, "Ad-hoc On-Demand Distance Vector Routing," in Second IEEE Workshop on Mobile Computing Systems and Applications, 1999, pp. 90–100
3. R. A. Santos, R. M. Edwards, and L. N. Seed, "A Location-Based Routing Algorithm for Vehicle to Vehicle Communication," vol. 00, no. C, pp. 221–226, 2004
4. NOW -Network on Wheels , http://www.network-on-wheels.de/, 2008,[Online; accessed 20-M ay-2013]
5. SEVECOM , " SeVeCom (Secure Vehicular Communication)," http://www.sevecom.org/, 2008,[Online; accessed 20-M ay2013]
6. G. Calandriello, P. Papadimitratos, J.-P. Hubaux, and A. Lioy, "Efficient and robust pseudonymous authentication in VANET," Proceedings of the fourth ACM international workshop on Vehicular ad hoc networks -VANET '07, p. 19,2007
7. D. Eckhoff, C. Sommer, T. Gansen, R. German, and F. Dressler, "Strong and affordable location privacy in VANETs: Identity diffusion using time-slots and swapping," 2010 IEEE Vehicular Networking Conference, pp. 174–181, Dec. 2010