August 2020

Security Evaluation of a Dedicated Short Range Communications (DSRC) Application

Fei Sun
Clemson University, fsun@clemson.edu

SECURITY EVALUATION OF A DEDICATED SHORT RANGE COMMUNICATIONS (DSRC) APPLICATION

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Computer Engineering

---

by
Fei Sun
August 2020

Accepted by:
Dr. Richard Brooks, Committee Chair
Dr. Lu Yu
Dr. Richard Groff

ABSTRACT

Applications using dedicated short-range communication (DSRC) are being developed to prevent automobile accidents. Many DSRC implementations, applications and network stacks are not mature. They have not been adequately tested and verified. This study illustrates security evaluation of a DSRC wireless application in vehicular environments (DSRC/WAVE) protocol implementation. We set up a simulation of a working road safety unit (RSU) on real DSRC devices. Our experiments work on the Cohda testbed with DSRC application wsm-channel. We extended the functionality of wsm-channel, an implementation of WAVE short message protocol (WSMP) for broadcasting GPS data in vehicular communications, to broadcast car information and RSU instructions. Next we performed Denial of Service attacks to determine how few packets need to be dropped to cause automobile crashes. Hidden Markov Models (HMM) are constructed using sniffed side channel information, since operational packets would be encrypted. The inferred HMM tracks the protocol status over time. Simulation experiments test the HMM predictions showing that we were able to drop necessary packets using side channels. The attack simulation following timing side-channel worked best to drop necessary packets with 2.5 % false positive rate (FPR) while the attack following size worked with 9.5% FPR.

ACKNOWLEDGMENTS

I would like to thank the following people, without whom I would not have been able to complete this research, and without whom I would not have made it through my master's degree.

First, I want to express my appreciation to my adviser Dr. Richard R. Brooks, whose insight and knowledge into the subject matter steered me through this research. He was patient and willing to hear my voice. And he always gave me professional suggestions or helpful experience. Working with him for more than two years has been a valuable and meaningful experience. And thanks to my committee members Dr. Lu Yu and Dr. Richard Groff. They supported me and gave advices along my thesis accomplishing.

My colleagues in Dr. Brooks research group, who have supported me with the programming skills and research progress. They are talent people and I am glad to work with them for two years.

My roommates Diejie Gao, Lu Yu and Chunpeng Shao. We spent a tough time during the Covid2019 while I was doing the thesis. They support me from both daily routine and the psychology.

Finally, thanks to my family for all the support they have shown me through these three-years of studying abroad.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

1. INTRODUCTION

Dedicated Short Range Communication (DSRC) is 802.11p based wireless communication technology. It's widely used for communication between vehicles and the surrounding infrastructure. Wireless access in vehicular environments (WAVE) is one of the communication protocols of DSRC. It provides stable, high-speed communication between connected vehicles.

Many applications based on DSRC/WAVE are being developed to improve traffic efficiency and assist driving [1]. Vehicle to vehicle (V2V) technology is in many new cars. V2V is DSRC based. Vehicles use V2V and global positioning system (GPS) to share and detect information within range. This could alert and warn drivers of emergencies which are not easy to see. For example, Left Turn Assist (LTA) systems help avoid blind spots when drivers turn left. It warns drivers if they are driving in front of another vehicle traveling in the opposite direction. It could help reduce traffic collisions.

With DSRC becoming the accepted automotive wireless mobility standard, DSRC development groups have the concern that DSRC protocols, applications, and stacks are not mature [1]. Similarly, many applications using the DSRC protocol have not been adequately tested and verified. In this thesis, we are interested in "black box" analysis of WAVE short message protocol (WSMP), the messaging protocol used by DSRC/WAVE. We assume the WSMP packets are encrypted and analysis does not depend on the

contents. We designed a traffic control system simulation of autonomous vehicles to analyze. The system can run on DSRC devices to replace traffic lights.

The application works to avoid crashes for automatic driving. We did side-channel analysis of the sniffed WSMP traffic. A Hidden Markov Model (HMM) was built using sniffed packet traces. We identified and predicted critical packets in the system using the HMM. The critical packet refers to the stop instruction packet which sent from RSU to ask a car stop. With the known weak points, we can do a targeted attack. We performed the flooding attack with HMM predictions in off-line simulation experiments. We dropped the important packets and caused car crashes.

The rest of this thesis is organized as follows: In Chapter Two, we provide background information and related work; in Chapter Three, we describe our experimental set-up and the testbed testing results and analysis; in Chapter Four, we build HMM to describe traffic in the experiment of Chapter Three and predict the key point to attack; and in Chapter Five, we provide a summary of the work and some prospects from future work.

2. BACKGROUND AND RELATED WORK

2.1. DSRC Status

At this time, DSRC is the only communication technology that could be used on connected vehicles [1][16]. As DSRC provides reliable and real-time communication between DSRC-equipped vehicles, it starts to be widely used to coordinate driving and road management. From the U.S. Department of Transportation (USDOT) report, we know that most lights and traffic signals will enable DSRC in twenty years [1]. It's reasonable because DSRC could provide real-time crash-avoiding alerts. DSRC-equipped vehicles can share critical information, so it provides the possibility of un-obstructed awareness.

Figure 2.1 shows the architecture of DSRC implement standard [2]. The physical protocol, including PHY layer and medium access control (MAC) sublayer, is defined in IEEE 802.11p WAVE [22] which enhance IEEE 802.11 (WIFI- standard) to support Intelligent Transportation System (ITS). It provides a real-time data exchanging by removing the general channel-establish in network communication. It defines the spectrum of channels for DSRC in US. Authentication and data confidentiality mechanisms provided by the IEEE 802.11 standard cannot be used. DSRC equipped vehicles in a certain sight range can receive data frames as soon as they arrive on the communication channel.

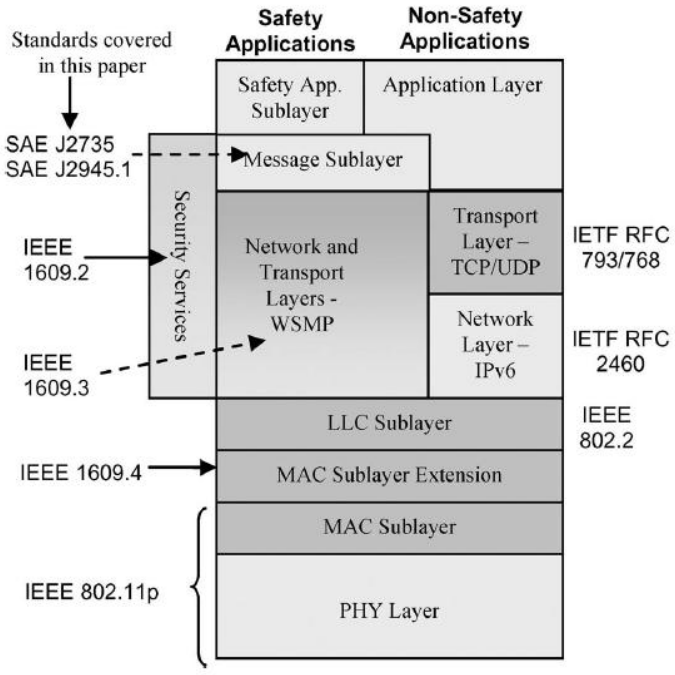


Figure 2. 1 DSRC layers and standards [2]

Although the IEEE1609.2 DSRC/WAVE stack standard defines mechanisms for authenticating and encrypting messages, WAVE implementations are still in field-test. these security mechanisms are not implemented in practice. This makes DSRC/WAVE systems particularly susceptible to wireless attacks.

Various DSRC security issues have been studied, such as message falsification, impersonation, message tampering, etc.[3][14][16] There is also research for DSRC network attack detection and prevention [4]. However, “black box” WSMP traffic analysis is lacking.

2.2. Traffic Analysis Tools

To perform the "black box" attack on WAVE short message protocol (WSMP), we use the hidden Markov model (HMM) approach. In probability theory, a Markov model is a stochastic model used to describe random process. It is assumed that future states depend only on the current state. The hidden Markov model is a Markov chain where the state is hidden or partially observable. Researchers need to determine the state from the Markov chain.

2.2.1. HMM concepts

A standard HMM has two sets of random processes, one governing state transition and the other governing symbol outputs. In this paper, we use the representation of an HMM in [17] where output symbols are associated with transitions. The two approaches are equivalent [13]. This representation uses a tuple G = <A, V, E, P>, where A is a finite alphabet of observations, V is a finite set of nodes or states, $E \subseteq V \times A \times V$ is a transition relation, and P : E→[0, 1] is a probability function such that $\sum_{a \in A, v_j \in V} p\left(v_i,\ a,\ v_j\right) = 1$.

Each element $p_{i,j} \in P$ expresses the probability the process transitions to state v_j once it is in state v_i. For each pair of (v_i, v_j), $E(v_i, v_j) = a_i$. It should also meet the requirement that if $E(v_i, v_j) = a$, then $E(v_i, v_k) \neq a$, where $v_i, v_j, v_k \in V$.

Both state transition probability matrix P and state output probability matrix O can be constructed from G. The state output probability matrix refers to the matrix described the probability distribution of next observation for each state. We use state

transition probability matrix for steady state probability calculation and figure plotting. We use state output probability matrix for generating a string from the HMM and HMM acceptance checking. Following are some important variable calculations in an HMM.

1) Conditional probability $p_{i,j} = \Pr(v_j \mid v_i)$

2) Transition count $c_{i,j} = \#transition\_from\_i\_to\_j\_happened$

   State count $c_i = \sum_j c_{i,j} = \#state\_i\_is\_entered$

3) Asymptotic probability (steady-state probability) matrix $\vec{\pi} = (\pi_1, \pi_2, ..., \pi_n)'$ can be calculated from $\begin{cases} \vec{\pi} P = \vec{\pi} \\ \sum_i \pi_i = 1 \end{cases}$

4) Confidence interval for each transition $CI = Z_{\alpha/2} \sqrt{\frac{p_{i,j}(1-p_{i,j})}{n_i}}$ [20], where $p_{i,j}$ is the conditional probability of the transition, $Z_{\alpha/2}$ is from either the normal or t-distribution, α is the significance level of confidence, n_i is the times of state v_i.

2.2.2. Model confidence test

After deriving a model from the data, we need to know whether the data is enough to derive this model. If not enough, how much more data do we need. Thus, we take the model confidence test algorithm from [17] to check the model.

With input of transition probability matrix *P*, transition count matrix *C*, and asymptotic probability matrix $\vec{\pi}$, we do the test as following:

1) Null hypothesis: data is not enough for any transitions

   Alternative hypothesis: data is enough for any transitions

2) Test statistic: $z=\min\left(\frac{p_{i,j}}{\sqrt{\frac{p_{i,j}(1-p_{i,j})}{n_i}}}\right)$, where $0 < p_{i,j} < 1$ is the conditional probability of the transition, $n_i = \sum_j c_{i,j}$ is the total counts of state *i*, $c_{i,j}$ is the element from transition count matrix *C*.

3) Rejection region: Reject *Ho* if $z > z_\alpha$ that we don't need to collect more data.

4) Otherwise we need to collect more data. Enough data $D = \max\left(\frac{z_\alpha^2(1-p_{i,j})}{p_{i,j}\pi_s}\right)$, where $0 < p_{i,j} < 1$.

2.2.3. Related work

C. R. Shalizi *et al.* [23] proposed Causal State Splitting and Reconstruction (CSSR) algorithm to generate HMMs from discrete sequences of data. The algorithm makes no prior assumptions about the model structure. The algorithm infers the model structure (the number of hidden states and their transition structure) from the sequence of observation and a maximum data window size. The derived HMMs from CSSR have predictive optimality properties.

R. R. Brooks *et al*.[7] proposed using confidence intervals (CI) with HMM to detect a behavior in a data stream. The novel approach of using CI and receiver operating

characteristic (ROC) determines whether or not a given HMM adequately matches a sequence of data.

Based on the approach of [23], J. M. Schwier *et al*. [13] presented a method for automatically inferring the maximum data window size from training data as part of the model construction process. Thus, they proposed a method inferring HMMs only from the sequence of observation.

J. M. Schwier *et al*.[20] considered detecting patterns in data streams in which two or more Markov model exists. They proposed methods of finding the proper sliding window size that can best detect changes when the behavior switches from one Markovian process to another.

L. Yu *et al*.[17] focused on the sufficiency of training data to infer an HMM. They proposed a method to determine if the observation data and constructed model fully express the underlying process with a given significant level. The method also included the calculation of an upper bound on the number of samples required to guarantee the sufficiency.

C. Lu *et al*.[18] presented a normalized statistical metric space for HMMs. With the proposed metric space, they were able to compare HMMs with a given level of statistical significance. The metric space can also provide calculation of distance between two HMMs.

Many researchers used HMM for network traffic analysis. A. Dainotti et al.[26] applied HMM to model Internet traffic of Age of Mythology, SMTP, and HTTP. The HMM built from inter-packet time and packet size side-channels of the traffic. Craven

Ryan [21] built an HMM according to the inter packet time of Tor data. Then he used the HMM for detecting the traffic through Tor. Yingbo Song et al.[24] used HMM for network behavior recognition. They modeling the dynamics of network traffic and detect the transitions between specific application layer protocols

H. Bhanu et al.[19] proposed timing side-channel analysis for detecting protocol tunneling. They used zero-knowledge approach [23] to extract HMMs for extracted keystroke dynamics of languages. They then used the HMM for language detection.

X. Zhong et al.[8][9] proposed the side-channel analysis of Phasor Measurement Unit (PMU) protocol used by the communications network of smart grid. They isolated the packets of the target PMU sent through a VPN channel shared with other PMUs, followed Denial-of-Service (DoS) attacks that selectively drops packets from the target PMU.

3. EXPERIMENT SET-UP AND RESULTS

We set up a simulation experiment of OBU and RSU working on DSRC devices. We performed penetration testing to show the communication is insecure. We consider Denial of Service (DoS) attacks. Important legitimate packets can be disrupted by flooding unexpected packets. Without instruction from RSU, we detected crashes in the intersection.

We describe our simulation road model of section 3.1 and introduce the flow of packet transmission in section 3.2. In section 3.3 we discuss how the OBU and RSU in the simulation work together to avoid crashes. In 3.4, we run simulations and perform DoS attacks.

3.1. Simulation Model

We set up a traffic light model for experiments. As shown in Figure 3.1, the traffic light is for bi-directional single lanes. Cars running on the road are following:

1) No pedestrians allowed in this crossroad.
2) Car can come from one of the four directions North (N), South (S), West (W), or East (E) and go straight, left or right. U-turn is not permitted in this crossroad.
3) All cars are autonomous vehicles and controlled by on board unit (OBU) speed control. There is a central RSU in the center of the intersection.

4) We set the center of the intersection as the origin ‘O’.

5) The roads are each 4 meters wide.

6) As shown in Figure 3.1, the exit points of each lanes are A, B, C, and D. Coordinates are A (-4, -2), B (4, 2), C (-2, 4), D (2, -4). Cars would start to report information 50 meters away from the RSU (O in Figure 3.1).

7) As shown in Figure 3.1, the entry points are A’, B’, C’, D’. Coordinates are A’ (-4, 2), B (4, -2), C (2, 4), D (-2, -4).

8) The path of the car in the intersection is calculated by linear distance from exit points to entry points. The distance a car should drive in the intersection is calculated by sum of path distance and car length. For example, if a car driving from East to North, the path in the intersection should be line BC’ (Figure 3.1).

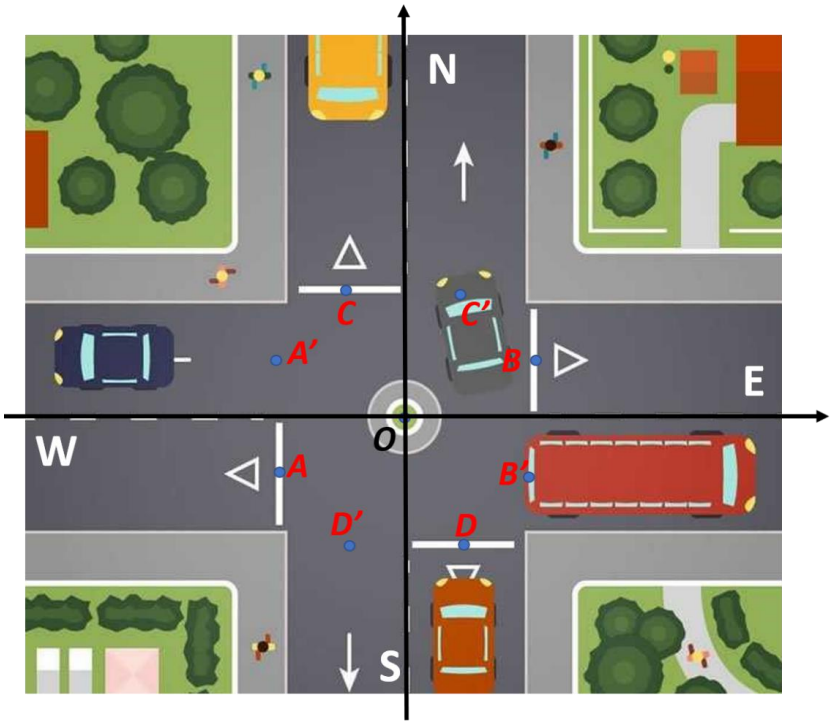


Figure 3. 1 Crossroad graph

3.2. Protocol Communication Flow

Our research group member, Jon Oakley, implemented a reliable WAVE short message protocol (WSMP) communication application wsm-channel. WSMP, IEEE 1609.3, is a DSRC based communication protocol which allows data rates parameters [2]. Wsm-channel could broadcast GSP information of host OBU on a WSMP channel. We implement the modes “FWDTX” and “FWDRX” on WSM-CHANNEL to forward packets through different protocols. FWDTX is forwarding received UDP packets to WAVE protocol. FWDRX is forwarding received WAVE packets to UDP protocol. Thus, using this extended application, processes on different OBUs can exchange data.

The flowchart of communication is in Figure 3.2. For example, if Process A on DSRC1 needs to send packet A to Process I on DSRC2; Process II receives packet A and need to send back packet B to Process I. The communication steps are as following:

1. WSM-CHANNEL FWDTX mode and FWDRX mode are running on DSRC1 and DSRC2. Process I and Process II are listening to UDP for receiving packets.

1a. DSRC1: Process I sends packet A to UDP.

1b. DSRC1: WSM-CHENNEL FWDTX thread receives packet A and send it to WSMP at interface “wave-raw”. Packet A is broadcasting at wave-raw.

2a. DSRC2: WSM-CHENNEL FWDRX thread receives packet A at wave-raw and send it to UDP.

2b. DSRC2: Process II receives packet A.

3a. DSRC2: Process II generates packet B and sends it to UDP.

3b. DSRC2: WSM-CHENNEL FWDTX thread receives packet B and send it to WSMP at interface “wave-raw”. Packet B is broadcasting at wave-raw.

4a. DSRC1: WSM-CHENNEL FWDRX thread receives packet B at wave-raw and send it to UDP.

2b. OBU1: Process I receives packet B.

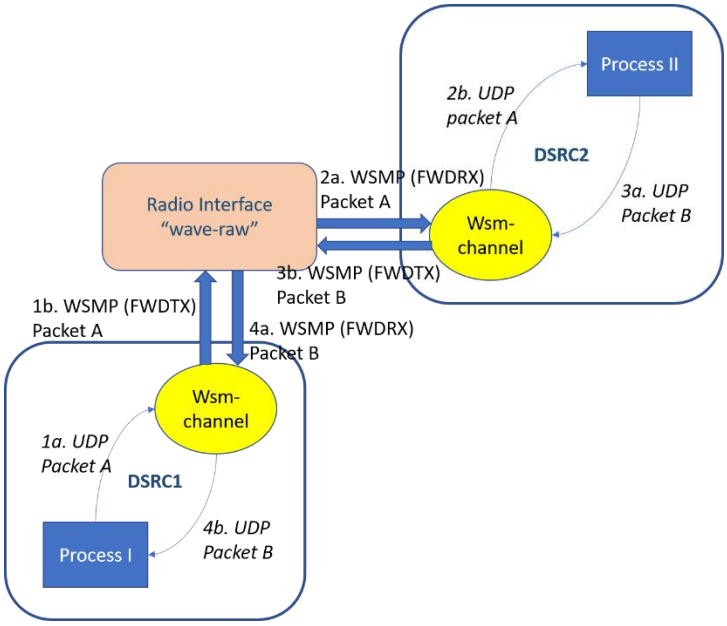


Figure 3. 2 Flowchart of packet within DSRC communication

3.3. Traffic Light Simulation Scenarios

In our simulation, we focused on the communications between an onboard unit (OBU) and a roadside unit (RSU). The OBU stores the information of all the cars

approaching the intersection; and the RSU serves as the road safety unit for the intersection. Thus, we observe communication between cars and RSU over the DSRC channel as illustrated in Figure 3.3.

The OBU has three modules: the traffic module (TM), the speed adjustment module (SAM), and the crash detection module (CDM). TM sends vehicle information to SAM, which then forwards the information about the vehicles around the intersection to the CDM. The OBU and RSU communicate via the WSMP channel interface – Wave-Raw (WR). The OBU can broadcast each vehicle's information over WR. The RSU can send stop instruction over WR. Module details are described in the following subsections.

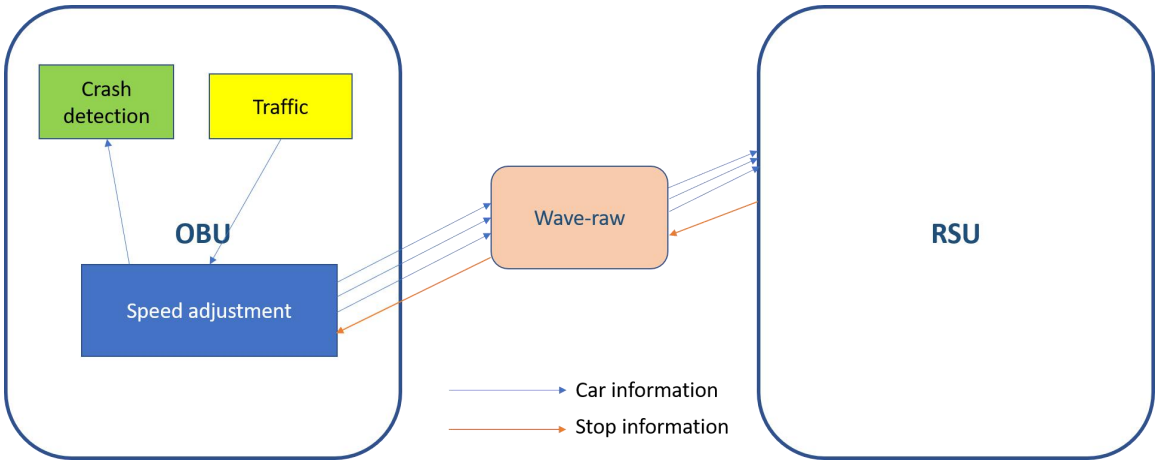


Figure 3. 3 Communication between OBU and RSU.

3.3.1. Traffic module

The traffic module (TM) generates car information in a predefined rate, which includes vehicle ID, timestamp, lane, moving direction, location, vehicle length, speed, and acceleration. Lane is chosen from West (W), East (E), North (N), South (S). Cars

coming from each direction may go straight, turn left or turn right. Note that U-turn is forbidden at an intersection.

Each car reports sends its information to when they are 50 meters away from the intersection. The length of the car is a random number between 3 m to 5 m. The initial speed is randomly chosen between 11 m/s to 14 m/s. The initial acceleration is set to 0 m/s^2. In each scenario, we assume there are at most three cars approaching the intersection. The goal is to avoid car crashing. No more than one car should be moving in a direction at one time. The interval between cars in one scenario is 0.1 to 0.3 seconds. The interval between two scenarios is a random choice of from 5.0 seconds to 8.0 seconds.

3.3.2. Speed adjustment module

The speed adjustment module (SAM) receives the vehicles data from the traffic module and then forwards it to the RSU. A car may receive a “stop” instruction while it is approaching the intersection. After the car coming into the intersection, the SAM sends the car information to the crash detection module. The communication is shown in Figure 3.4.

```
send contoller: 0
send contoller: 1
stop: ['stop', 1, 1593384540.2245257, 1, 2, 49.97379891726753, 3.469499209584144
6, 11.08821573091079, 0, 1593384545.2267432, 1593384546.272803, [1, 2, 3, 4, 5,
7, 8, 11, 12]]
cross:  0
cross:  1
send contoller: 2
cross:  2
```

Figure 3. 4 Speed adjustment module

The speed adjustment module traces car information updates per unit time (0.1s), including the vehicle's location, speed and acceleration. Vehicles adjust their speed and acceleration following the received instructions.

1) When the speed adjustment module receives a car's information from the traffic module, it starts tracing the car and sends the collected data to RSU.
2) The speed and acceleration of a vehicle changes according to the received stop instructions.
3) If a car stops before the intersection as instructed, it later starts with initial speed of 8 m/s at the instructed time.

A car will run as instruction from RSU, or with the initial parameters until it receives an instruction.

3.3.3. Road Safety Unit (RSU)

The RSU stops vehicles from crashing at the intersection. It receives the vehicles' information from the OBU and sends "stop" instructions to the cars as needed. Refer to Figure 3.5 for the traffic data received and sent on this module.

Vehicles come from any direction and go into one of the other three directions. This results in 12 types of car transitions in total. details of transitions are in Table 3.1. We list the forbidden behaviors for vehicles coming from different directions in Table 3.1. For example, for a car coming from west street and going through the intersection, behaviors with ID numbers {1, 2, 3, 6, 7, 8, 10, 11, 12} (Table 3.1) are forbidden to avoiding accidents.

Lane	ID Number	Driving Direction	Yield To
West (W)	1	Forward (from west to east)	1, 2, 3, 6, 7, 8, 10, 11, 12
	2	Left Turn (from west to north)	1, 2, 3, 4, 5, 7, 8, 11, 12
	3	Right Turn (from west to south)	1, 2, 3, 6, 8
East (E)	4	Forward (from east to west)	2, 4, 5, 6, 7, 8, 9, 11, 12
	5	Right Turn (from east to north)	2, 4, 5, 6, 11
	6	Left Turn (from east to south)	1, 3, 4, 5, 6, 7, 8, 11, 12
North (N)	7	Left Turn (from north to east)	1, 2, 4, 6, 7, 8, ,9, 10, 11
	8	Forward (from north to south)	1, 2, 3, 4, 6, 7, 8, ,9, 12
	9	Right Turn (from north to west)	4, 7, 8, ,9, 12
South (S)	10	Right Turn (from south to east)	1, 7, 10, 11, 12
	11	Forward (from south to north)	1, 2, 4, 5, 6, 7, 10, 11, 12
	12	Left Turn (from south to west)	1, 2, 4, 6, 8, 9, 10, 11, 12

Table 3. 1 Driving rules for vehicles approaching the intersection.

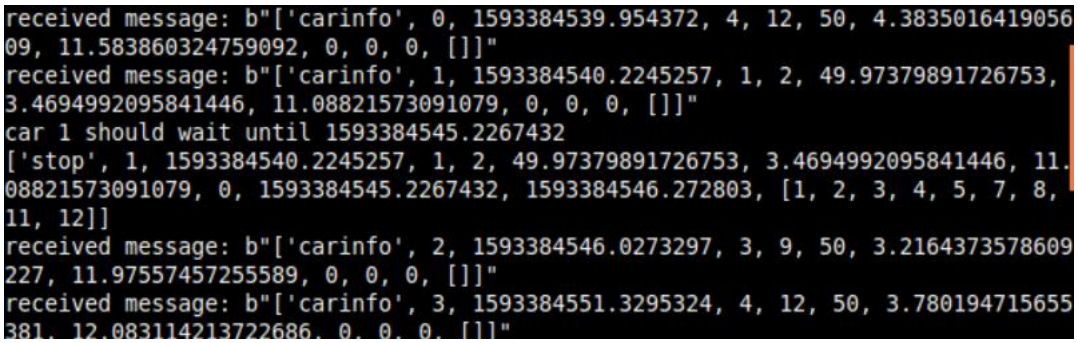


Figure 3. 5 Traffic Controller Module

The RSU keeps a list of reservation time for each driving direction. When the first packet of a car arrives, the traffic module estimates the arrival time of the car at the intersection. The calculation is based on the distance of the car from intersection (Figure 3.1), its speed, acceleration, and the length of the car. The RSU compares the car's arrival time with the reservation time of its driving direction in the list. If the arrival time is later than the reservation time, RSU will estimate the car's exiting time at the intersection and update the reservation time for its moving direction and associate forbidding moving directions. If the arrival time is earlier than the reservation time, RSU will send the stop instruction which includes the available time into the intersection. Then it will allocate the reservation time for this transition. The car that received a stop instruction should not enter the intersection until the available time.

3.3.4. Crash detection module (CDM)

When a car enters the intersection, crash detection module reports the simple road condition and keeps tracking the new arrived car. As shown in figure 3.4(a), the road condition reports the driving direction ID numbers of cars in the intersection.

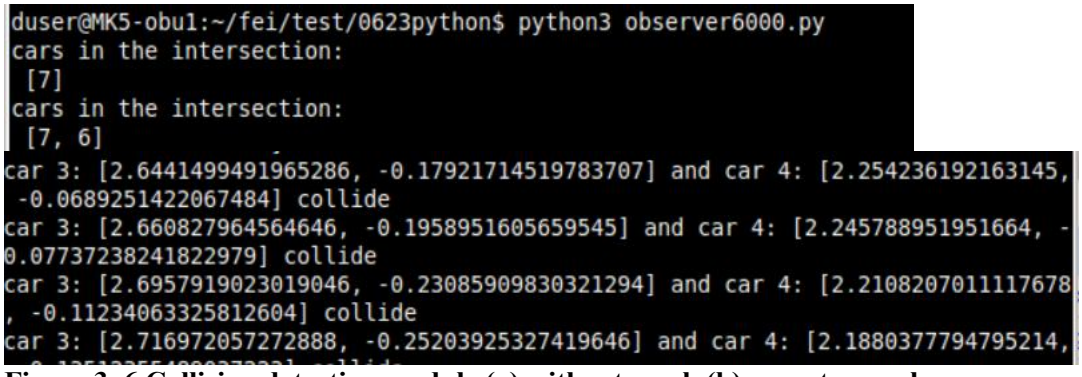


Figure 3. 6 Collision detection module (a) without crash (b) report a crash

The CDM updates car position every 0.1s and calculates the distance between any two cars. If the distance is smaller than 2 meters, CDM reports a crash (Figure 3.4 (b)). As a car drives away from the intersection, the CDM stops tracing it.

3.4. Simulation Experiments and Results

We ran the simulation on DSRC devices as Figure 3.3: on the first DSRC device, we ran car processes; on the second DSRC device, we ran the traffic controller.

First, we did the test for the simulation function. In continuous 2000 scenarios of cars arrival, the RSU works well to avoid crashes. We captured the traffic using *tcpdump*. From figure 3.7 of stop instruction packet details, we can see the time shift for this packet is approaching 0 seconds which shows the real-time data exchange.

```
Frame 3224: 253 bytes on wire (2024 bits), 253 bytes captured (2024 bits)
    Encapsulation type: Linux cooked-mode capture (25)
    Arrival Time: Jun 24, 2020 13:12:19.361308000 Eastern Daylight Time
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1593018739.361308000 seconds
    [Time delta from previous captured frame: 0.030123000 seconds]
    [Time delta from previous displayed frame: 0.030123000 seconds]
    [Time since reference or first frame: 7182.082375000 seconds]
    Frame Number: 3224
    Frame Length: 253 bytes (2024 bits)
    Capture Length: 253 bytes (2024 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: sll:ethertype:wsmp]
```

Figure 3. 7 Stop instruction packet

Then we performed Denial of Service attack by packet flooding. We kept the simulation running with GPS information broadcasting on OBU at a very high speed rate to cause packet dropping on the RSU (figure 3.8). While the flooding attack is ongoing, the Collision Detection Module detected several crashes immediately (figure 3.9). With 110 packets from RSU dropped, 27 crashes are detected by Collision Detection module.

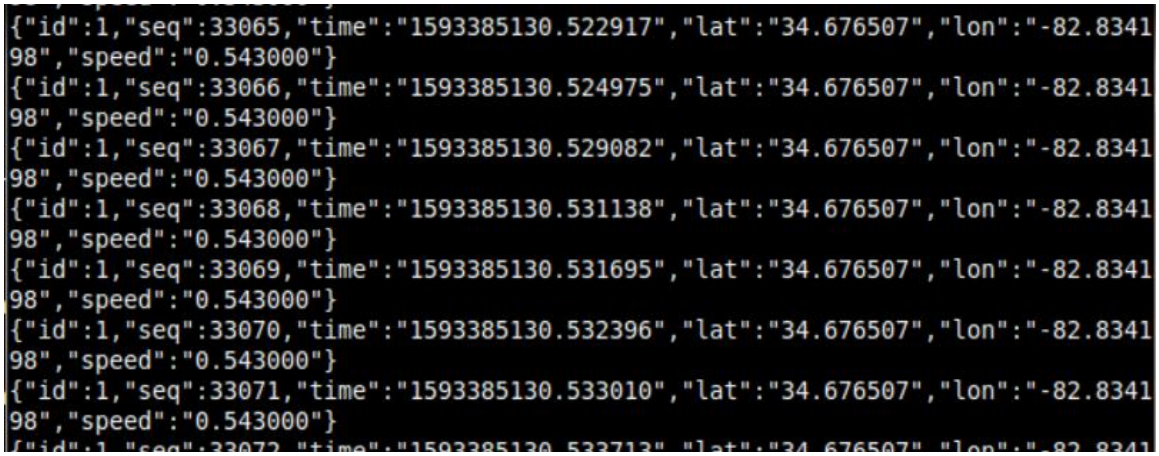


Figure 3. 8 Flooding GSP information

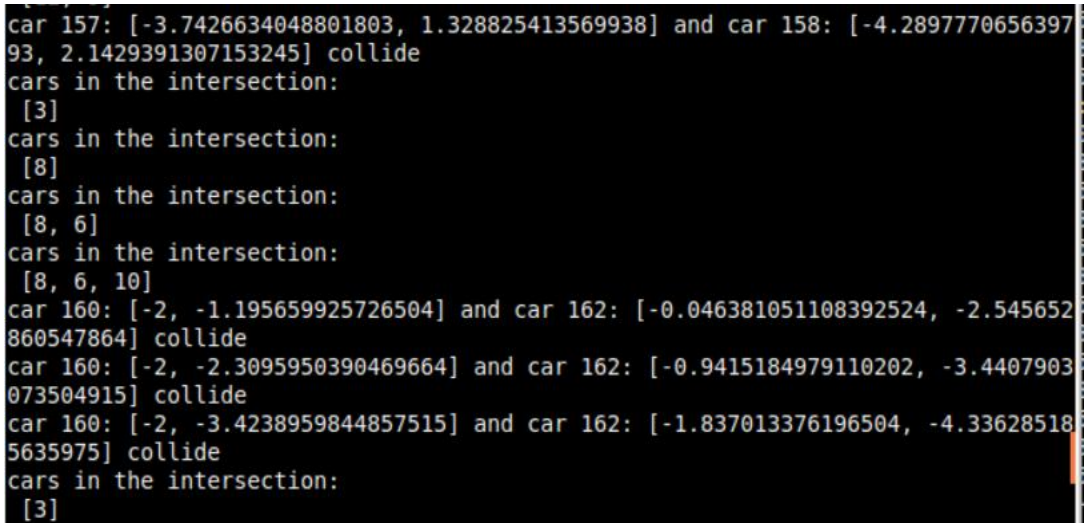


Figure 3. 9 Collision detected while drop stop instructions

4. SIDE-CHANNEL ANALYSIS AND RESULTS

In addition to the "white-box" testing in Section 3.4, we also look at the side-channel characteristics (packet size, packet inter-delay) of WAVE short message protocol (WSMP). Even if encryption and authentication are implemented as specified in the IEEE 1609.2 standard, DSRC/WAVE may still be susceptible to such "black box" analysis that does not depend on in the contents.

From the sniffed traffic (Figure 4.1), the packets are not arriving at the same rate all the time, which means the protocol is not active all the time. If we perform the attack at an inactive time, we cannot cause any trouble. Moreover, since flooding traffic is easy to recognize, the devices may lose access to the channel. Therefore, we build Hidden Markov Model (HMM) for the system protocol to understand the protocol regulations. As we are assuming that the WAVE packets will be encrypted, we apply size and timing side channels. We sniff traces of DSRC network protocols. We can identify network protocol states by using observed packet characteristics to associate each sniffed packet with a class. Protocol participants are known. Transitions between protocol states are given by their positions in the sequence. With the HMM, we successfully isolate the target packets of stop information sent by RSU, followed Denial-of-Service (DoS) attacks that selectively drops packets from RSU. The goal of is to side-channel vulnerabilities of WAVE protocol assuming all the security services are implemented.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	b8:ff:36:ff:36…		WSMP	176	WAVE Short Message Protocol IEEE P1609.3
2	0.004006	b8:ff:36:ff:36…		WSMP	178	WAVE Short Message Protocol IEEE P1609.3
3	0.270763	b8:ff:36:ff:36…		WSMP	177	WAVE Short Message Protocol IEEE P1609.3
4	0.025839	a8:ff:36:ff:36…		WSMP	234	WAVE Short Message Protocol IEEE P1609.3
5	6.853376	b8:ff:36:ff:36…		WSMP	177	WAVE Short Message Protocol IEEE P1609.3
6	0.027991	a6:ff:36:ff:36…		WSMP	233	WAVE Short Message Protocol IEEE P1609.3
7	0.212516	b8:ff:36:ff:36…		WSMP	177	WAVE Short Message Protocol IEEE P1609.3
8	0.230590	ba:ff:36:ff:3c…		WSMP	177	WAVE Short Message Protocol IEEE P1609.3
9	0.030930	a8:ff:30:ff:36…		WSMP	234	WAVE Short Message Protocol IEEE P1609.3
10	6.368706	b8:ff:36:ff:36…		WSMP	193	WAVE Short Message Protocol IEEE P1609.3
11	0.029879	a6:ff:36:ff:36…		WSMP	252	WAVE Short Message Protocol IEEE P1609.3
12	6.531107	b8:ff:36:ff:36…		WSMP	178	WAVE Short Message Protocol IEEE P1609.3

Figure 4. 1 Sniffed DSRC traffic

In this chapter, we develop a network protocol analysis method based on side channel and HMM. The overall process flow is shown in figure 4.2. In section 4.1, we discuss the method to symbolize side channel information. In section 4.2, we propose our HMM inference method. In section 4.3, we sniff DSRC traffic of training data and apply the analysis method. In section 4.4, we test the HMM on the training data and new collected data.

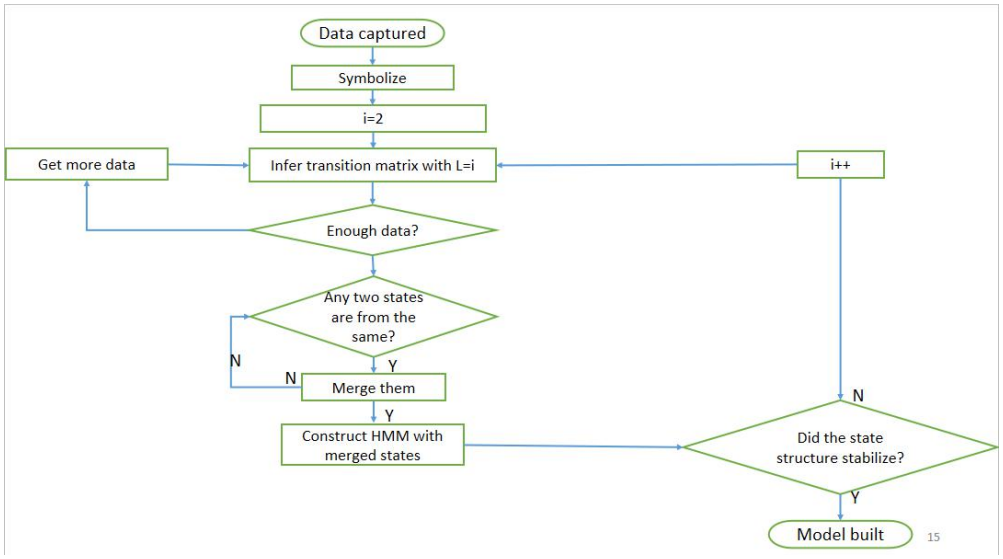


Figure 4. 2 Flowchart of inferring HMM [21]

4.1. Side Channel Symbolization

According to the sniffed traffic (figure 4.1), we can get two important side channels: timing and size. We use inter-packet time instead of receive time for analysis. The inter-packet time, also known as delta time Δt , is calculated by subtracting the receive time of the previous packet from the time of the current packet. In other words, $\Delta t = t_i - t_{i-1}$, where it is the receive time of packet i. To exclude $\Delta t = 0$ of first packet, we start with i=2.

We have two side channels, so we can build an HMM for each side channel. First, we build the timing HMM. We group the data by plotting histogram of timing and finding different ranges. We assign anything in a timing range a unique symbol. Finally, we can get a long sequence string from the data. Later we will do the same for size when build the size HMM.

4.2. HMM Inference

We use HMM to analyze side channel information. We extend previous approaches [21][13], adding hypothesis tests when determine the HMM. We apply z-test to HMMs to determine the statistical significance of the inferred model, which indicates data sufficiency [17]. Pearson chi-square test proves the significance of evidence to merge two similar states [6]. Confidence interval approach provides level of acceptance for putting a string into an HMM [20].

4.2.1. Inferring an HMM from sequence $\mathcal{A}$ with significance level α

1) i=2.
2) State space parameter L=i.
3) a) Infer G_n = <A, V, E, P> from the sequence $\mathcal{A}$.

   b) Merge states in V using Algorithm 1 Pearson chi-square test.

   c) Do model confidence test for G_n. If doesn't enough, get more data and start over. Details of Model confidence test are described in section 4.2.3.
4) Get output confidence interval matrix *CI* from G_n
5) a) Infer G_{n+1} = <A', V', E', P'> from the sequence $\mathcal{A}$.

   b) Merge states in V' using Pearson chi-square test in section 4.2.2.

   c) Do model confidence test for G_{n+1}. If the training data doesn't enough, get more data and start over.
6) Generate a long sequence $\mathcal{B}$ from G_{n+1} whose length longer than the result from model confidence test. The generation method see section 4.2.3.
7) Put the sequence $\mathcal{B}$ into G_n. Get match probability matrix F.
8) Calculate $|P-F|-CI$, the elements less than zero in the result matrix donates the rejection proportion. Determine the rejection proportion by $P_{rj} = \sum_i d_{i,j} * p_i$, where $d_{i,j} = D_{i,j} - CI, D = \{D \in (P-F) \mid D > 0\}$, p_i is the probability of state i is entered.
9) If P_{rj} greater than α, i++. Repeat steps from 2).

10) Otherwise, quit with G_n as the correct HMM for sequence $\mathcal{A}$.

4.2.2. State merging algorithm

We use pairwise Pair wise Pearson chi-square test for state merging. The test result shows whether two states are coming from the same state. We merge the pair of most likelihood at one time and update the merging in output count matrix. We keep doing the pairwise test until all pairs reject the null hypothesis of two states from the same state. With input of state transition count matrix *M,* state output matrix *O*, and significant confidence level α, we do the state merging as:

1) Do Pearson pairwise chi-square test of independence [6] of rows in transition count matrix M as following:
   a. Denote the population proportion (or probability) falling in row i, column j as π_{ij} . The total proportion for row i is $\pi_{i.}$ and the total proportion for column j is $\pi_{.j}$. If the row and column proportions are independent, then $\pi_{ij}=\pi_{i.}\pi_{.j}$.
   b. The estimated expected value in row i, column j is
   $$E_{ij} = n\pi_{ij} = n\frac{n_{i.}}{n}\frac{n_{.j}}{n} = \frac{(n_{i.})(n_{.j})}{n}$$
   c. Test statistic: $\chi^2 = \sum_{i,j}[\frac{(n_{ij} - E_{ij})^2}{E_{ij}}]$
2) Determine the $\chi^2_{\alpha,df}$ statistic for the χ^2 test with significant level α and df = *(r - 1)(c - 1)* where *r* = number of rows, *c* = number of columns.

3) If $\chi^2 <= \chi^2_{\alpha,df}$ for any pairwise tests, the test accepts with significant level α the hypothesis that the two rows are from same state. Find the minimum χ^2 value $\chi^2_{\min}$, and index *i, j* (*i*< *j*) of the pair of states it comes from.
   a. In the state transition count matrix M, add column j to column i, add row j to row i. Set zero of column j and row j.
   b. In the state output count matrix O, add row j to row i. Set zero of column j.
4) Repeat steps 1), 2), 3) until $\chi^2 > \chi^2_{\alpha,df}$ for all pairwise tests.
5) Remove zero columns and zero rows in M and O. Then quit with merged states transition count matrix and output count matrix.

4.2.3. Generate a sequence of length *l* from an HMM G

We restrict our discussion to ergodic Markov processes, which for all states possibly going to any states.

1) Randomly choose an initial state $v_i = v_0$ from state set V
2) Using the probabilities of the outgoing transitions, select a transition $p_{i,j}$ to move to state v_j from state v_i.
3) Record the label $a_i = E(v_i, v_j)$, where a_i is associated with the chosen transition $p_{i,j}$.
4) Repeat steps 2) and 3) until *l* labels have been recorded.

4.2.5. Put sequence $\mathcal{A}$ into an HMM G [13]

For every state v_i in V of G, we calculate the state transition probability F in sequence $\mathcal{A}$. If there's no transition in G for a window in sequence $\mathcal{A}$ going to the next window, record it as a rejection and turn to next window.

4.3. Experiment Data Analysis

We ran the simulation as described in chapter 3. In addition, we used *tcpdump* on the third DSRC device to sniff the OBU/RSU traffic. Since *tcpdump* is less stable than DSRC based communication software, the pipe on the third DSRC was broken after captured about 3,000 packets. In order to get enough data, we did the experiment twice. In total, we got 3197+3236 = 6433 packets.

4.3.1. Timing side channel analysis

We followed the steps in section 4.1 and 4.2 to process the data. First, we did the symbolization of timing side channel information. We plotted histograms of inter-packet timing (figure 4.3) and got three normal distribution curves in range (0, 0.06), (0.06, 1), (1, 9). So, we conclude there are three types of packets according to inter-packet time features.

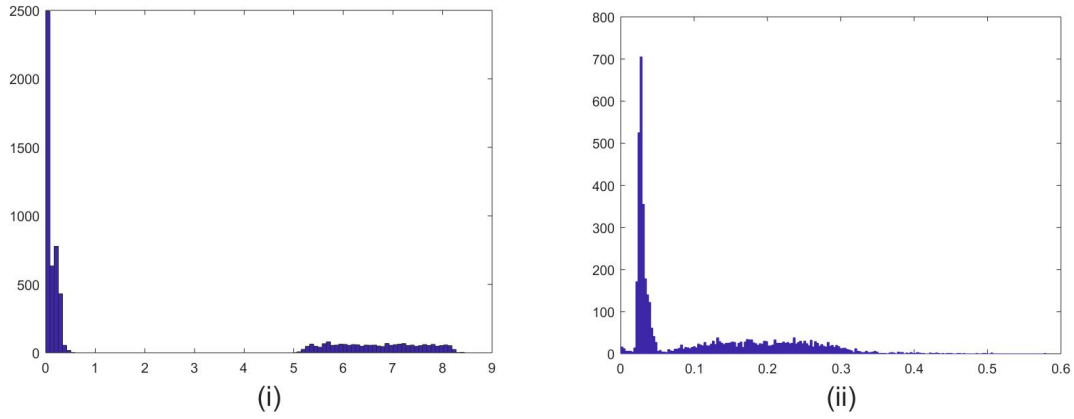


Figure 4. 3 Histogram figure of timing (i) range (0,9) (ii) range (0,1)

We labeled the traffic flow with different letters and get a long string of observations. Each letter in the string represents a packet and we want to build a model to find states (hidden) and transition between states. Symbolization range details as shown in table 4.5.

Observation type	timing range
a	$\Delta t \leq 0.06$
b	$0.06 < \Delta t \leq 1$
c	$\Delta t > 1$

Table 4. 1 Timing observation ranges

Then we applied our HMM inference approach (Algorithm 3). We got correct HMM with state space parameter L=2 since it accepted the sequence $\mathcal{A}$ generated by HMM with L=3. The plotted figure is shown in Figure 4.4. The HMM with L=2 states detail is shown in Table 4.2. The transition probabilities matrix $P_{i=2}$ and its confidence interval size $CI_{L=2}$ is shown in Table 4.3. The inferred transition probability matrix F of $\mathcal{A}$ is shown in Table 4.4. We calculated $|F - P_{L=2}|$ and compared it with $CI_{L=2}$. Each element of $|F - P_{L=2}|$ is less than associated element of $CI_{L=2}$, in turn indicates the HMM of i=2 accept the sequence generated from HMM of i=3.

State	Asymptotic probability	Strings
1	0.1528	ab
2	0.2289	ba
3	0.2061	ac
4	0.1301	ca
5	0.0342	bb
6	0.1237	cb
7	0.0159	aa
8	0.0608	cc
9	0.0476	bc

Table 4. 2 **Timing HMM of i=2 states**

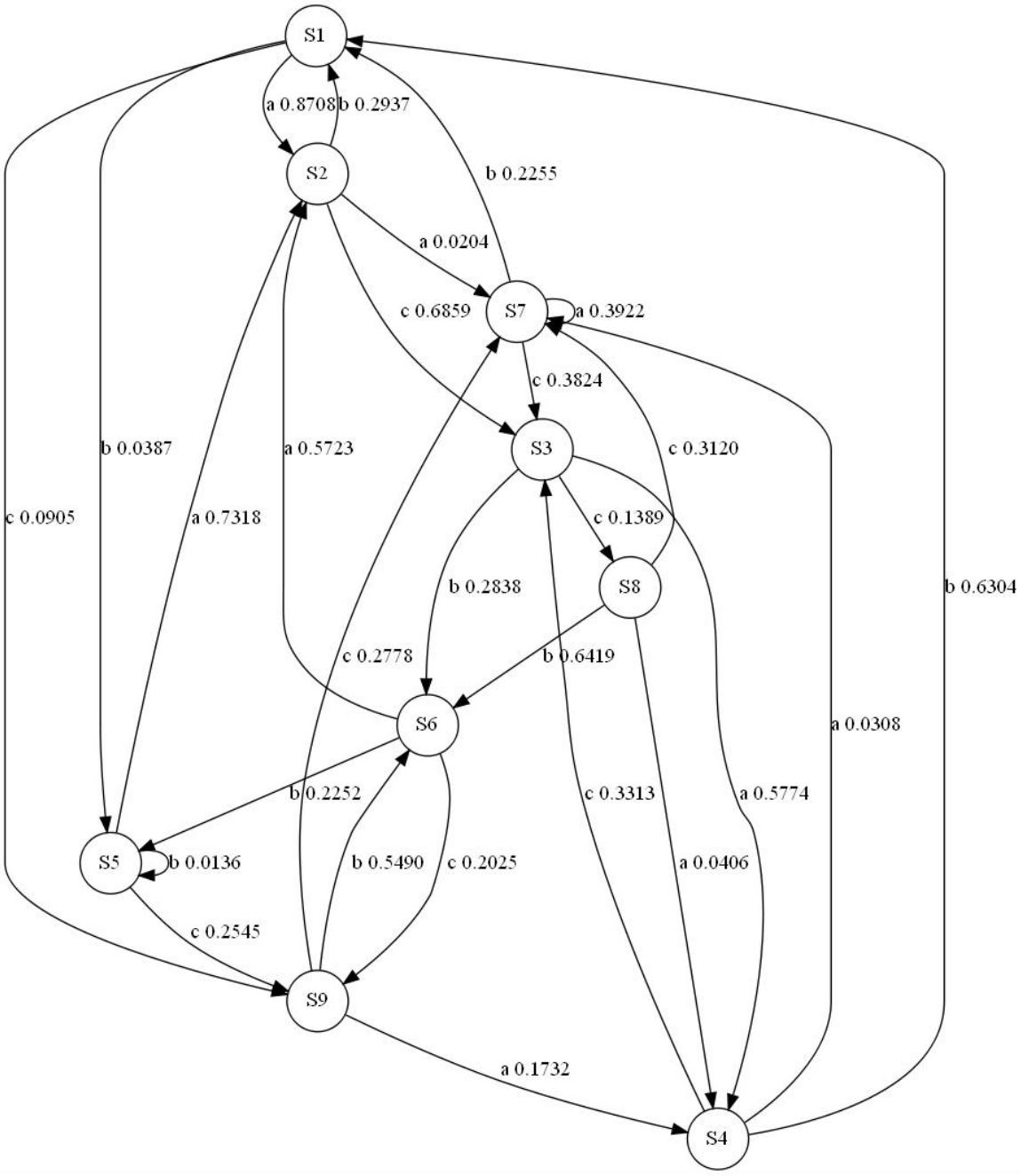


Figure 4. 4 Timing HMM of i=2

Output	a		b		c	
States	$P_{L=2}$	$CI_{L=2}$	$P_{L=2}$	$CI_{L=2}$	$P_{L=2}$	$CI_{L=2}$
1	0.8708	0.0210	0.0387	0.0210	0.0905	0.0179
2	0.0204	0.0072	0.2937	0.0072	0.6859	0.0237
3	0.5774	0.0266	0.2838	0.0266	0.1389	0.0186
4	0.0383	0.0130	0.6304	0.0130	0.3313	0.0319
5	0.7318	0.0585	0.0136	0.0585	0.2545	0.0576
6	0.5723	0.0344	0.2252	0.0344	0.2025	0.0279
7	0.3922	0.0948	0.2255	0.0948	0.3824	0.0943
8	0.0460	0.0208	0.6419	0.0208	0.3120	0.0459
9	0.1732	0.0424	0.5490	0.0424	0.2778	0.0502

Table 4. 3 Timing transition probability matrix of HMM with L=2

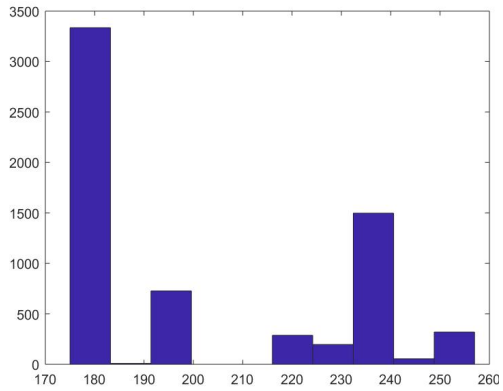
States	a	b	c
1	0.8800	0.0364	0.0836
2	0.0194	0.2972	0.6835
3	0.5873	0.2723	0.1404
4	0.0438	0.6321	0.3241
5	0.7300	0.0114	0.2586
6	0.5957	0.2066	0.1977
7	0.4000	0.2339	0.3661
8	0.0645	0.6150	0.3205
9	0.1634	0.5673	0.2693

Table 4. 4 Inferred transition probability matrix F

4.3.2. Size side channel analysis

We plotted histogram figure of inter-packet time (figure 4.5) and got three normal distribution curves in range (175, 190), (190, 200), (200,260). So, we conclude there are three types of packets (table 4.5) according to inter-packet time features.

We got sequence generated by i=3 HMM accepted by i=2 HMM with significant level 0.05. The plotted figure (figure 4.6), state detail (table 4.6), and output probability matrix are as followed.



Observation type	size range
a	$s \leq 190$
b	$190 < s \leq 210$
c	$s > 210$

Table 4. 5 Observation ranges

Figure 4. 5 histogram figure of packet size

State	Asymptotic probability	Strings
1	0.2688	aa(0.6665), bb(0.0312), ab(0.1462), ba(0.1561)
2	0.3644	ac(0.8241), bc(0.1759)
3	0.3644	ca(0.8168), cb(0.1832)
4	0.0023	cc(1)

Table 4. 6 Size HMM state

	a	b	c
S1	0.4960	0.1110	0.3931
S2	0.8117	0.1819	0.0064
S3	0.2408	0.0491	0.7101
S4	0.8000	0.2000	0

Table 4. 7 Size HMM output probability matrix

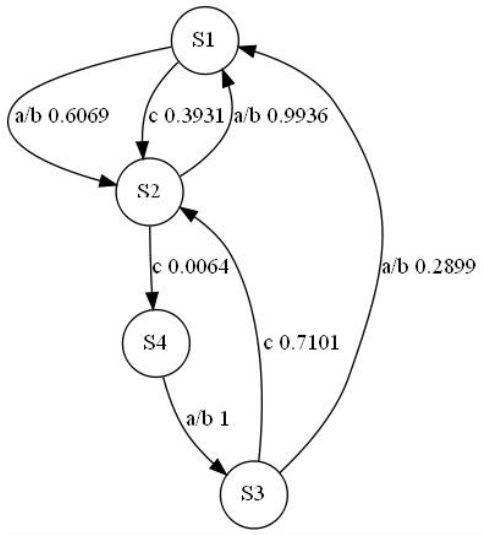


Figure 4. 6 size HMM

4.3.2.1. Discussion of size HMM

With the state detail table (table4.6) and the HMM (figure 4.6), it's easy to find packet a and b playing the same role. In the state table, if a string X containing the symbol 'a' is in the state S, the string Y which replacing any 'a' in X with 'b' is also in the state S. In the HMM figure, the transition between states are output 'c' or 'a/b' which means symbol 'a' and 'b' are leading same transitions. So, we merge symbol 'a' and symbol 'b' into one symbol 'x'. We mark symbol 'c' as symbol 'y'. See Table 4.8 the merged packet size classification and Figure 4.7 the merged packet size HMM.

Observation type	size range
x	$s \leq 210$
y	$s > 210$

Table 4. 8 Merged packet size classification

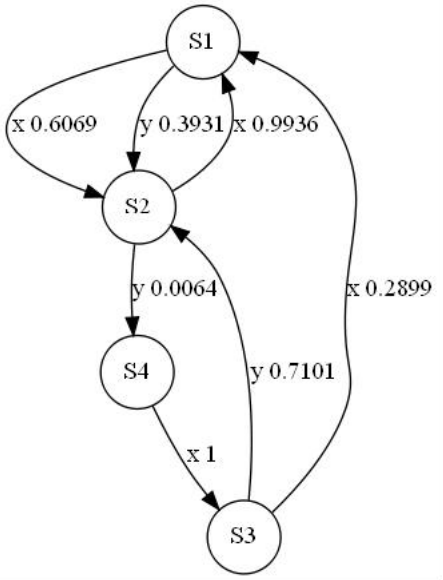


Figure 4. 7 merged packet size HMM

State	Asymptotic probability	Strings
1	0.2688	xx
2	0.3644	xy
3	0.3644	yx
4	0.0023	yy

Table 4. 9 Merged packet size HMM state detail

4.4. Targeted Attack Simulation with HMMs Prediction

In our simulation, four attack scenarios are defined, including the control group, timing side-channel analysis, size side-channel analysis, and attacks combining both timing side-channel and size side-channel. We describe the different states of the target in section 4.4.1. In section 4.4.2, we introduce the experiment set-up. In, The simulation results and the analysis of the results are presented in section 4.4.3 and section 4.4.4, respectively.

4.4.1. Target states

We first constructed the transition probability matrix (shown in Table 4.10) from the timing HMM in section 4.3.1. Type ‘a’ packet is sending by the RSU, which includes the “stop” instruction. According to Table 4.9, the packet leaving state S1 has the highest likelihood (0.8708) to be an ‘a’ packet. And the packet leaving state S5 has the second highest likelihood (0.7718) to be an ‘a’ packet. According to Table 4.6, state S1 refers to string ‘ab’ and state S5 refers to string ‘bb’.

Timing Packet	a	b	c
S1	0.8708	0.0387	0.0905
S2	0.0204	0.2937	0.6859
S3	0.5774	0.2838	0.1389
S4	0.0383	0.6304	0.3313
S5	0.7318	0.0136	0.2545
S6	0.5723	0.2252	0.2025
S7	0.3922	0.2255	0.3824
S8	0.0460	0.6419	0.3120
S9	0.1732	0.5490	0.2778

Table 4. 10 Timing HMM transition probability matrix

Size packet	x	y
S1	0.6069	0.3931
S2	0.9936	0.0064
S3	0.2899	0.7101
S4	1	0

Table 4. 11 Size HMM transition probability matrix

We then constructed the transition probability matrix (Table 4.11) from the size HMM in section 4.3.2. Type 'y' packet sent by the RSU includes the "stop" instruction. According to Table 4.11, we found the packet leaving state S3 has the highest likelihood (0.7101) to be a 'y' packet. According to Table 4.9, state S3 refers to string 'yx'.

As shown in Table 4.12, we have three target states: timing state 'ab', timing state 'bb', and size state 'yx'. We set up attack simulation to test the HMMs prediction by dropping the packets leaving the target states.

Target State	Category	Strings
1	**Timing**	ab
2	**Timing**	bb
3	**Size**	yx

Table 4. 12 Target states

4.4.2. Attack simulation set up

In the simulation, we set five processes: traffic, speed adjustment, crash detection, wave-raw, and RSU. The network topology is shown in Figure 3.3. We used the wave-

raw process to simulate the DSRC/WAVE communication channel. The wave-raw process sends packets from the speed adjustment module and RSU to each other.

To perform the attack, we need to recognize the timing type and size type for each packet. For every received packet *i*, the wave-raw process calculates the delta time for it by $\Delta t = t_i - t_{i-1}$. The corresponding timing symbol ('a', 'b' or 'c') was assigned to each delta time according to Table 4.1. The size of each packet is symbolized in the same way using data size range of Table 4.13. The real WSMP packet consists of the header and the payload. Take the packet shown in Figure 4.9 as an example, the data length for the packet is 97 bytes while the packet size is 177 bytes . So, we fix this difference when doing the off-DSRC simulation experiment (as shown in Table 4.13).

Observation type	Packet size range	Data size range
x	$s \le 190$	$s' \le 110$
y	$190 < s \le 210$	$110 < s' \le 130$

Table 4. 13 Simulation data size range

```
> Frame 5: 177 bytes on wire (1416 bits), 177 bytes captured (1416 bits)
> Linux cooked capture
v Wave Short Message Protocol(IEEE P1609.3)
    Version: 178
    0... .... = Length: 0x00
    .000 0010 = PSID: psid-freight-fleet-management(2)
    Transmit Power: 64
    Data Rate: 12
    Channel: 178
    WAVE element id: WSMP (128)
    WSM Length: 97
    Wave Short Message
```

Figure 4. 8 The WSMP packet

Given the string of timing symbols and the string of size symbols. The wave-raw process detects the states by looking at the two symbols in the end of each . If the target state (Table 4.12) is recognized, the next received packet is dropped, and is not forwarded to its destination protocol. After a packet is ignored, the wave-raw process starts over to detect the next defined state.

The attack was simulated under six different scenarios. The first scenario is a control group to see the crash rate if all packets from RSU dropped. In this scenario, the wave-raw process didn't forward any packets from RSU to OBU. In the second scenario, we dropped the packet after the first timing state 'ab' (see Table 4.12) was observed. In the third scenario, we dropped the packet after either timing state 'ab' or 'bb' (see Table 4.12) was seen. In the fourth scenario, we dropped the packet after size state 'yx' (Table 4.12) occurred. In the fifth scenario, we dropped the packet after any defined target state. In the sixth scenario, we dropped the packet after the state is recognized as a combination of a target timing state and a size state.

4.4.3. Simulation results

We targeted the packets containing the "stop" instruction in this simulation. We mark packets from RSU as positive packet, packets from OBU as negative packets. The true positive (TP), true negative (TN), false positive (FP) and false negative (FN) of our attack as defined as:

- TP is the attack drops a packet sent from RSU.
- TN is the attack doesn't drop a packet sent from OBU.

- FP is the attack drops a packet sent from OBU.
- FN is the attack doesn't drop a packet sent from RSU.

We record the packets numbers of these four types. We also record the number of car crashes in each experiment scenario.

For each scenario, there are 2000 cars approaching the intersection in total. The simulation results are shown in Table 4.14. The second column shows the target state for each scenario. The third column shows the portion of crashes in dropped packets $p_c = \frac{\#crash}{\#dropped}$. The fourth column shows the false positive rate (FPR) calculated by $FPR = \frac{FP}{FP + TN}$. The fifth column shows the true positive rate (TPR) calculated by $TPR = \frac{TP}{TP + FN}$.

Scenario	Attack target state	Crash proportion(%)	FPR (%)	TPR (%)
1	Control group	22.74	0	100
2	Most likelihood timing state	11.46	2.50	28.48
3	Any target timing states	16.14	3.05	35.81
4	Target size state	13.74	9.45	39.53
5	Any defined target state	16.69	9.29	45.09
6	Combination of a target timing state and a size state.	9.70	1.9	28.29

Table 4. 14 Timing side channel attack simulation results

4.4.4. Analysis of the simulation results

To evaluate different attack scenarios, we plotted column chart of true positive rate (TPR), false positive rate (FPR) and crash proportion with the confidence interval (CI) for each scenario. The confidence interval is calculated by $p \pm Z_{1-\alpha}\sqrt{\frac{p(1-p)}{N}}$, where $Z_{.95} = 1.96$. The charts are shown in Figure 4.9.

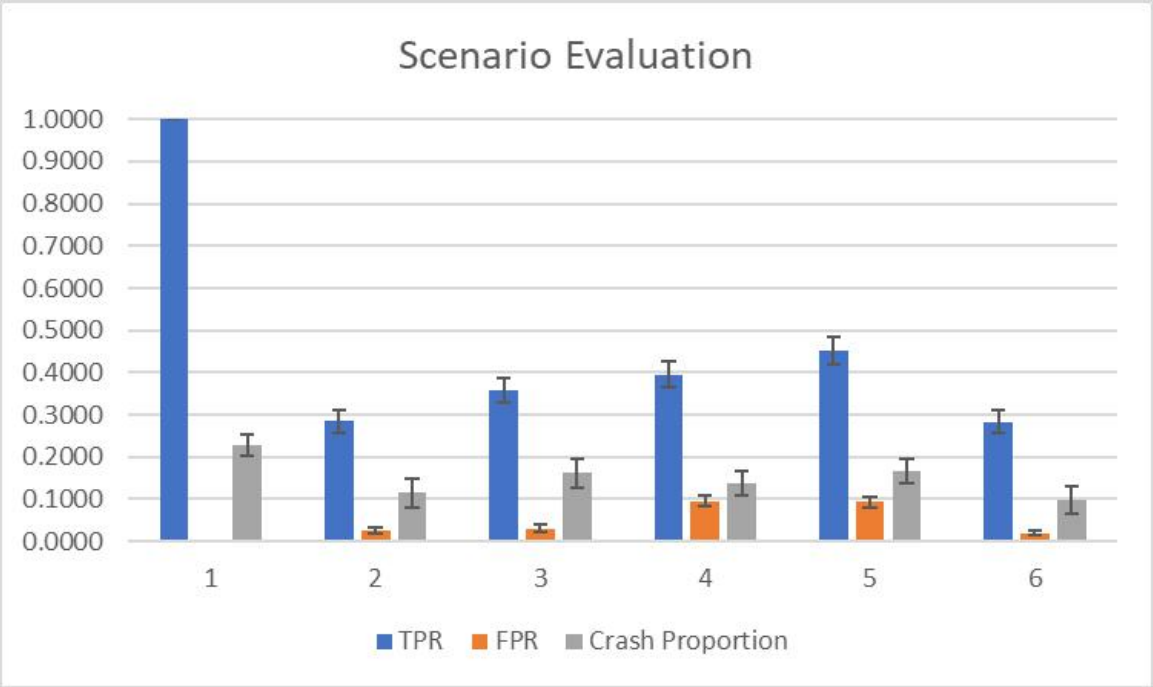


Figure 4. 9 Scenario Evaluation

As shown in the Figure 4.9, the control group is the first group of charts where the TPR is 100%, FPR is 0% and crash proportion is 22.74%. This means that each RSU packet dropping has about 22.74% of crash if the attacker only drops RSU packets and no drops all RSU packets. The goal of side-channel analysis is to cause vehicle crashes with less unnecessary packets dropping. Each scenario caused crashes and did targeting attack.

The FPR are less than 10% for all scenarios. The effective of side-channel analysis is proved. We evaluate the attack scenarios based on crash proportion and FPR.

We firstly compare the attacks based on one type of side channel information: timing side channel attack for the most likely state, timing side channel attack for two most likely states and size side channel attack. As shown in the Figure 4.9, the second and third scenarios have the lowest FPR in second, third and fourth scenarios. With the windows of confidence interval, there's no significant difference of FPR between the second scenario and the third scenario. Moreover, the third scenario also has the highest crash proportion. So we can conclude the third scenario of timing side channel attack for two most likely states is best in side channel analysis based on one type of information.

Then we compare all attack scenarios to find the best attack method for this application. As shown in Figure 4.9, the third scenario and the fifth scenario have the highest value of the crash proportion while the third scenario has much lower FPR than the fifth scenario. So the third scenario is the best in five attack scenarios.

As a conclusion, timing side channel analysis has better performance on predicted states. The attack targeting the packet leaving two most likely timing states worked best to cause crashes while avoid dropping unnecessary packets.

5. Conclusion and Future Work

This thesis focuses on the evaluation analysis of DSRC/WAVE applications. To do this, we set up a DSRC stop light application based on a developed WSMP implement. We sniffed the data through WSMP. The sniffed result of clear-text WSMP data content shows the current implement is unsecure. Lack of security services, such as content encryption, makes it easy for attackers knowing critical car/road information with DSRC equipped devices. Then we performed DoS attack and successfully dropped packets at the communication channel and caused crashes.

Assuming all the security services will be implemented in the future, we did "black box" attack. Hidden Markov Models (HMM) are constructed using sniffed inter-packet timing and packet size side channels, since operational packets would be encrypted. We set up attack simulation to test the HMM predictions of important packet arrival. The simulation result shows the effectivity of the side channel analysis. And timing side channel analysis worked better in the attack experiments.

The DoS result of packet dropping shows neither the application nor WSMP has a detection or prevention mechanism for DoS attack. In DSRC communication, entropy based DoS detection could be a good tool against DoS attack. In DoS attack detection, entropy measures the amount of disorder in the observed data. For example, in this application, the road safety unit (RSU) system could calculate the entropy value of packet rate and packet size. The RSU can also detect abnormal network traffic from vehicle by cooperating with other RSU nearby. The vehicle volume could be estimated according to

the information from other RSU. To prevent DoS attack, DSRC should add the authentication mechanism to the standard.

To prevent side-channel attack, the WSMP of DSRC should improve the packet formatting. For example, it could define the length of packet through WSMP to prevent packet size side-channel attack.

In the future, we will do following work:

1. Collect more data and do the joint side channels analysis;
2. Apply this evaluation approach on more DSRC applications;
3. Test other attack method, e.g. radio signal jamming.

REFERENCES

[1] Caitlin Bettisworth, “Status of the Dedicated Short-Range Communications Technology and Applications: Report to Congress”, U.S. Department of Transportation, July 2015.

[2] J. B. Kenney, "Dedicated Short-Range Communications (DSRC) Standards in the United States," in Proceedings of the IEEE, vol. 99, no. 7, pp. 1162-1182, July 2011.

[3] Laurendeau C., Barbeau M. (2006) Threats to Security in DSRC/WAVE. In: Kunz T., Ravi S.S. (eds) Ad-Hoc, Mobile, and Wireless Networks. ADHOC-NOW 2006. Lecture Notes in Computer Science, vol 4104. Springer, Berlin, Heidelberg

[4] Islam, Mhafuzul & Chowdhury, Mashrur & Li, Hongda & Hu, Hongxin. (2017). Cybersecurity Attacks in Vehicle-to-Infrastructure (V2I) Applications and their Prevention. Transportation Research Record: Journal of the Transportation Research Board. 10.1177/0361198118799012.

[5] J. B. Kenney, "Dedicated Short-Range Communications (DSRC) Standards in the United States," in Proceedings of the IEEE, vol. 99, no. 7, pp. 1162-1182, July 2011.

[6] Ott, L., Longnecker, M. (2001). An introduction to statistical methods and data analysis. Australia: Duxbury.

[7] R. R. Brooks, J. M. Schwier, and C. Griffin, “Behavior Detection Using Confidence Intervals of Hidden Markov Models,” IEEE Transactions on System Man and Cybernetics, Part B: Cybernetics, 39(6), 1484-1492 (2009).

[8] Zhong, Xingsi, Afshin Ahmadi, Richard Brooks, Ganesh Kumar Venayagamoorthy, Lu Yu, and Yu Fu. “Side channel analysis of multiple PMU data in electric power systems.” In Power Systems Conference (PSC), 2015 Clemson University, pp. 1-6. IEEE, 2015.

[9] Zhong, Xingsi & Fu, Yu & Yu, Lu & Brooks, R. & Venayagamoorthy, Ganesh. (2015). Stealthy Malware Traffic - Not as Innocent as It Looks. 110-116. 10.1109/MALWARE.2015.7413691.

[10] R. R. Brooks, Introduction to Computer and Network Security: Navigating Shades of Gray, (2014) CRC Press, Boca Raton, FLA.

[11] R. R. Brooks, S. Sander, J. Deng, and J. Taiber, “Automobile Security Concerns,” IEEE Vehicular Technologies, 4(2), 52-64 (2009).

[12] P. Ryan and S. Schneider, Modelling and Analysis of Security Protocols, Addison-Wesley, Harlow, UK, 2001.

[13] J. Schwier, C. Griffin, and R.R. Brooks, “Zero Knowledge Hidden Markov Model Inference,” Pattern Recognition Letters, 30(14), 1273-1280, (2009).

[14] Al-Kahtani, Mohammed Saeed. “Survey on security attacks in Vehicular Ad hoc Networks (VANETs).” In Signal Processing and Communication Systems (ICSPCS), 2012 6th International Conference on, pp. 1-9. IEEE, 2012.

[15] Lee, Hyeryun, Kyunghee Choi, Kihyun Chung, Jaein Kim, and Kangbin Yim. “Fuzzing can packets into automobiles.” In Advanced Information Networking and Applications (AINA), 2015 IEEE 29th International Conference on, pp. 817-821. IEEE, 2015.

[16] Yang, D., Jiang, K., Zhao, D. et al. Intelligent and connected vehicles: Current status and future perspectives. Sci. China Technol. Sci. 61, 1446–1471 (2018).

[17] L. Yu, J. M. Schwier, R. M. Craven, R. R. Brooks and C. Griffin, "Inferring Statistically Significant Hidden Markov Models," in IEEE Transactions on Knowledge and Data Engineering, vol. 25, no. 7, pp. 1548-1558, July 2013, doi: 10.1109/TKDE.2012.93.

[18] C. Lu, J. M. Schwier, R. M. Craven, L. Yu, R. R. Brooks and C. Griffin, "A Normalized Statistical Metric Space for Hidden Markov Models," in IEEE Transactions on Cybernetics, vol. 43, no. 3, pp. 806-819, June 2013, doi: 10.1109/TSMCB.2012.2216872.

[19] H. Bhanu, J. Schwier, R. Craven, R. Brooks, K. Hempstalk, D. Gunetti and C. Griffin, "Side-Channel Analysis for Detecting Protocol Tunneling," Advances in Internet of Things, Vol. 1 No. 2, 2011, pp. 13-26. doi: 10.4236/ait.2011.12003.

[20] J. M. Schwier, R. R. Brooks and C. Griffin, "Methods to Window Data to Differentiate Between Markov Models," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 41, no. 3, pp. 650-663, June 2011, doi: 10.1109/TSMCB.2010.2076325.

[21] Craven, Ryan, "Traffic Analysis of Anonymity Systems" (2010). All Theses. 837. https://tigerprints.clemson.edu/all_theses/837

[22] https://en.wikipedia.org/wiki/IEEE_802.11p

[23] C. R. Shalizi, K. L. Shalizi, and J. P. Crutchfield, “An algorithm for pattern discovery in time series,” arXiv:cs.LG/0210025, November 2002. [Online]. Available: http://arxiv.org/abs/cs. LG/0210025

[24] Song, Yingbo, Salvatore J. Stolfo and Tony Jebara. “Markov Models for Network-Behavior Modeling and Anonymization.” (2011).

[25] Boukhtouta, A., Mokhov, S.A., Lakhdari, N. et al. Network malware classification comparison using DPI and flow packet headers. J Comput Virol Hack Tech 12, 69–100 (2016). https://doi.org/10.1007/s11416-015-0247-x