Vehicle Tag Information Update on Roads using RFID
Kavitha.D,1 Karishma.V2 Bhakyalashmi.R3 K.Tamilarasi4 Scholars1,2,3 Assistant professor4 Department of Computer Science and Engineering Velammal Institute of Technology, Ponneri, Thiruvallur, India

Abstract:
The Point of the venture is to concentrate on the issue of refreshing RFID (Radio-recurrence distinguishing proof) labels in RSR (RFID frameworks on streets). The new data and refresh status of these grouped labels can be perused by vehicles cruising by. These vehicles can trade such data with each other by means of vehicle-to-vehicle (or vehicle-to-framework) (V2V or V2I) correspondences. Our broad reproduction comes about show that our proposed data dispersion plan can successfully refresh the labels inside a specific time oblige under different situations. RFID innovation has been generally utilized as a part of our everyday lives because of its accommodation and minimal effort. In RSR, extensive scale aloof RFID labels with street related data are sent on street surfaces or roadside units. A vehicle with an installed RFID peruse can procure street data by means of perusing from these RFID labels.

Keywords: Tag update, Radio-frequency identification (RFID), Reader.

I. INTRODUCTION

The Aim of the project is to focus on the problem of updating RFID (Radio-frequency identification) tags in RSR (RFID systems on roads). The emergent information and update status of these clustered tags can be read by vehicles passing by. These vehicles can exchange such information with each other via vehicle-to-vehicle or vehicle-to-infrastructure (V2V or V2I) communications. Our extensive simulation results demonstrate that our proposed information diffusion scheme can effectively update the tags within a certain time constrain under various scenarios. RFID technology has been widely used in our daily lives due to its convenience and low cost. In RSR, large-scale passive RFID tags with road-related information are deployed on road surfaces or roadside units. A vehicle with an onboard RFID reader can acquire road information via reading from these RFID tags. Radio Frequency Identification (RFID) technology has been widely used in our daily lives due to its convenience and low cost. RFID tags are deployed on the road surface or roadside units, and the readers are mounted on vehicles are designed to provide road-related information to vehicles. Specifically, RFID tags store the road information, and vehicles acquire the information by reading the tags as they pass nearby. An example of these applications is Electronic Toll Collection (ETC). The primary objectives of our proposed system are to update all the tags as soon as possible while limiting the communication overhead. Radio Frequency Identification innovation has been generally utilized as a part of our day by day lives because of its accommodation and minimal effort. RFID labels are sent out and about surface or roadside units, and the pursuers are mounted on vehicles are intended to give street related data to vehicles. In particular, RFID labels store the street data, and vehicles get the data by perusing the labels as they pass adjacent. A case of these applications is Electronic Toll Collection. The essential goa of our proposed framework are to refresh every one of the labels at the earliest opportunity while constraining the correspondence overhead. There are two types of RFID application in vehicle networks. For Type-I applications, RFID readers are installed on roadside units, and tags are mounted on vehicles. Sample Type-I applications include ETC, vehicle access control, and automatic vehicle identification. In Type-II applications, RFID readers are attached to vehicles, and RFID tags are deployed on road surfaces or roadside units. A recent example of Type-II applications is RSR. The feasibility of communications between readers and tags in high-speed vehicle networks is analyzed. In addition, the tag reading latency of RFID readers installed on vehicles under a wide range of speeds is studied. Zhangetal proposed a method that could be adopted to identify the defective tags in RSR. However, to the best of our knowledge, the problem of RFID tag update in RSR has not been studied in the literature. Our proposed tag update scheme is inspired by the information diffusion problem in social networks and the information collection problem in wireless sensor networks (a reverse direction to the tag information update problem). Next, we summarize the work that is the most relevant to ours in these two areas. There are many works that study the problem of information diffusion in social networks. The most relevant work that is related to ours Specifically, adopted the idea that information spreads much faster among a group of people with frequent interactions than that of a relatively isolated group. This work inspires the design of our proposed tag information update scheme. However, RFID tags are unlike people in that passive RFID tags cannot talk with each other directly. As a result, the
existing information diffusion techniques for social networks cannot be directly applied to the tag update problem in RSR. We also consider the reverse data collection procedure in wireless sensor networks (i.e., sinks sending information to sensors) to be similar to the tag information update problem. For example, Chen et al. studied the multiple-sink data collection problem in a large-scale sensor network and proposed an approximation algorithm to minimize data collection latency. A novel approach to significantly improves the utilization of available network resources for information collection is proposed. Furthermore, flooding-based algorithms that aim to transmit data from one source to all other nodes have been studied. However, the proposed information diffusion solutions to the sensor network cannot be directly. At the dawn of a new computing era is parallel computing. Driven by the efforts of hardware manufacturers, the capabilities of parallel computing will become available in almost every consumer device and not only be used in large distributed systems. When we talk about the advantages of multicore systems, we often forget that the former only come with a deep understanding of what parallelism means. It is the responsibility of programmers to move from an age where sequential programming seemed to be everlasting to an age of natively parallel code. We present an overview of the struggles and issues in the parallel computing revolution and provide insight into the efforts of research groups that try to overcome most of those problems. We also discuss the possible future of parallel computing and describe how the principles of computer science can be applied in this area. The use of “many core” (i.e. with a comparably high number) over multicore systems is undeniably the only future of high performance computing, because it allows for cheaper and more power-efficient operability. Parallel computing will take place on the desktop PC as well as on large grids of distributed machines. The introduction of parallel architecture however came without deep knowledge in the field of parallel programming. Problems arise from the fact that programmers need to understand the implications of parallelism and how to transform common tasks into parallel programs. This is essential in order to optimize the usage of available computational power, since sequential programs will not benefit from the parallel architecture. When thinking about making use of parallel processing power, mostly bottom-up approaches are considered. Since every architecture is specific in its implementation of operations, memory access and caching, different compilers will have to be written. Generic optimizations need to be included at this low-level stage to allow for all possible applications that may run on this specific platform. Another way of addressing the lack of compelling applications for parallel architectures is to develop them in a top-down fashion. Computational problems nowadays are inherently driven by an underlying business case since the applications are human-centered and potentially groundbreaking. The ideas for applications are elaborated upon first and then the appropriate platform is chosen.

II. RELATED WORK

A RFID system is composed of RFID tags and RFID readers. A RFID tag stores data, and a RFID reader accesses the tag to collect the data through wire-less communications. There exist two types of RFID tags: active tags, which contain power modules to support wireless communications, and passive tags, which power their transmissions through the energy absorbed from the radio waves of the RFID readers. Compared to active RFID tags, passive RFID tags are easier to maintain as they do not need power, and their cost can be as low as several cents. Therefore, passive RFID tags are more appropriate for applications that require a large number of tags. Traditionally RFID tags were designed for commercial applications to replace the bar codes for asset counting and identification. One important challenge in such applications is how to handle the read collision problem that occurs when one or more RFID readers query multiple RFID tags roughly simultaneously in a small area. As a result, most existing research focuses on anti-collision protocol design to schedule the reader’s read requests and the tag’s responses. Hence read collision is not possible as our design guarantees the one-to-one coupling of a RFID reader and a tag in a restricted area. RFID systems have been deployed for VANETs, in which RFID tags are installed on vehicles while RFID readers are deployed on stationary infrastructures.

III. RFID SYSTEM ON ROADS

The RFID System on Roads (RSR) is designed to support future intelligent vehicular applications. It consists of RFID tags, RFID readers, Information Processing Units (IPU), and Information Sharing Units (ISU). RFID tags are deployed on road surfaces, and RFID readers are installed at vehicles. Vehicles obtain information from embedded sensors and RFID tags, which will be processed by IPU and then broadcast to other vehicles and base stations via V2V/V2I (Vehicle to Vehicle/ Vehicle to Infrastructure) communications employing DSRC (Dedicated Short-Range Communication). RSR is a new platform that aims to provide accurate and timely road-related information to drivers. Such information may be stored in passive RFID tags that are deployed in the center of a lane. A vehicle can acquire the information via an onboard RFID reader when the vehicle passes by the tags. Once the vehicle obtains the information, it spreads the information to other RFID tags in two different ways. First, the vehicle can directly update the RFID tags that do not contain the information and are within these communication distance (CD). Second, the vehicle can share the information among other vehicles via V2V and/or V2I communications. Subsequently, the other vehicles can directly update their neighboring RFID tags using the first approach.

1. STORAGE SYSTEM

We divide the RFID tag’s storage space into the following five areas:

1) read-only; 2) static information; 3) update information; 4) update status; and 5) digital signature.

The read-only area may store permanent information such as factory number, tag configuration, etc. The static information area can store information that typically does not change. Examples of such information include tag location, lane direction, and speed limits. The update information area is used to hold the information to be updated such as accidents and traffic congestion. The update status area may contain the status of the tag and its cluster. The digital signature area contains a digital signature to authenticate the communications between the tag and readers.
Figure 1. Cluster tags

IV. PROBLEM DESCRIPTION AND PRELIMINARIES

In this section, we first describe the system model and adversary model. We then formulate the problem of deterministic clone detection for anonymous RFID systems.

A. System Model

Anonymous RFID system. We consider an anonymous RFID system in which readers cannot retrieve tag IDs from the server or tags. Such a strict requirement is necessary for protecting tag IDs’ associated privacy and enabling privacy-sensitive applications. Following a canonical RFID system architecture, the system consists of a backend server, a reader, and a number of objects each affixed with a tag. For inventory control, a tag has a unique ID across the system and represents the object that it is attached to. When multiple readers are in use for covering all tags and are well synchronized, they can be logically treated as one. The reader communicates with the server via a wired or wireless yet secure channel while with tags via a wireless channel. Normally, the server does not directly communicate with tags; it dictates what monitoring operations the reader needs to execute over tags. Most existing monitoring operations entail the ID information of tags. For example, tag IDs are necessary for detecting missing tags whose IDs’ correspond to no response and for detecting clone tags whose IDs appear at different locations. However, we do not assume the knowledge of tag IDs in anonymous RFID systems. Only a tag itself knows its own ID; tags use IDs to determine when to respond under a certain anticollision protocol.

B. Assumptions

To explore deterministic clone detection solutions for anonymous RFID systems, we assume that 1) ID cardinality (i.e., the number tag IDs) is known, and 2) opcodes are in use for readers to inform tags what data to transmit. First, we assume the knowledge of ID cardinality rather than ID specifies. The reader can retrieve ID cardinality from the server. In an intact system, ID cardinality is equal to tag cardinality as each tag has a unique ID. However, when clone tags exist, tag cardinality exceeds ID cardinality. This observation motivates an intuitive clone detection solution. Second, we assume that to fulfill various monitoring operations, the reader adopts opcodes to query corresponding data from tags. Such opcodes facilitate most established RFID protocols, whether or not being explicitly emphasized. We observe that varying tag responses is necessary for protocol efficiency rather than protocol efficacy. Its purpose is to require as few data as possible from tags. For example, a 10-bit string with CRC embedded is sufficient for distinguishing an intact response from a collided one while one bit is sufficient for transmitting binary information (e.g., tag presence, battery’s residual capacity). Transmitting more data than necessary therefore brings no benefits but efficiency degradation. For conciseness, we will directly indicate what data to transmit without emphasizing the use of opcodes during protocol design.

C. Adversary Model

Launching a cloning attack, the attacker first compromises genuine tags and then replicates their data to clone tags [2]. Since clone tags hold all valid data (e.g., IDs, keys) of compromised tags, clone tags can easily impersonate genuine tags. Detecting clone tags is thus practically important as most RFID applications equate tag genuineness to tagged objects’ authenticity. However, clone tags are more challenging to detect than are counterfeit tags holding valid IDs but forged keys. Common solutions for detecting counterfeit tags such as cryptographic authentication cannot conquer clone tags, which pass authentication using their copies of valid keys of
compromised tags. We assume that clone tags faithfully respond to reader queries. In other words, clone tags cannot selectively respond to evade detection. As with detecting counterfeit tags, a detection protocol fails to uncover clone tags if they keep silent to detection queries. There are two cases in which clone tags may violate the assumption of faithful response. We observe that in both cases the assumption could be satisfied with or without extra efforts. First, the attacker manipulates clone tags while eaves dropping the communication between the reader and tags. When the eavesdropped query is likely for clone detection, the attacker informs clone tags not to send responses. If this is the case, we can leverage advanced jamming techniques that can jam the attacker’s communication without interfering the communication between the reader and tags. Second, clone tags may be made sophisticated enough such that they themselves decide to which queries not to respond. We tackle such sophisticated clone tags by making reader queries for clone detection indistinguishable from that for other monitoring operations. For example, when the clone detection protocol requires a 1-bit response from tag, clone tags can hardly infer whether the query verifies its genuineness or its presence in it. We suggest that strategically interweaving different monitoring operations could obfuscate not only sophisticated clone tags but also sophisticated attackers; this is not the focus of this paper.

D. Problem Formulation

Consider an anonymous RFID system with ID cardinality \( N \) and at least one cloned ID that associates with some clone tag(s). The deterministic clone detection problem is to detect the existence of clone tags with certainty, as fast as possible, without tag IDs as a priori. Such a formulation enforces the following three requirements for designing desirable clone detection protocols.

Anonymity: To enable anonymous RFID systems, clone detection protocols cannot be built upon the values of tag IDs. Most existing clone detection protocols, however, require tag IDs as a priority and thus provide few hints about detecting anonymous clone tags. We will explore new features for detecting clone tags while preserving tag anonymity.

Accuracy: Clone tags are hardly distinguished from genuine tags by the reader and thus significantly threaten security-relevant applications such as RFID access control. For these applications, clone detection protocols should accurately ascertain the existence of clone tags if any. The only related work for detecting anonymous clone tags is, however, a probabilistic design. We will propose new techniques for deterministic detection of anonymous clone tags.

Scalability: RFID is experiencing an ever-increasing demand for more pervasive applications. For example, as Lux Research predicts, China’s RFID card/tag market volume will expand to 2.11 billion units. Making clone detection protocols scalable to large-scale systems, we borrow ideas from related work. The key idea is to collect as a few data from tags as sufficient for clone detection. This way, transmitting fewer data yields higher protocol efficiency and thus scalability.

V. PROPOSED SYSTEM

In this paper, we propose a novel cluster-based information diffusion scheme for quickly and accurately updating on-road RFID tags with emergent road information. To the best of our knowledge, our work is the first to focus on the problem of updating RFID tags in RFID systems on roads (RSR). RSR is a recently developed platform for improving transportation safety and efficiency. It can provide unique safety features for hazard driving environments (such as ice/snow-covered roads, storm, or fog), where other intelligent technologies cannot provide satisfactory road information. To be specific, our proposed scheme categorizes RFID tags into clusters, which define the groups of tags that may mutually help in their information updates. The emergent information and update status of these clustered tags can be read by a vehicle passing by. The vehicle can exchange this information with other vehicles via vehicle-to-vehicle (V2V) [vehicle-to-infrastructure (V2I)] communications. After synthesizing all the received information, a vehicle can update the nearby tags and spread the information accordingly. The size of a cluster will affect the information update speed and the communication overhead. Our extensive simulation results demonstrate that our proposed scheme can successfully update tags and vehicles with the emergent information in a timely fashion.

ADVANTAGE

- Update Tags in timely manner
- Road Safety
- Update Emergent Information.

VI. SYSTEM DESIGN

Here, we describe our system design and our proposed algorithms for updating RFID tags in RSR. The primary objectives of our proposed system are to update all the tags as soon as possible while limiting the communication overhead.

a) Tag Updating-Algorithm

Since RFID tags cannot communicate directly with each other, a vehicle needs to function as a relay to update tags and their CUSTs. To make the tag-updating process efficient, we introduce a Tag Status Synthesis Table (TSST) for each vehicle, as shown in Fig. 3. This table includes an odd number of rows of tags. The size of the TSST is typically larger than that of the CUST. Each entry in the TSST represents the update status of the corresponding tag. When a vehicle passes a tag (e.g., \( T \)), the vehicle reads the CUST of tag \( T \). After reading the CUST, the vehicle first updates its TSST by adding the new row of tags and deleting the obsolete row of tags. Moreover, the vehicle updates its TSST entries by correlating the information obtained the tag’s CUST using Algorithm 1 presented below. Specifically, there are following four possible cases for information correlation. We refer to an entry in the CUST as \( C_{k,j} \) and its corresponding TSST entry (if exists) as \( T_{i,j} \). In particular, \( C_{k*1} \) and \( T_{i*1} \) represent the status of the tag that is currently being read (\( T \) in this example) in the CUST and the TSST, respectively.

Algorithm 1 Tag Updating

1: // A vehicle has successfully read a tag.
2: if \( C_{k*1} == 0 \) and the vehicle has the emergent information then
3: Write the emergent information to the tag and set the tag as in update status.

4: Ti\*j\*= 1;
5: for each pair of Ckl and Tij do
6: Ckl = Tij;
7: end for
8: else
9: if Ti\*j\*= 0 then
10: Copy the emergent information from the tag to the vehicle;
11: end if
12: for each pair of Ckl and Tij do
13: if Ckl > Tij then
14: Tij = Ckl;
15: else
16: Ckl = Tij;
17: end if
18: end for
19: end if
20: // Check tag status in CUST
21: if all Ckl > 0 & Ckl < 2 then
22: Ckl = 2;
23: Ti\*j\*= 2;
24: end if
25: Write CUST to the tag if CUST has been updated;
26: Broadcast TSST if TSST has been updated;

1) Neither the vehicle nor the tag has any emergent information:
In this case, there is no update necessary. In addition, the vehicle
does not send its TSST to other vehicles.
2) The vehicle has the emergent information, but not the tag
The vehicle writes the emergent information to the tag and sets the
tag in the update status (i.e., Ckl = Tij). Moreover, the vehicle
copies the values of the entries in its TSST to the corresponding
entries in the tag’s CUST. Finally, the vehicle sets Ti\*j\*= 1 in
its TSST.
3) The tag has the emergent information, but not the vehicle
(Algorithm 1: Lines 9–11): In this case, the vehicle acquires the
emergent information from the tag. Next, the vehicle copies the
entries in the tag’s CUST to the corresponding ones in its TSST.
4) Both the vehicle and the tag have the emergent information
(Algorithm 1: Lines 12–18): We synchronize the elements in the
CUST with the corresponding ones in the TSST to increase the
vehicle’s and the tag’s awareness of the update status of the tags
that are in both the TSST and the CUST.

Once all the actions of any of the four aforementioned cases are
complete, the vehicle needs to broadcast the updated TSST and
the updated CUST. If there is no update, the vehicle does not
broadcast these tables. In addition, if the tag’s CUST indicates
that all the tags have obtained the emergent information, the
vehicle sets the tag’s status back to normal.

b) V2V Information Exchange

To assist in the information propagation and reduce the
communication overhead, vehicles need to exchange the
following information:
A. the current emergent information;
B. the vehicle’s current location (i.e., the location of the
most recently read tag);
C. the vehicle’s current TSST; and
D. the set of tags that need to be updated with the
emergent information. After receiving such information, vehicles
synthesize this information to update their TSSTs using
Algorithm 2.

ALGORITHM 2

V2V Information Exchange

1: //Vehicle vm has received TSSTn.
2: if vm does not have the emergent information then 3: //vn has
the emergent information
4: Copy the emergent information to vm;
5: end if
6: for each pair of am ij and an ij do
7: if am ij < an ij then
8: //vn has the updated information for an ij
9: am ij = an ij;
10: end if
11: end for
12: for each am ij do
13: if am ij == 2 then
14: Change the value of the entry in CUSTm ij ∩ TSSTm to 1 if
it was zero;
15: else if CUSTm ij ⊆ TSSTm then
16: //vm has all the tags’ information in CUSTm ij
17: if all the entries in CUSTm ij is nonzero then
18: //all the tags in CUSTm ij have been informed
19: am ij = 2;
20: end if
21: end if
22: end for
23: Broadcast TSST if TSST has been updated;

VIII. SYSTEM ARCHITECTURE

The system architecture design has the following modules in the
diagram, it show the exact scenario of the implementation of
RFID tag on roadways. Some of the modules are listed below,

Modules of the Architecture:
✓ Tag and Reader Formation
✓ Reader and Tag information update
✓ Reader to Reader information exchange
✓ Tag to Reader communication

MODULES:

1. Tag and Reader Formation
Tag is the place where the emergent information has to be
updated in a timely manner. And reader stands for a vehicle like
thing that is used to pass the emergent information to the Tags.
Here we assumed that the distance of the road be 400m and 3
tags are deployed each at 100m location and interval. The road is
a single lane and two readers are allowed to pass through the
lane one after the other.
Figure 3. Architecture Diagram

2. Reader to Tag Information update
Here we assumed that the Reader1 starts at 0m location with emergent information. When it reaches 100m, it will update the tag located at 100m and when it reaches 200m, it will update the tag located at 200m. But with emergent information when it tries to reach Tag3, it halted due to some problem.

3. Reader to Reader Communication
Here another Reader2 starts from the same location where Reader1 starts. When it reaches 100m, it will update the information from the tag located at 100m and when it reaches 200m, it will update the information from the tag located at 200m. When it reaches 250 m where the Reader1 stopped. It will update the new emergent information from the stopped Reader and travels towards Tag 3.

3. Tag to Reader Communication
Here another Reader2 which updated the emergent information from the Reader1 travels towards the Tag located at 300m and updates this information on it. Own all the 3 tags has been successfully updated in a timely manner.

VIII. PERFORMANCE ANALYSIS

Performance Requirements
The performance of the project, to execute this project on LAN or wifi communication channel. So we need to one or more than machine to execute the demo. Machine needs the enough hard disk space to install the software and run our project.

Safety Requirement
1. The software may be safety-critical. If so, there are issues associated with its integrity level.
2. The software may not be safety-critical although it forms part of a safety-critical system. For example, software may simply log transactions.
3. If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level.
4. There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable.
5. If a computer system is to run software of a high integrity level then that system should not at the same time accommodate software of a lower integrity level.
6. Systems with different requirements for safety levels must be separated.
7. Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

Security Requirements
Do not block the some available ports through the windows firewall.

Software Quality Attributes
Functionality: are the required functions available, including interoperability and security.
Reliability: maturity, fault tolerance and recoverability
Usability: how easy it is to understand, learn, and operate the software system
Efficiency: performance and resource behavior.
Maintainability: Maintaining the software.
Portability: can the software easily be transferred to another environment, including install ability.

IX. EVALUATION RESULTS

We illustrate our evaluation results in terms of all-tag informed time, all-tag updated time, and the number of uninformed vehicles as follows.

TABLE I SIMULATION PARAMETERS

A. All-Tag Informed Time
The all-tag informed time measures the speed of information diffusion among tags. Fig. 5 reports the all-tag informed time for various TSST sizes and traffic densities. First of all, we can see that the all-tag informed time decreases when the CD increases. This is because more vehicles can be involved in the tag-informing process under a longer CD. In addition, we can see
that the size of the TSST has limited impact on the all-tag informed time when we fix the traffic density and CD. In contrast, the role of the traffic density on the all-tag informed time is mixed: 1) When the traffic density is very low or very high, the all-tag informed time is long because either there are not enough vehicles to update the tags, or there are too many collisions due to excessive V2V communications; 2) when the traffic density is in a medium range (e.g., 10–30 vel/km), the all-tag informed time is quite low (about 10 s).

**B. All-Tag Updated Time**

The all-tag updated time is usually longer than the all-tag informed time because the former includes the time to set tags back to the normal status. Fig. 6 shows the all-tag updated time for different TSST sizes and traffic densities. Note that the case of TSST size being 3 does not appear in the figures because the actual all-tag updated time of the case exceeds the maximum allowable time. In general, we observe that the TSST size, the traffic density, and the CD have a similar impact on the all-tag updated time compared with that of the case for the all-tag informed time. Specifically, it can be seen that a smaller TSST size (particularly, TSST = 11) can lead to poor performance in low-traffic-density cases. With the increase in the TSST size, the all-tag updated time decreases as vehicles can obtain more tag update status exchange. However, if the TSST size grows too large (e.g., 39), it can incur a higher all-tag updated time because of excessive communication overhead.

<table>
<thead>
<tr>
<th>TABLE I</th>
<th>SIMULATION PARAMETERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Road length</td>
<td>500m</td>
</tr>
<tr>
<td>Road width</td>
<td>20m</td>
</tr>
<tr>
<td>Number of lanes</td>
<td>4</td>
</tr>
<tr>
<td>Traffic density</td>
<td>1, 3, 5, 10, 25, 50, 100 vel/km</td>
</tr>
<tr>
<td>Communication distance</td>
<td>50, 100, 200, 300m</td>
</tr>
<tr>
<td>Maximum speed</td>
<td>120 km/h</td>
</tr>
<tr>
<td>Tag’s interval</td>
<td>10m</td>
</tr>
<tr>
<td>MAC protocol</td>
<td>802.11</td>
</tr>
<tr>
<td>Packet loss probability</td>
<td>15%</td>
</tr>
</tbody>
</table>

**Figure 4. simulation table**

**C. All-Tag Updated Time with Active Request**

We notice that our proposed algorithm suffers when the traffic density is low. To address this issue, we allow vehicles to actively send out requests to other vehicles to help update uninformed tags. Assume that a vehicle vn has successfully finished updating a tag (i.e., after completing Algorithm 1). If vn finds zero-valued entries in the tag’s CUST, vn broadcasts a request that includes the tag’s location and its CUST. Assume that another vehicle vm receives the request. If an entry in the CUST (e.g., Ckl) corresponds to a nonzero entry in vm’s TSST (e.g., Tm ij), vm runs the following algorithm to generate a response.

**X. CONCLUSION**

Thus our proposed scheme can successfully update tags and vehicles with the emergent information in a timely fashion. Note that we assign higher priority to communications between tags and vehicles than the communications between vehicles. This is because the contact time between a tag and a vehicle can be very short when the vehicle’s speed is high. This project will update the information in a timely manner. This will do updating in the single lane i.e. using only one tag to update the all kind of information in an accurate time. The most important thing in the world is time considering this as a major factor the project has been build in a efficiently time manner. This system not only update the tags in time also provide the reader to update their own kind of information (dynamic information) in the correct time. Thus our project focuses on enhancing transportation safety.

**XI. FUTURE ENHANCEMENT**

In the future, the idea of supporting real-time applications which will be really useful to the society and not only for safety measures it will also concern about the security of the vehicle. Though all system are available but in a separate manner in these enhancement all such types of events are too be occurred in one system. This would be a multiple-event update system by adding a field into the tag’s status for each event.

**XII. REFERENCES**


