Vehicle Software Update over ICN Architectures

Ali Elgammal[1,2(✉)], Mena Safwat[1,2], Wael Badawy[5], Eslam G. AbdAllah[3], Marianne A. Azer[2,4], and Changcheng Huang[6]

[1] Valeo, Giza, Egypt
me.safwat@nu.edu.eg
[2] School of Information Technology and Computer Science, Nile University, Giza, Egypt
{a.elgammal,mazer}@nu.edu.eg
[3] Master of Information Systems Security Management, Concordia University of Edmonton, Edmonton, AB, Canada
eslam.abdallah@concordia.ab.ca
[4] National Telecommunication Institute, Cairo, Egypt
[5] Badr University, Cairo, Egypt
wael.badawy@buc.edu.eg
[6] Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, Canada
huang@sce.carleton.ca

Abstract. The Internet Protocol (IP) architecture could not fully satisfy Vehicular Ad-hoc Networks (VANETs) needed efficiency due to their dynamic topology and high mobility. This paper presents a technique to update the software of Electronic Control Units (ECUs) in vehicles using Information Centric Network (ICN) architecture. The proposed technique replaces Flashing Over The Air (FOTA) using IP with FOTA using ICN. The importance of FOTA is illustrated as well as the impact of applying the ICN architecture on VANETs. Through our experiments, we compare between the known FOTA over IP and the newly introduced FOTA technique over ICN.

Keywords: Flashing Over The Air (FOTA) · Information Centric Networking (ICN) · ndnSIM · Vehicle software update · Vehicular Ad-hoc Networks (VANETs).

1 Introduction

In the recent years, data transfer rates have rapidly increased. The number of connected nodes exponentially increases to accommodate the demand of users and services such as mobile phones, vehicles and Internet of Things (IoT). The rapid growth in the number of nodes introduces prime challenges such as resource consumption, mobility, security and scalability to the Internet Protocol (IP) addressing network architecture (host to host). One of the key examples is the

L. Foschini and M. El Kamili (Eds.): ADHOCNETS 2020, LNICST 345, pp. 44–54, 2021.
https://doi.org/10.1007/978-3-030-67369-7_4

Vehicular Ad-hoc Networks (VANETs), where the bottleneck or contingency of data results in loss of connectivity and consequently leads to possible vehicle crashes and people fatal injuries. The increase in network latency or centralized network overhead leads to 1.35 million injuries per year [9].

Information Centric Networking (ICN) reduces the latency of VANETs where information is labeled into objects with descriptors. In ICN, the data source node announces the data availability modeled as an object. The requester node establishs a communication channel to the nearest node to transfer an interest request. The interest request propagates through the intermediate nodes until it reaches the source node. Subsequently, the source node sends the required data reversely to the same routing path without needing to know who the requester is. The data is transferred to the neighboring nodes trough the network, until the data reaches the requester node or the nearest node. In ICN, each node in the routing path can cache the data in the content store to satisfy the same upcoming interests in order to decrease the response latency.

Table 1. IP and ICN architecture comparison

	ICN architecture	IP architecture
Caching	Occurs in all the intermediate nodes in routing path	Occurs in servers only
Naming	Hierarchical/flat	Source/destination address embedded in the packet name
Security	Focusing on object security	Focusing on channel security
Data availability in high mobility nodes	Highly associated/achieved with the cached data versions	Achieved with response latency

One of the main features of ICN is the decoupling between a transmitter and a receiver. The ICN uses a message content instead of destination address. Hence, the source/requester node announces and registers a request using the object name and does not use the node address to reduce the network overhead. ICN takes advantage of the distributed control instead of central control to reduce the bottleneck [11]. Table 1 summarizes the main differences between IP and ICN network architectures [4].

An important applications, called Flashing Over The Air (FOTA) has been required recently in automotive industry. In the following sections, we will describe in detail the network challenges, proposed ICN architecture, and FOTA application [1].

The advances in automotive industry use electronic circuits called Electronic Control Units (ECUs) for better control and automatic operations in the vehicle, they connect vehicles through different networks. The connectivity enables data exchange about local information such as road conditions, or weather. Data

exchange between vehicles serves in vehicle ECUs update [8]. Consequently, the VANETs use a large numbers of ECUs with complex hardware and software to run algorithms for Advanced Driver Assistant Systems (ADAS) and autonomous driving. For example, 100 ECUs with 250 embedded/graphic CPUs [14] are used to execute more than 100 million lines of code inside the vehicle for handling vehicle normal operation [15].

ECU's errors, in either software or hardware, result in accidents, injuries and losses. Due to the higher reliability of hardware, the software is a more probable source of defects. Software updates are either regular or on demand. They are necessary to offset the liability of potential errors through vehicle recalls. Currently Original Equipment Manufacturers (OEMs) plan to perform these updates periodically for adding new features, fixing defects and handling security vulnerabilities. ECU software updates occur at the OEM service center. FOTA technology [7] is designed to avoid user inconvenience, unnecessary consumed time, and to manage frequently ECU software update on the fly [2].

FOTA uses software data transfer over wireless communication network and requires the following features: (1) minimal software update time, (2) data integrity, and (3) security goals. The proposed technique of using FOTA over ICN eliminates the need for source node addressing and the specific message routing, which improves the VANETs performance.

In this paper, we run our experiments to evaluate the performance of our proposed technique versus FOTA over IP in VANETs using Named Data Networking Simulator (ndnSIM). The results show that using our proposed technique enhances VANETs performance in all analyzed scenarios. Accordingly, we propose to use the aforementioned ICN advantages in FOTA.

The remainder of this paper is organized as follows. Section II illustrates FOTA and ICN background. Section III demonstrates a platform for FOTA over ICN. Section IV presents FOTA simulation framework. Finally, conclusions and future work are present in Section V.

2 FOTA and ICN Background

This section introduces the FOTA over IP network and the ICN platform in section A and B, respectively. Using the IP network, vehicles receive the updated software through an On Board Diagnostics (OBD) port and then the gateway ECU routes the software data to the required ECU.

The IP network uses a Telematics Control Units (TCU) to interface the vehicle to the network. The vehicles are developed to centralize the remote diagnostics communication via internet [6]. Remote vehicle diagnostics facilitate the vehicle's ECUs software update. When software artifact or vulnerabilities are detected, or new features are added; the ECU supplier updates/releases an updated software version. The following steps describe the positive scenario for FOTA process.

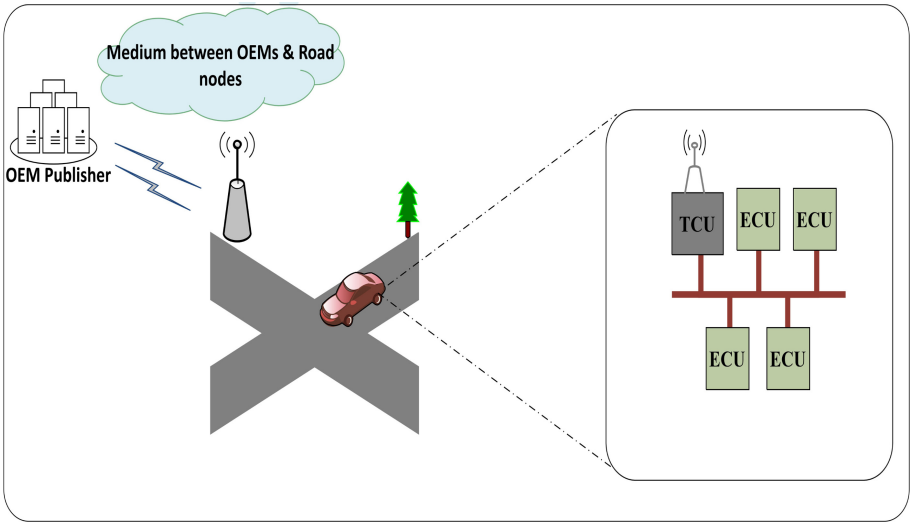

Fig. 1. Flashing over the air, IP network

2.1 Flash over the Air over IP Scenario

As shown in Fig. 1, FOTA over IP is performed by the following steps.

1 - An OEM uploads the new software version to its cloud server.
2 - The cloud server notifies the vehicle with the availability of a new software version.
3 - The main/gateway ECU authenticates the server and accordingly receives the new software data information (metadata).
4 - The main ECU verifies/compares the received metadata with the already flashed version and defines the targeted ECU to be flashed. Then the main ECU decides either to continue the software update or not.
5 - The main ECU receives the new software data, while the vehicle's whole functionalities are running.
6 - The main ECU checks the software integrity and signature, then requests the flash writing from the targeted ECU.

2.2 Information Centric Networking (ICN)

In this section, we present an ICN architecture, specifically Named Data Networking (NDN). An NDN node contains three main data structures. Forwarding Information Base (FIB) memorizes the information about the sent interests. Pending Interest Table (PIT) holds the awaiting information to be satisfied. Content Store (CS) contains the cached data version locally.

The requests and satisfactions in ICN architecture are illustrated in Fig. 2. Node 1 (subscriber 1) sends an interest request "/lib_1/book_ai/2018.txt" to the

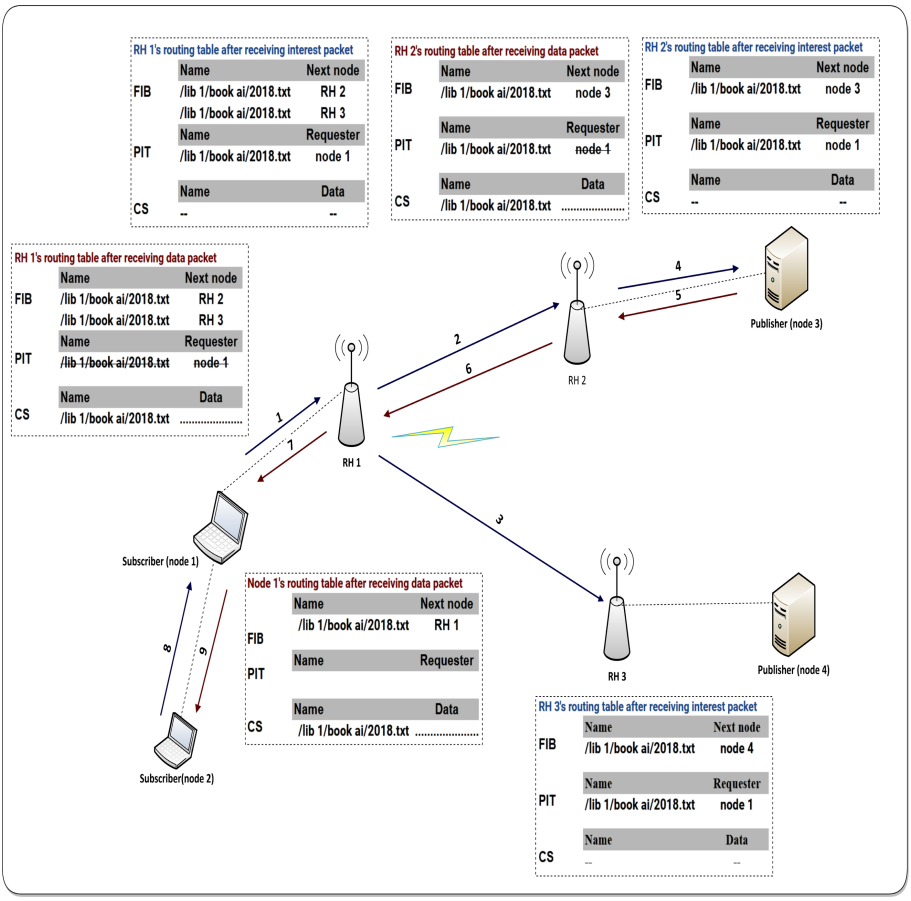

Fig. 2. Data flow in ICN architecture

Resolution Handler 1 (RH_1), RH_1 exports the interest name and searches for the content on its content store. Assuming that RH_1 does not have the requested interest data, RH_1 forwards the interest to RH_2 and RH_3 and stores the interest information in its PIT and FIB.

RH_2 and RH_3 do the same as RH_1, then node 3 (publisher 1) satisfies the "/lib_1/book_ai/2018.txt" interest by forwarding the data back to node 1 on the same routing path. The intermediate nodes between node 3 and node 1 (RH_1, RH_2 and RH_3) store the interest data at the CS then remove the interest request from the PIT.

Node 2 (subscriber 2) sends an interest request "/lib_1/book_ai/2018.txt" to node 1, node 1 responses directly with the interest data from its content store [13].

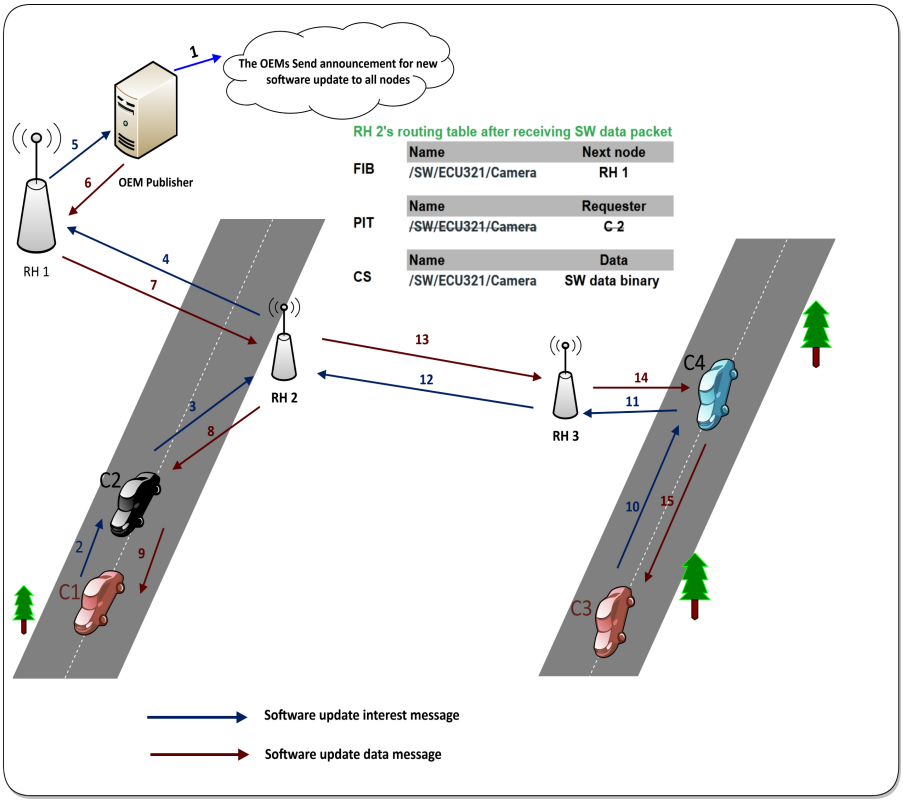

Fig. 3. Software data exchange sequence

3 Proposed Platform for FOTA over ICN

In this section, we propose the vehicle software update through ICN architecture procedure. First, the vehicle manufacturer node announces new software update for specific ECUs in one of their vehicle models. The targeted vehicles in the software update request the data from the nearest node/vehicle and the request propagates towards the data source. New software data is routed back to the requester through the same request path. All the intermediate vehicles/nodes can cache the data to satisfy the upcoming requests. Each OEM has its own technique to identify the software update file naming and versioning schema.

ICN in-network caching refers to caching the response data inside the intermediate vehicles/nodes to decrease the number of requests that are directed to the data provider. Accordingly, the manufacturer node response time is improved. Hence, the software update feature could be delivered in a more efficient manner for the rest of the vehicles.

FOTA over ICN procedure is described in the following steps, and is depicted in Fig. 3. The figure describes the steps while the vehicles are under coverage of ICN network.

1 - The OEM publisher sends an announcement to indicate that there is an updated software version for specific ECU. Vehicle C_1 and C_3 have the targeted ECU.
2 - C_1 sends an interest request to another vehicle C_2.
3 - C_2 adds the C_1's interest to PIT and forwards the request to RH_2.
4 - RH_2 and RH_1 perform the same step as step 3 until interest request reaches the OEM publisher.
5 - The OEM publisher transmits the software chunks to RH_1 then RH_1 transfers the software until reaching C_1. All the intermediate nodes between the OEM publisher and C_1 cache the data in CS.
6 - Once C_1 receives the full software, it checks the validity of the software and then starts the update procedure.
7 - Meanwhile, C_3 requests the updated software from the nearest node C_4.
8 - C_4 and RH_3 repeat step 3 till interest request reaches RH_2, as the RH_2 already has the software update in the CS.
9 - RH_2 forwards the data directly to RH_3.
10 - RH_3 and C_4 save the data in the CS and forward it back to C_3.
11 - C_3 repeats step 6.

Each time a vehicle needs to update the software, there is no need for C3 request to reach the main server (OEM publisher). This reduces the load on the server side and decreases the latency. The latency is decreased as data response takes shorter path and data comes from many nodes to satisfy same interest from different nodes.

4 FOTA Simulation Framework

In this section, our objective is to highlight the flashing time measurements for both IP and ICN networks through simulation experiments using Named Data Network simulator (ndnSIM) [5].

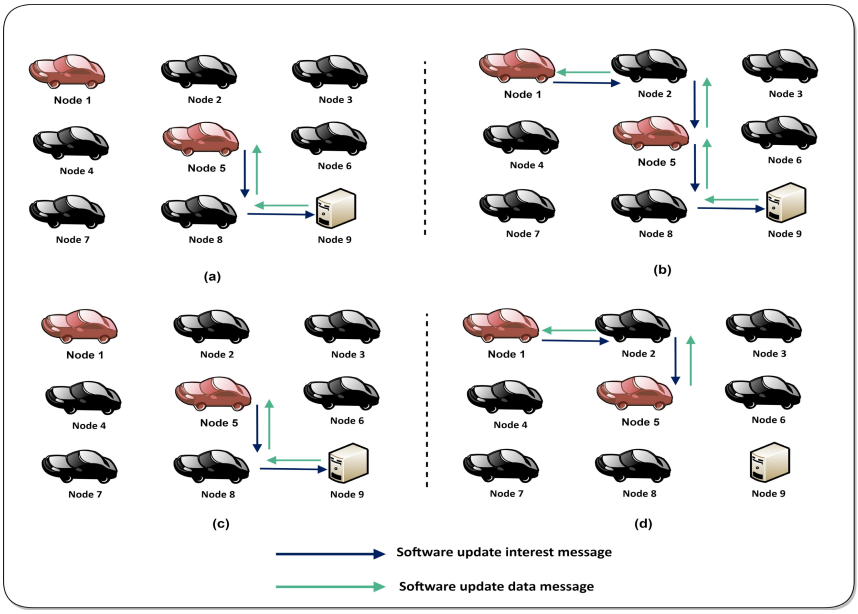

Fig. 4. FOTA Software data exchange for the first scenario 3*3 grid, node 1, 5 act as consumers and node 9 act as a producer. Parts a & b in the figure show FOTA over IP simulation data flow, and parts c & d show FOTA over ICN simulation data flow.

Table 2. Simulated vehicle Software flashing scenarios

Scenario ID	Network topology	Consumer nodes ID	Producer node ID
Scenario_1	Grid of 3*3 nodes	C1:Node_1 C2:Node_5	Node_9
Scenario_2	Grid of 10*10 nodes	C1:Node_1 C2:Node_45	Node_100
Scenario_3	Grid of 15*15 nodes	C1:Node_1 C2:Node_81 C3:Node_129	Node_225

4.1 Simulation Set-Up

ndnSIM is an open source package that uses the Named Data Networking (NDN) protocol stack for the NS-3 simulator [12]. The ndnSIM simulator is widely used to figure out NDN attributes such as FIB, PIT and CS and ICN network [3] [5]. This simulator is based on C++, our simulation parameters are as follow.

- Network Topology: Point to point.
- Routing Strategy: Best route strategy.
- Data Rate: 1 Mbps.
- Packet Size: 1024 Kbytes.

The analyzed case is to transfer one software for specific ECU, transmitted as one segment, and vehicle connectivity is ensured during the software transfer.

Three scenarios are implemented for flashing software with the packet size as descried in Table 2. The producer node “server” has the updated software, and

Consumer vehicles request the updated software from nearest node (in case of ICN) or from the server directly (in case of IP).

4.2 Simulation Results

We used the NetAnim tool to visualize the NS-3 simulator output results by investigating the generated .xml files. In Fig. 4 we present FOTA software data transfer between nodes for the scenarios mentioned earlier.

Figure 4-a & Fig. 4-b show FOTA over IP simulation data flow for the first scenario as follows.

1. - Node 5 “consumer 2” sends a software update request to node 8.
2. - Node 8 forwards the request to node 9 “producer”.
3. - Node 9 transmits the software update to node 8.
4. - Node 8 forwards the software update to node 5.
5. - Meanwhile, node 1 requests the updated software from node 2.
6. - Nodes 2, 5 and 8 repeat step 3 until the request reaches node 9 “producer”.
7. - Node 9 transmits the software update to node 8.
8. - Nodes 2, 5 and 8 repeat step 4 till software update reaches node 1.

Figure 4-c & Fig. 4-d show FOTA over ICN simulation data flow for the first scenario as follows.

1 - Node 5 “consumer 2” sends a software update request to node 8.
2 - Node 8 adds the node 5’s interest to PIT and forwards the request to node 9 “producer”.
3 - Node 9 transmits the software update to node 8.
4 - Node 8 caches the data in CS and forwards the software update to node 5.
5 - Meanwhile, node 1 requests the updated software from node 2.
6 - Node 2 adds the node 1’s interest to PIT and forwards the request to node 5. Node 5 already has the software update in the CS.
7 - Node 5 forwards the software update directly to node 2.
8 - Node 2 forwards the software update to node 1.

In Fig. 5, we present the experimental results for the three cases, where the data transfer in ICN is faster than data transfer in IP. In IP protocol, the data source is centralized in one node “producer”, and hence the producer is the only one that can provide the data. In ICN protocol, the data source is decentralized and the consumers can get the data from the nearest node that has the data in the CS. In scenario 1, the transfer time for the consumer 1 in case of IP is less than by 0.003 and this is due to network protocol and caching time as it is the same node, server, routing path and distance. Using our proposed technique, the data transfer time is reduced by 30% in most of the scenarios.

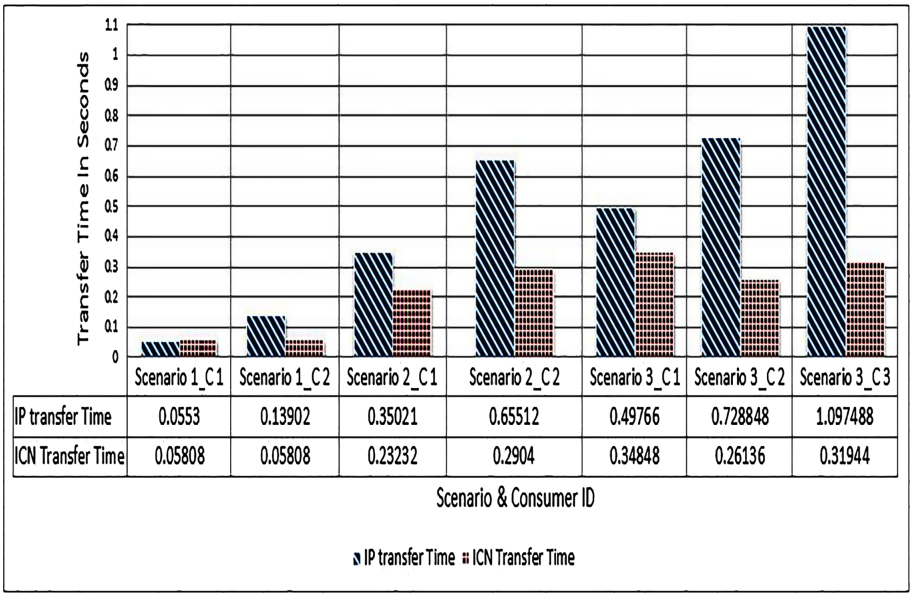

Fig. 5. Experimental results for the three FOTA Scenarios over both IP and ICN networks [1- 3*3 Grid of vehicles "two consumers (C_1 & C_2)" 2- 10*10 Grid of vehicles "two consumers (C_1 & C_2)" 3- 15*15 Grid of vehicles "three consumers (C_1 & C_2 & C_3)"]

5 Conclusions and Future Work

IP networks have several issues such as resource consumption, security, and scalability especially when it comes to vehicles high mobility. With the current type of concerns, researchers take the opportunity to develop new alternative solutions. Information Centric Network (ICN) is one of the proposed alternatives to overcome the IP architecture concerns. This paper presents our novel methodology for providing Flashing Over The Air (FOTA) for vehicle software updates over ICN architectures. This paper proposes and simulates three different FOTA scenarios by changing the number of vehicles, the number of consumers and the network architecture. This simulation is handled using NDN simulator. Results have proven that FOTA over ICN improves many aspects such as software flashing time, network response and servers bottleneck.

In the future, we plan to analyze more scenarios for FOTA over ICN while taking into consideration the application hex data segmentation [10], topology changes and vehicle mobility. This is in addition to analyzing the impact of the possible attacks for both FOTA over IP and FOTA over ICN.

Acknowledgments. The authors would like to acknowledge Valeo Interbranch Automotive Software, Egypt, and Nile university research center.

References

1. Gurmani, M.S., Möller, D.P.F.: Mechanism protecting vehicle-to-vehicle communication. In: Akhilesh, K.B., Möller, D.P.F. (eds.) Smart Technologies, pp. 335–343. Springer, Singapore (2020). https://doi.org/10.1007/978-981-13-7139-4_26
2. Halder, S., Ghosal, A., Conti, M.: Secure ota software updates in connected vehicles: A survey. arXiv preprint arXiv:1904.00685 (2019)
3. Raman, A., Chou, K., Mastorakis, S.: Experimenting with a simulation framework for peer-to-peer file sharing in named data networking. arXiv preprint arXiv:1911.07289 (2019)
4. Yao, H., Li, M., Du, J., Zhang, P., Jiang, C., Han, Z.: Artificial intelligence for information-centric networks. IEEE Commun. Mag. **57**(6), 47–53 (2019)
5. Kato, T., Minh, N.Q., Yamamoto, R., Ohzahata, S.: How to implement NDN manet over ndnSIM simulator. In: 2018 IEEE 4th International Conference on Computer and Communications (ICCC), pp. 451–456. IEEE (2018)
6. Mayilsamy, K., Ramachandran, N., Raj, V.S.: An integrated approach for data security in vehicle diagnostics over internet protocol and software update over the air. Comput. Electr. Eng. **71**, 578–593 (2018)
7. Mirfakhraie, T., Vitor, G., Grogan, K.: Applicable protocol for updating firmware of automotive hvac electronic control units (ecus) over the air. In: 2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), pp. 21–26. IEEE (2018)
8. Onuma, Y., Terashima, Y., Nakamura, S., Kiyohara, R.: A method of ECU software updating. In: 2018 International Conference on Information Networking (ICOIN), pp. 298–303. IEEE (2018)
9. Organization, W.H., et al.: Global status report on road safety 2018. World Health Organization (2018)
10. Teraoka, H., Nakahara, F., Kurosawa, K.: Incremental update method for vehicle microcontrollers. In: 2017 IEEE 6th Global Conference on Consumer Electronics (GCCE), pp. 1–2. IEEE (2017)
11. AbdAllah, E.G., Hassanein, H.S., Zulkernine, M.: A survey of security attacks in information-centric networking. IEEE Commun. Surv. Tutorials **17**(3), 1441–1454 (2015)
12. Mastorakis, S., Afanasyev, A., Moiseenko, I., Zhang, L.: ndnSIM 2.0: a new version of the NDN simulator for ns-3. NDN, Technical report NDN-0028 (2015)
13. Fang, C., Yu, F.R., Huang, T., Liu, J., Liu, Y.: A survey of energy-efficient caching in information-centric networking. IEEE Commun. Mag. **52**(11), 122–129 (2014)
14. Georgakos, G., Schlichtmann, U., Schneider, R., Chakraborty, S.: Reliability challenges for electric vehicles: from devices to architecture and systems software. In: Proceedings of the 50th Annual Design Automation Conference, p. 98. ACM (2013)
15. Charette, R.N.: This car runs on code-IEEE spectrum. IEEE Spectrum: Technology, Engineering, and Science News (2009). http://spectrum.ieee.org/green-tech/advanced-cars/this-car-runs-on-code