Operation STOP: secure itinerary verification for smart vehicle inspections

Henrique F. Santos[0000−0003−1158−9378] and
Miguel L. Pardal[0000−0003−2872−7300]

Instituto Superior Técnico, Universidade de Lisboa, Portugal
{hfigueiredosantos,miguel.pardal}@tecnico.ulisboa.pt

Abstract. A significant part of the transportation of commercial goods is done by road vehicles. Authorities need to conduct inspections on these vehicles to ensure compliance with laws, for example, to verify food safety and correct animal transportation. Most road transportation companies already collect detailed location and vehicle data of their fleet and are able to access it in near real-time. However, authorities do not have access to this data. Inspectors still use road-side stop operations to select vehicles. When a vehicle is inspected, the relevant information is retrieved from paper documents and is not automatically linked to existing information.

In this paper we present the security design and assessment of STOP, a system to improve the inspection of transportation vehicles using mobile devices and location proofs. A system prototype was implemented and evaluated with promising results for better vehicle inspections. The vehicle on-board mobile device reports its GPS location to authorities and, when ordered to stop, interacts with an inspector device to create evidence that the vehicle has been inspected. Additionally, inspectors are able to cross-check information regarding the vehicle and what it is carrying with their authority servers.

Keywords: Smart vehicle inspections · Itinerary tracking · Mobile applications · Location proofs

1 Introduction

Currently, there is a focus on improving the efficiency of the transportation and logistics industries. The usage of mobile devices is referred as one of the approaches for improvement [11] and there are already examples of it.[1] In 2016, heavy road vehicles carried 148.6 million tons of goods in Portugal alone [4]. On the side of governments, the transportation of goods is verified in occasional inspections at road sites.[2] These inspections can be lengthy as inspectors have to

[1] https://www.fleetowner.com/blog/how-mobile-technology-making-waves-trucking
[2] https://www.rtp.pt/noticias/economia/operacao-da-asae-nas-estradas-fiscaliza-transportes-de-mercadorias_v1099919

ask the driver of the vehicle for the documents that describe the transportation, analyze them and then inspect, checking the freight and other legal requirements.

The usage of smartphones may reduce inspection time and, possibly, improve the meticulousness of the process by allowing inspectors to focus on the important details and not on the bureaucracy. The portability of these devices allows for continuous communication with web servers even when at different locations. By knowing the location of circulating vehicles, authorities can identify the most effective location to conduct inspections and also prepare for incoming vehicles.

This paper presents **STOP**, **S**ecure **T**ransport l**O**cation **P**roofs, an information system for improving road transportation inspection with the use of mobile devices. The system can present information regarding the circulation and inspection of vehicles as they happen.

2 Solution Design

The STOP system addresses the entities involved in the transportation of goods and provides functions to enable mobile inspections. We start by describing the roles that need to be played by the entities involved. Then we present the solution architecture and its components.

2.1 Roles

The system considers the roles of *Authority*, *Inspector*, *Company* and *Carrier*. The *Authority* is the entity responsible for the rules for goods inspection in a given country and audits the system to ensure rules are being followed. It defines the user authentication and authorization policy, the required information associated with each vehicle and transport, including the sender and receiver, freight description and planned itinerary. It also sets the policy for selecting vehicles for inspection. The *Inspector* is the agent conducting an inspection at a checkpoint and it is trusted by the Authority for this task. The *Company* is the entity sending goods to another enterprise or individual and it registers the trip details to comply with the rules defined by the Authority. The *Carrier* is the entity actually transporting the goods and is represented by the driver of the vehicle. The Carrier and Company roles may be played by the same entity, as some companies perform the transportation of their goods and do not subcontract an external company for that purpose.

The system uses *pseudonyms* instead of the real identities of the participating entities as it does not need this information to operate. The additional information, such as the location coordinates, is stored as required by the Authority that can audit its correct use.

2.2 System Components

The system architecture has three main components: a server and two mobile applications, illustrated in Figure 1.

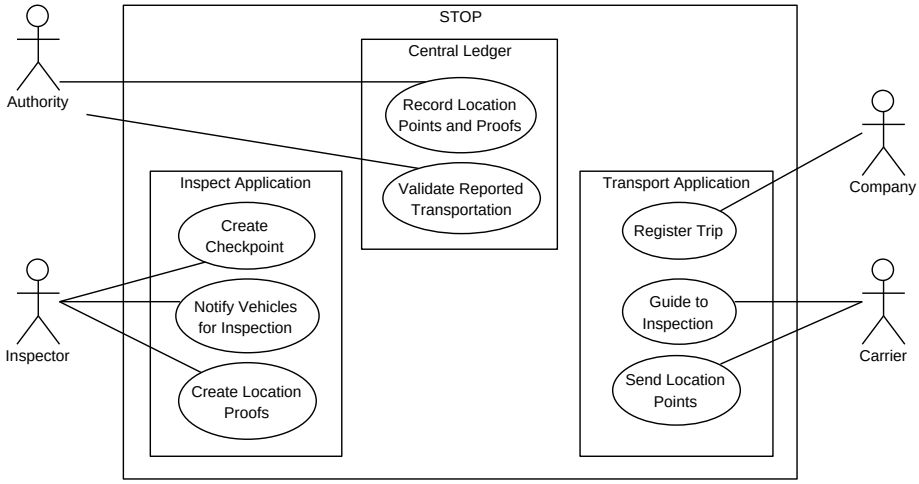

Fig. 1. STOP Entities and Use Cases

The *Central Ledger* is a server that records transportation detail data, location data from each vehicle and inspection data. The server provides a web service endpoint for the mobile applications: *trip* and *inspect*. The choice for a central server reflects the existence of an authority that has oversight over the whole system and requires access to all the data. It can and should be audited to ensure that it is operating in accordance with legal mandates.

The *Transport* application runs on a mobile device inside of the vehicle transporting the goods. It reports the vehicle location to the central ledger via cellular network and communicates with nearby *Inspect* devices when requested. It also sends location proofs generated after the inspection. Section 3.3 details the protocol at an inspection checkpoint.

The *Inspect* application runs at a location to conduct inspections on road vehicles. The application is in constant communication with the central ledger using cellular network. The application communicates with the vehicle device via short-range communication in a way that guarantees that the correct vehicle is inspected. This protocol is also detailed in Section 3.3.

2.3 System Operation

Let us now present how the system will operate. The Company registers an upcoming transportation and the Authority specifies which parameters must be present in the trip registration. By default the system requests the following parameters for registration:

- The fiscal numbers of the entities sending and receiving the goods;
- The location coordinates of the locations where the goods are loaded into the vehicle and when the goods are delivered;

– The description of the freight, indicating the quantity and weight of products;
– The license plate of the vehicle transporting the goods.

Both the actual fiscal numbers and the license plates are replaced with pseudonyms.

When the transportation begins, the vehicle periodically reports its location to the authorities. The goal now is to build a valid *Location Chain* that can later be verified. The chain represents the location positions of the vehicle during the transportation of a set of goods, in chronological order. A location chain item is either a *Location Point* or *Location Proof*, as illustrated in Figure 2. Both contain the signature of the previous location item, as it enables the verification of the sequence of the items in the location chain of the trip. By checking the previous signature in one location item, it is possible to assess if the previous item was modified or is missing, proving protection against record tampering. The main difference between the two is the source of the location position.

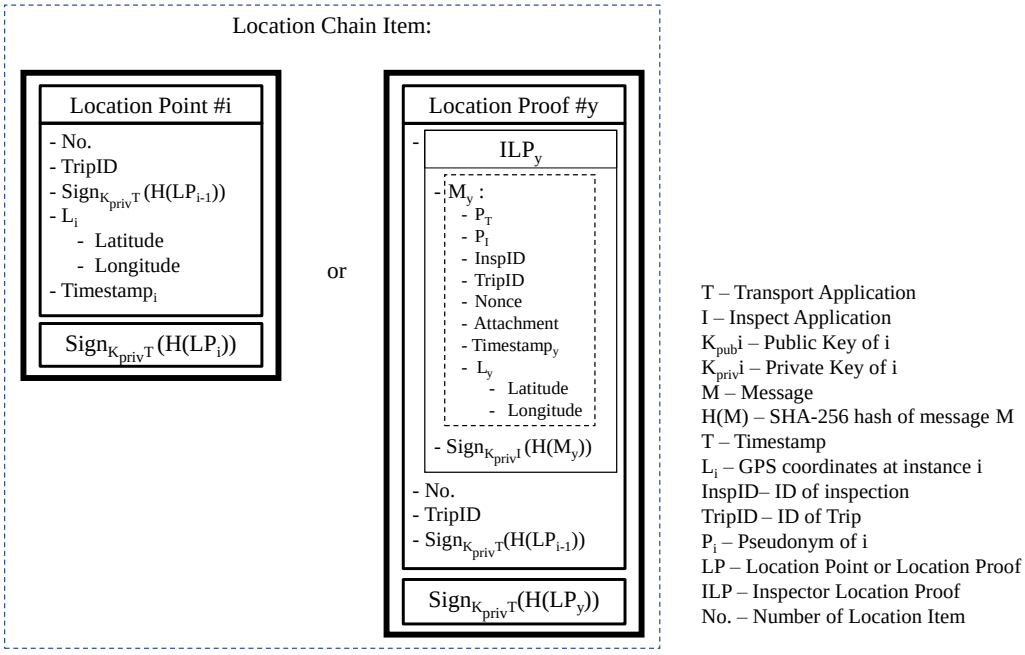

Fig. 2. Types of Location Chain Item

A location point contains the geographic coordinates retrieved by the transporter device GPS, at a time point of the trip. Every item is stored in the location chain instance of the *Transport* device and is sent to the central ledger.

A location proof contains the geographic and time coordinates retrieved by an inspector device at a checkpoint. It is intended to prove that the vehicle was

checked so it is digitally signed by an authorized inspector. The proof contains pseudonyms of the *Transport* and *Inspect* devices, a trip identifier, and a random nonce generated by the central ledger for the occasion. The proof also contains an attachment parameter where an inspector may add text or a picture related to the inspection.

3 Security

In this section we present the attacker model and the design of the security protocols for the STOP system.

3.1 Attacker Model

For our system, we have considered two types of attackers: an *authenticated user* who wants to deceive the system and an *unauthenticated user* who wants to attack the system. We consider the first type as attacker A and the second type as attacker B, with the following intentions:

A1 Report false location point;
A2 Create false location proofs;
A3 Turn off transporter device.
B1 Impersonate an inspector device at a checkpoint;
B2 Impersonate an transporter device at a checkpoint;
B3 Intercept communication between transporter and inspector devices;
B4 Intercept communication between devices and the central ledger.

3.2 Cryptographic Keys and Functions

Each user generates a pair of RSA public and private cryptographic keys for asymmetric encryption and for signature. The public key is stored in the central ledger for encrypted communication and signature validation. The central ledger acts, effectively, as a Certification Authority (CA) for the public keys.

Every message or object requires a digital signature to be considered authentic. A signature is computed by calculating the hash value of the object with the SHA-256 algorithm, which is then encrypted with RSA private key of the device that created the message. The signature is validated by comparing a recomputed hash of the received object with the hash value decrypted using the corresponding public key of the sender.

Additionally, for each inspection, the central ledger generates random *pseudonyms* for the transporter and inspector, used for short-range communication as transient device names. Each pseudonym has its unique key pair generated by the device using such pseudonym and the public key is certified by the central ledger.

Bluetooth is used for short-range communication. The connection setup needs to be fast and seamless to the user. However, in our prototype, we discovered

that Android only provides encrypted Bluetooth communication when the devices are paired[3] and pairing requires user interaction and additional time. To overcome this obstacle, we used insecure Bluetooth sockets instead and implemented encryption at the application level messages. We use *hybrid encryption*, where a message contains the object encrypted with a random AES symmetric key and the key is sent encrypted with the RSA public key of the receiver. The receiver decrypts the AES key with its RSA private key and retrieves the message key. The message is also signed with the private key of the sender to allow the receiver to check the integrity of the received message.

3.3 Communication Protocols

The system defines protocols for communication between the components.

Communication between mobile applications Figure 3 shows the interaction when a vehicle is selected for inspection. The *Inspect* and *Transport* devices obtain the public key certificate of the other device from the central ledger, along with a nonce and a pseudonym for each device. This is necessary to encrypt the Bluetooth communication between these devices and to prevent replay, eavesdropping and tampering attacks.

When the vehicle arrives to the checkpoint, the *Transport* application starts searching for the Bluetooth device announcing as device name the pseudonym of the device of the inspector. When found, the transporter device starts the communication by broadcasting a proof request. The broadcast message is encrypted with the public key of the inspector to guarantee that this message is only decrypted by the inspector. The broadcast message contains the proof request, represented in the figure as PR, and the signature of the hash of the proof request, made with the private key of the transporter, to guarantee that the proof request was created by the transporter. The proof request contains pseudonyms of the devices, the ID's of the inspection and trip, the nonce generated by the central ledger, the timestamp of the transporter device and its GPS coordinates.

When the inspector device receives a message from a device with the pseudonym of the transporter device, it validates if it is a proof request and, if correct, notifies the inspector to conduct the inspection. When the inspection is done, the outcome is reported in a message containing the proof, represented in the figure as ILP, signed by the inspector. The message is encrypted with the public key of the transporter. The message is then sent through the established Bluetooth socket to the transporter device. The inspector device additionally sends a copy of the ILP to the central ledger. The transporter device receives the proof, decrypts and validates it, adds the signature of the previous location item and sends it to the central ledger. If the transporter device did not receive the proof after successfully sending a proof request, it will request the central ledger to

[3] https://developer.android.com/reference/android/bluetooth/BluetoothDevice.html

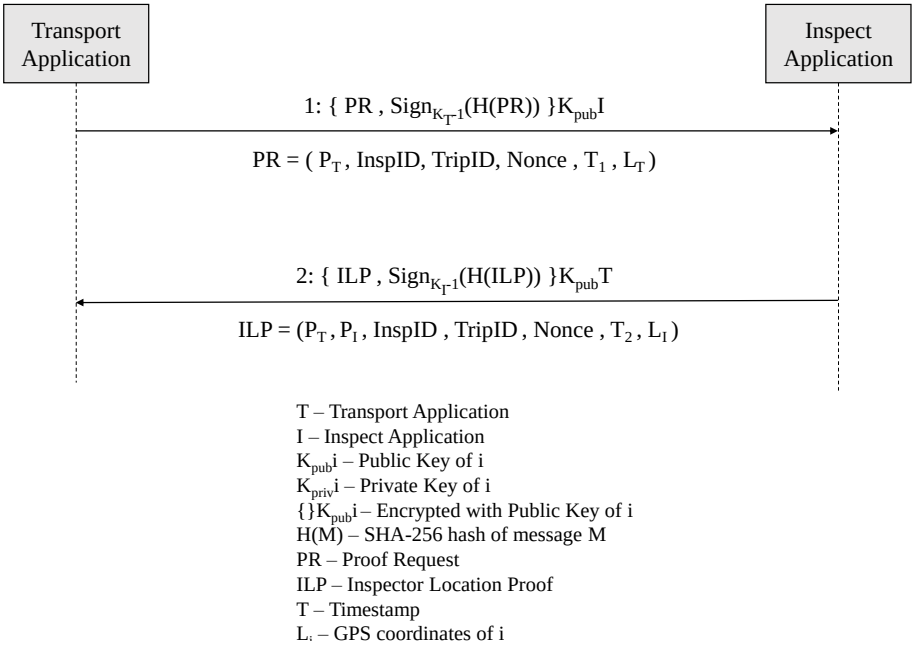

Fig. 3. Inspection Protocol

produce a new nonce and pseudonym for that inspection. Messages with the same nonce, pseudonyms and identifiers are rejected as possible replay attacks.

Communication between applications and Central Ledger The communication between applications and central ledger is done through the RESTful API web service provided by the central ledger via cellular network. This API uses standard HTTP over TLS [4] to protect the messages [6].

The mobile applications keep persistent records of the objects and are able to submit them to the central ledger as soon as possible or later, to tolerate momentary communication faults. However the system requires that devices are able to communicate frequently with the central ledger, as they have to be informed of inspections and the ledger requires the information they submit.

3.4 Assessment

We assessed the security of the system taking into account the malicious intentions of attackers defined in section 3.1.

Regarding the *malicious authenticated user* attacks (attacker A): if the transporter reports false locations (A1) or turns off the device (A3), the inspector will report the vehicles that are not complying with the regulations, such as reporting

[4] https://tools.ietf.org/html/rfc8446

their location, and the company will be held accountable; if the transporter or someone else tampers with proof data (A2), the digital signature can be used to detect the change; if the attacker tries to use an alternative key pair to produce false signatures, he cannot replace the legitimate key certified by the central ledger. Consequently, all the intentions of attacker A are thwarted.

Regarding the *malicious unauthenticated attacks* (attacker B): the messages exchanged over Bluetooth at the inspection checkpoint are protected with confidentiality and integrity mechanisms, as described in section 3.3, preventing the interception (B3). If the attacker tries to impersonate a transporter (B1) or an inspector (B2) the attacker is not able to decrypt received messages and send messages with the correct signature. If the attacker tries to replay old messages, the use of fresh pseudonym and nonce values allows the devices to detect the reuse and discard the messages. Finally, if the attacker tries to intercept the communication between devices and the central ledger (B4), it is protected by the industry standard HTTP over TLS. The server certificate is pinned by the applications. For user authentication, both the inspector and transporter use passwords and API tokens that are stored hashed and salted. We can conclude that the malicious intentions of attacker B are also stopped.

4 Prototype Implementation

We chose Android devices as the mobile device platform as they represent most of the smartphone market [5] and their lower cost and high availability allows for testing with different devices from different manufacturers. Additionally, Android provides the *Google Play services location* API [6], which combines GPS, Wi-Fi and cell network information [7], to retrieve location information. We chose Ubuntu Server 18.04 LTS as the operating system of the server.

The applications were mainly developed in the Java programming language. As support libraries, the Gson library was used to serialize Java objects to JSON objects, for communication with the central ledger, and the OsMoDroid library was used to display maps from OpenStreetMap in both applications. The Central Ledger code was also developed in Java, as it shares some of the code modules with the mobile applications. The Central Ledger interface was specified in OpenAPI format and used the Swagger Editor tool to generate code to use as basis for the implementation of the Central Ledger. The Jersey and FasterXML Jackson frameworks were used to implement the RESTful API of Central Ledger and to serialize received JSON objects to Java objects, respectively. The program was deployed in an Apache Tomcat application server instance.

5 Related Work

Location tracking and proofing systems are relevant to STOP.

[5] https://www.gartner.com/newsroom/id/3876865
[6] https://developer.android.com/training/location
[7] https://developer.android.com/guide/topics/location/battery

5.1 Location Tracking Systems

Location tracking systems are widely used. They primarily use GPS to collect location information regarding a device [1]. This device is usually attached to an object or used by a person. The set of retrieved location points during a time period enables the location tracking of such devices. GPS tracking units are widely used for personal and professional use. Transportation companies often have or sub-contract a *fleet management system* with vehicle location tracking to optimize the costs and use of vehicles. Providers of these systems like InoSat [8] and CarTrack [9] install a GPS tracking unit connected to the vehicle and this device reports location and other relevant data, like vehicle speed and temperature, to servers of the provider. There are also indoor location tracking solutions using other technologies, like *Locix* [10], which uses dedicated and proprietary devices to track assets in a warehouse. These units are positioned together with the object to track and the system enables the location and tracking of inventory by using the 802.11ac Wi-Fi standard [11] and proprietary algorithms.

There are also examples of location tracking being used for regulatory scenarios. *T-Box* [8] is a customized tachograph, a mandatory device in business vehicle in some countries that records vehicle speed, location and driving time, that connects to a remote server to report the collected information. The authors modified the logging mechanism to use the Trusted Platform Module (TPM) of the device to validate the software stack and ensure integrity of stored data. Siegel proposed a system for remote monitoring of vehicles using the standardized automotive port OBD-II [10], using a OBD-II reader with a GSM modem and GPS antenna to report location information to servers. The motivation behind this system is the travel-distance-based taxes, where a vehicle is taxed by distance travelled and not by annual and fuel taxes. This alternative taxing scheme is targeted at electrical vehicles that do not use traditional fuel.

Centralized GPS tracking systems are considered very useful when the used devices are fully trusted. They provide a global scope of the tracked devices in real-time with reasonable implementation costs as GPS technology becomes more accessible. However, GPS technology is vulnerable to attacks [5, 7, 12, 13], especially *spoofing* where the device retrieves wrong location information.

5.2 Location Proofing Systems

Saroiu and Wolman defined *location proof* as a mechanism to allow untrusted mobile devices to prove their location to applications and services [9]. The authors considered that a component of an existent wireless infrastructure such as a Wi-Fi Access Points (AP) and a cellular tower can issue meta-data which mobile devices can use to prove their location. A device can therefore request a

[8] http://www.inosat.pt/
[9] https://www.cartrack.pt/
[10] https://www.locix.com/
[11] https://standards.ieee.org/standard/802_11ac-2013.html

location proof from the infrastructure and this proof can be sent to applications with the intent of proving the location of the mobile device.

Zhu and Cao proposed a location proof system called APPLAUS using Bluetooth enabled mobile devices [14], using five entities: *Prover*, the mobile device who collects proofs from neighbors, *Witnesses*, untrusted mobile devices that generate location proofs, *Location Proof Server*, to store proofs, *Certificate Authority*, to store and validate public keys, and *Verifier*, that verifies submitted proofs. The system uses pseudonyms for each Prover and Witness to prevent device tracking. A Prover broadcasts through Bluetooth a location proof request. If it is received and accepted, a witness creates a proof, signs the proof with its private key and the proof is encrypted with the public key of the location proof server, to guarantee it is only decrypted by the server. This proof is sent to the Prover, who then sends it to the location proof server. The system may ask the Prover to obtain a threshold number of proofs from Witness nodes, becoming more difficult for an attacker to have the number of devices requested to successfully create a false proof. Validation is performed by a Verifier with access to the location proof server.

Canlar et al. [2] created CREPUSCOLO to address both the *neighbor-based* type of proof-based solutions, where nearby mobile devices create proofs, and the *infrastructure-based* type, where location proofs are acquired from trusted infrastructure elements, such as Wi-Fi Access Points. The system uses the same entities of APPLAUS with the addition of the *Token Provider*, a trusted entity placed at a strategic location that generates a proof, called *token*, that may contain an object, such as a picture from a surveillance camera, that proves the device was at that location. Location proofs are exchanged and created like in APPLAUS, with the addition of a nonce in the proof request and in the associated location proof, to prevent replay attacks. The Token Provider is used to mitigate attacks where one device may broadcast messages from another device located at a different site and therefore witnesses may create proofs of the prover located at a different place.

SureThing [3] aims to provide correct location proofs to other applications and services, indoors or outdoors, using as motivation improving the APPLAUS and CREPUSCOLO works. It uses multiple entities similar to the ones in the two previous works presented, *Prover*, *Witness*, *Verifier* and *Certification Authority*, and it also uses geographical coordinates, Wi-Fi fingerprinting and Bluetooth beacons as location proof techniques. The *Witness* can be similar to the Witness entity found in APPLAUS and CREPUSCOLO, called *Mobile Witness*, or to the Token Provider entity in CREPUSCOLO, called *Master Witness*, trusted by the system. The *Verifier* is the central entity of the system who validates and stores all submitted location proofs. Ferreira and Pardal introduced two methods for collusion avoidance. The *Witness Redundancy* mechanism forces the Prover to gather proofs for more than one Witness and chooses the number of witnesses according to the level of service possible. Each proof has a different value associated with it. *Witness Decay* ensures that if a Prover is getting proofs

from the same Witness, they gradually become less valuable and the Verifier will not validate the location if the Prover can not gather proofs with enough value.

The STOP system aims to combine GPS tracking with the location proofing mechanism. The presented location proofing works were used as basis for the STOP inspection protocol to create proofs. The STOP Inspector role is based on the Token Provider and Master Witness entities presented by CREPUSCOLO and SureThing respectively. The Central Ledger combines the Verifier, Location Proof Server and Certification Authority roles of the presented works. STOP reuses location proof concepts and techniques, but applies them to a novel application context, with itineraries, which goes beyond what previous work did.

6 Conclusion

In this paper we presented STOP, a location proofing system for inspection of transportation of goods, together with a detailed design of the security mechanisms. The system is novel in the way it uses mobile devices to track and select vehicles for inspection and assist inspectors with the presentation of required information and the submission of necessary reports. Location proofs are generated at each inspection to prove the vehicle has been inspected. A prototype was implemented and tested in the lab with different Android devices. It is now ready for evaluation in the field.

6.1 Future Work

The current working prototype will be evaluated in the field, to further assess its viability and efficiency. An important aspect is the *quality of the reported location*, as it is used for inspection selection. We will evaluate the location provided by the mobile devices on board of vehicles to estimate its accuracy. It is important to validate the system in actual road conditions, as a wrongful reported location may inaccurately report that the device is in another road. We will also define specific *itineraries*, follow them while using the *Transport* application and calculate the error between the reported location and the correct location coordinates to test the accuracy of the used devices in motion. An additional improvement to the system is to *increase location privacy* by not reporting actual GPS coordinates and replacing them with alternative values that can still be used for inspection selection.

The time interval between the selection of a vehicle for inspection and the time when the transporter device retrieves the notification for inspection is crucial, as the vehicle may not be near the checkpoint at the moment when the on-board device presents the inspection selection to the driver. The response times of the central ledger API calls need to be below a practical threshold, that will be determined and verified experimentally. Additionally, the time interval of the communication between the transporter and inspector devices must be evaluated, as it influences the efficiency of the inspection. The devices will record the timestamps for each Bluetooth interaction, and compute statistics for them in realistic conditions.

Acknowledgements

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (FCT) with reference UID/CEC/50021/2019 (INESC-ID) and through project with reference PTDC/CCI-COM/31440/2017 (SureThing).

References

1. Bajaj, R., Ranaweera, S.L., Agrawal, D.P.: Gps: location-tracking technology. Computer **35**, 92–94 (2002)
2. Canlar, E.S., Conti, M., Crispo, B., Di Pietro, R.: CREPUSCOLO: a Collusion Resistant Privacy Preserving Location Verification System. In: 2013 International Conference on Risks and Security of Internet and Systems (CRiSIS) (2013)
3. Ferreira, J., Pardal, M.L.: Witness-based location proofs for mobile devices. In: 17th IEEE International Symposium on Network Computing and Applications (NCA) (11 2018)
4. Instituto Nacional de Estatística - Portuguese National Statistics Institute: Estatísticas dos Transportes e Comunicações 2016 (2017)
5. Jafarnia-Jahromi, A., Broumandan, A., Nielsen, J., Lachapelle, G.: Gps vulnerability to spoofing threats and a review of antispoofing techniques. International Journal of Navigation and Observation (2012)
6. Krawczyk, H., Paterson, K.G., Wee, H.: On the security of the tls protocol: A systematic analysis. In: CRYPTO 2013: Advances in Cryptology. pp. 429–448 (2013)
7. Onishi, H., Yoshida, K., Kato, T.: Gnss vulnerabilities and vehicle applications. In: 2016 13th Workshop on Positioning, Navigation and Communications (WPNC) (2016)
8. Park, K.W.: T-Box: Tamper-resistant vehicle data collection system for a networked digital tachograph. In: International Conference on ICT for Smart Society (6 2013)
9. Saroiu, S., Wolman, A.: Enabling new mobile applications with location proofs. In: ACM (ed.) Proceedings of the 10th workshop on Mobile Computing Systems and Applications. p. 9 (2009)
10. Siegel, J.E.: Design, Development, and Validation of a Remotely Reconfigurable Vehicle Telemetry System for Consumer and Government Applications. Master's thesis, Massachusetts Institute of Technology (2011)
11. Siuhi, S., Mwakalonge, J.: Opportunities and challenges of smart mobile applications in transportation. Journal of Traffic and Transportation Engineering (English Edition) **3**, 582–592 (2016)
12. Zaabi, K.A.: Android device hacking tricks and countermeasures. In: 2016 IEEE International Conference on Cybercrime and Computer Forensic (ICCCF) (2016)
13. Zhang, Z., Trinkle, M., Qian, L., Li, H.: Quickest detection of gps spoofing attack. In: MILCOM 2012 - 2012 IEEE Military Communications Conference (2012)
14. Zhu, Z., Cao, G.: Applaus: A privacy-preserving location proof updating system for location-based services. In: IEEE Conference on Computer Communications (INFOCOM) 2011 (2011)