J Intell Robot Syst (2016) 84:705–723
DOI 10.1007/s10846-016-0356-x

# The HAMSTER Data Communication Architecture for Unmanned Aerial, Ground and Aquatic Systems

## Aims, Scope and Definitions

**Daniel Fernando Pigatto · Leandro Gonçalves · Guilherme Freire Roberto · Julio Fernando Rodrigues Filho · Natássya Barlate Floro da Silva · Alex Roschildt Pinto · Kalinka Regina Lucas Jaquie Castelo Branco**

Received: 15 December 2014 / Accepted: 14 February 2016 / Published online: 14 March 2016


**Abstract** This paper presents HAMSTER, the HeAlthy, Mobility and Security based data communication archiTEctuRe. HAMSTER is designed for Unmanned Vehicles and addresses mainly three types of communications: machine-to-machine, machine-to-infrastructure and internal machine communications. It is divided into three main versions: Flying HAMSTER (for aerial systems), Running HAMSTER (for terrestrial systems) and Swimming HAMSTER (for aquatic systems). Every version of such architecture is also equipped with Sphere and Nimble. Sphere deals with Safety & Security aspects regarding communication, components "health" and modules authentication. Nimble is aimed at increasing the overall mobility in such scenarios, strongly actuating with inherent communications of each application field. This paper details every aspect of HAMSTER and presents, as a plus at the end, two case studies: the first one consists of an evaluation of five communications schemes for internal communications in airplanes; the second one is a cryptographic evaluation of two Elliptic Curve Cryptography algorithms.

**Keywords** Data communication architecture · Unmanned vehicles · Unmanned systems · UAV · UGV · USV · UUV · UAS · UGS · UWS · Security · Safety · Mobility · HAMSTER · Sphere · Nimble

## 1 Introduction

Recently, the development of Unmanned Vehicles (UV) and Unmanned Systems[1] has increased, which allowed the existence of many different types of vehicles e. g., aerial, terrestrial and aquatic vehicles. Such vehicles should be integrated into the airspace, on public roads and even on aquatic environments following specific laws and requirements of each scenario. Thus, it is essential that communications elements meet healthy, mobility and security requirements, increasing the system overall capabilities and, consequently, allowing the vehicles to be certified and integrated into their operation space, obeying the specific rules determined by authorities.

D. F. Pigatto (✉) · L. Gonçalves · G. F. Roberto · J. F. Rodrigues Filho · N.B.F. Silva · A. R. Pinto · K. R. L. J. C. Branco
Av. Trabalhador São-carlense, 400, São Carlos, SP, Brazil
e-mail: pigatto@icmc.usp.br

[1] Unmanned Systems, in this paper, refers to everything present in a limited environment that allows the execution of a mission, e. g., the Unmanned Vehicle, the Ground Station etc.

This paper presents HAMSTER, the HeAlthy, Mobility and Security based data communication archiTEctuRe. The main objective is to help developers of UV to efficiently implement communications in their systems by considering the internal and external communications. The architecture specification also considers security, safety and mobility constraints.

Apart from the architecture specification, two case studies will be presented at the end of this article. Both are focused on the aerial segment, allowing us to demonstrate how HAMSTER could be helpful for developers. Although the case studies are focused on aerial vehicles, one can apply several different techniques permitting HAMSTER to be used with terrestrial and aquatic vehicles as well. The first case study is related to the fly-by-wireless paradigm, which is a trend for internal communications in aircraft systems and demonstrates how HAMSTER is flexible i. e. not exclusively restricted to wired internal communications. The second case study consists on evaluating the application of an Elliptic Curve Cryptography algorithm for secure communications in unmanned vehicles and systems, highlighting how HAMSTER is aligned with modern cryptographic algorithms as part of its security & safety platform.

This paper is organized as follow: Section 2 presents topics regarding communications in Unmanned Systems and introduces specific ad hoc networks for different application scenarios; Section 3 details HAMSTER data communication architecture, its three versions and its two specialized modules: Sphere, which deals with Security & Safety, and Nimble, that concentrates mobility related advances; Sections 4 and 5 are the case studies; and Section 6 presents the conclusions.

## 2 Communications in Unmanned Systems

Unmanned vehicles are better known as smart cars, unmanned aerial vehicles, drones, water surface and underwater vehicles, among others. In [39], such systems are called Mobile Intelligent Autonomous Systems (MIAS), which are meant to comprise theory and practice of several closely related technologies that have some elements of mobility, intelligence and/or autonomy operating and envisaged not only for robots, but also for other mobile vehicles.

Communication is one of the biggest challenges in designing systems with multiple vehicles and also a crucial aspect for cooperation and collaboration [5, 8]. There are three main types of communications in the context of unmanned vehicles systems: (a) internal machine communications (IMC); (b) machine-to-machine communications (M2M); and (c) machine-to-infrastructure communications (M2I).

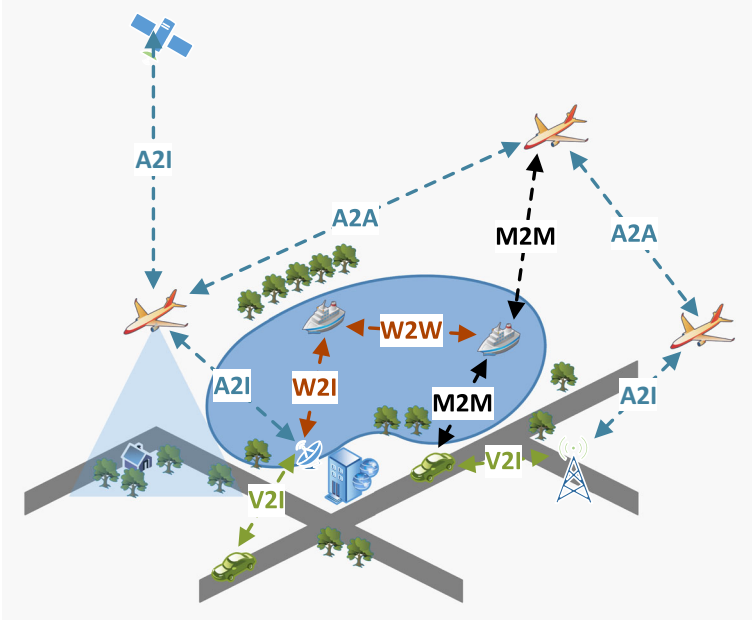



**Fig. 1** Inherent communications to the scenario composed by autonomous aerial, aquatic and terrestrial vehicles, including the interactivity among different vehicles types. Sections 3.1, 3.2 and 3.3 will show details on how HAMSTER deals with the specific differences among the three vehicles types

Figure 1 illustrates a scenario which includes the three main types of vehicles and their respective communications types specifically adapted for their requirements. There are minimal characteristics that differentiate each type of vehicle (aerial, aquatic or terrestrial) regarding communication aspects, e. g., velocity and degrees of freedom of each scenario.

Once all vehicles are directly connected to an infrastructure, such as a control station or a satellite, communication among vehicles can be performed by using such infrastructure. However, infrastructure based communication strongly restricts the capabilities of an UV system. A viable solution to such problem is to apply ad hoc networks connecting all the vehicles. As the number of vehicles on a system increases, the projection of efficient network architectures emerges as a vital issue to be solved.

Besides direct communication among vehicles, in an unmanned system it is possible and very common the existence of a communication link with a central control station, which can be a direct communication or mediated by a satellite [14]. In some approaches, machine-to-machine communications can be accomplished through the infrastructure, which introduces several new problems. First, each vehicle must be equipped with a more expensive and complex hardware to enable communication with the control station or satellite. Furthermore, the reliability of communication is also a disadvantage of such network structure. Due to changing environmental conditions, movements by aerial, aquatic or ground vehicles and different characteristics of relief terrain or obstacles, vehicles may face problems to maintain communication with the infrastructure.

Moreover, one more problem is the restriction of range between the vehicle and the control station. If a vehicle is outside the coverage area of the control station, it is consequently disconnected from the network. An alternative communication solution for unmanned systems with multiple vehicles is the use of ad hoc networks to connect vehicles, also known as UANETs (underwater ad hoc networks, specifically designed for aquatic vehicles), VANETs (vehicular ad hoc networks, specifically designed for terrestrial vehicles) [6, 15, 22, 23, 40, 47] and FANETs (flying ad hoc networks, specifically designed for aerial vehicles) [42, 47]. As long as some vehicles may be out of the coverage range of the control station or satellite, all vehicles compose an ad hoc network, which enables every vehicle to communicate with the control station through communications hops through other vehicles. Although there may exist or not a connection with the base station, vehicles in the air may form a FANET in order to share information and even work in cooperation.

Ad hoc networks are classified according to their utilization, implementation, communication and mission objectives [4]. By definition, FANETs, UANETs and VANETs share several characteristics and can be observed as subsets among themselves. They are considered as special types of MANETs (mobile ad hoc networks) [4], as illustrated in Fig. 2. However, each

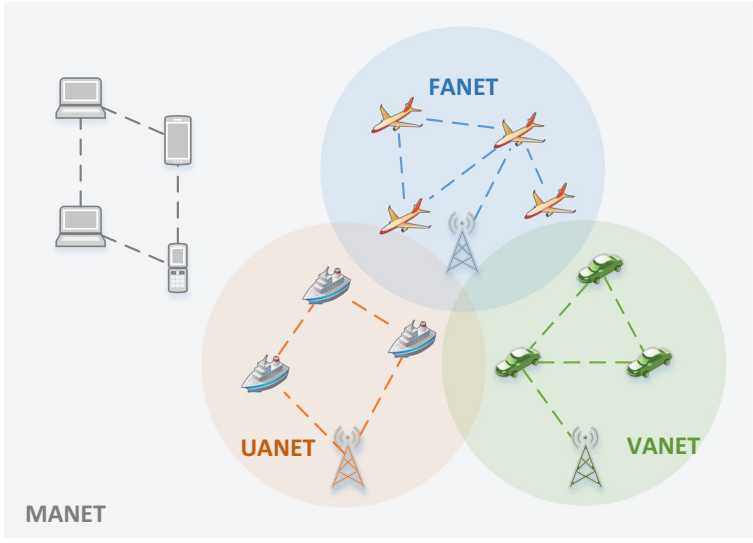



**Fig. 2** Relations among mobile ad hoc networks (MANET) and the derivations: underwater ad hoc networks (UANET), vehicular ad hoc networks (VANET) and flying ad hoc networks (FANET). Figure adapted from [4]

of them face different challenges e. g. the transmission medium in aquatic vehicles could be completely underwater; the obstacles in roads could be extremely unexpected and dangerous; the speeds achieved by aircrafts could be highly challenging for communications; and so on.

The scenario of FANETs is specially more challenging than other ad hoc networks. Some reasons may be pointed out:

- The degree of nodes mobility in FANETs is much greater than in MANETs or VANETs. On the one hand, in MANETs and VANETs nodes may be carried by or embedded on people and cars, respectively. On the other hand, the nodes in FANETs are embedded on an aircraft in flight.
- Due to the high mobility of nodes in FANETs, the topology changes more frequently than the network topology in MANETs, VANETs and UANETs.
- Existing ad hoc networks aim to establish peer-to-peer connections. FANETs also need peer-to-peer connections for the coordination and collaboration of UAVs. Furthermore, most of the time, these networks collect environmental data retransmitted to the control station, similarly to wireless sensor networks [41]. Consequently, FANETs need to support peer-to-peer communication and manage traffic information for monitoring stations.
- Typical distances among nodes in FANETs are much longer than in MANETs and VANETs [9]. Thus, the communication range in FANETs must also be greater than other networks. This fact directly impacts the radio links and hardware elements.
- Multiple UAV systems may include different types of sensors and each sensor may require different strategies for data distribution.

An architecture that explores similarities among different ad hoc networks and vehicles, and also deals with their exclusive characteristics would be helpful for integrating different vehicles and scenarios to increase cooperation. Furthermore, it would also increase the overall security and safety, leading to more chances of getting certification/authorizations to operate. Next section introduces the HAMSTER architecture, which aims to address the aforementioned features.

## 3 HAMSTER: Healthy, Mobility and Security-Based Data Communication Architecture

The Healthy, Mobility and Security-based Data Communication Architecture is divided into three main versions according to the most common types of UV: aerial, aquatic and terrestrial. Two extra modules were also defined: one to deal with security and safety aspects under all three versions of HAMSTER, and another one for all the mobility aspects. Figure 3 presents an overview of HAMSTER hierarchical organization and common abbreviations used in this paper. Every module and architecture version will be discussed in Sections 3.1 to 3.5.

Before focusing on each version of the architecture, the very first step must be the identification of criticality in general modules that compose an UV and a control station, allowing appropriate approaches for each of these elements, ensuring the proper operation of the UV and the entire system. Figure 4 shows the internal organization of a general UV and a general ground station with inherent modules placed according to the need for low latency, medium latency or no real-time constraints. Three communication buses were defined: real-time (red), where latency is expressed in milliseconds, is focused on the essential parts of the UV for its safe operation (e.g. basic sensors); near real-time (yellow), where latency is expressed in tens of milliseconds, comprising modules less critical but still sensitive to real-time (e.g. communication with other vehicles); and no real-time (green) that allows the use of services with non-deterministic time operations, which are not essential to the basic aircraft operations. In the case of the ground station, the organization follows the same idea, despite the fact that it operates with only two buses, since its operations are not as complex as the vehicle.

From such organization (which is a modern version of the idea published in [32]), next subsections will present HAMSTER more accurately.

### 3.1 Flying HAMSTER: the Aerial Systems Architecture

Flying HAMSTER is the version of the architecture which deals exclusively with the aerial segment. It was defined based on specific characteristics and requirements of unmanned aerial vehicles

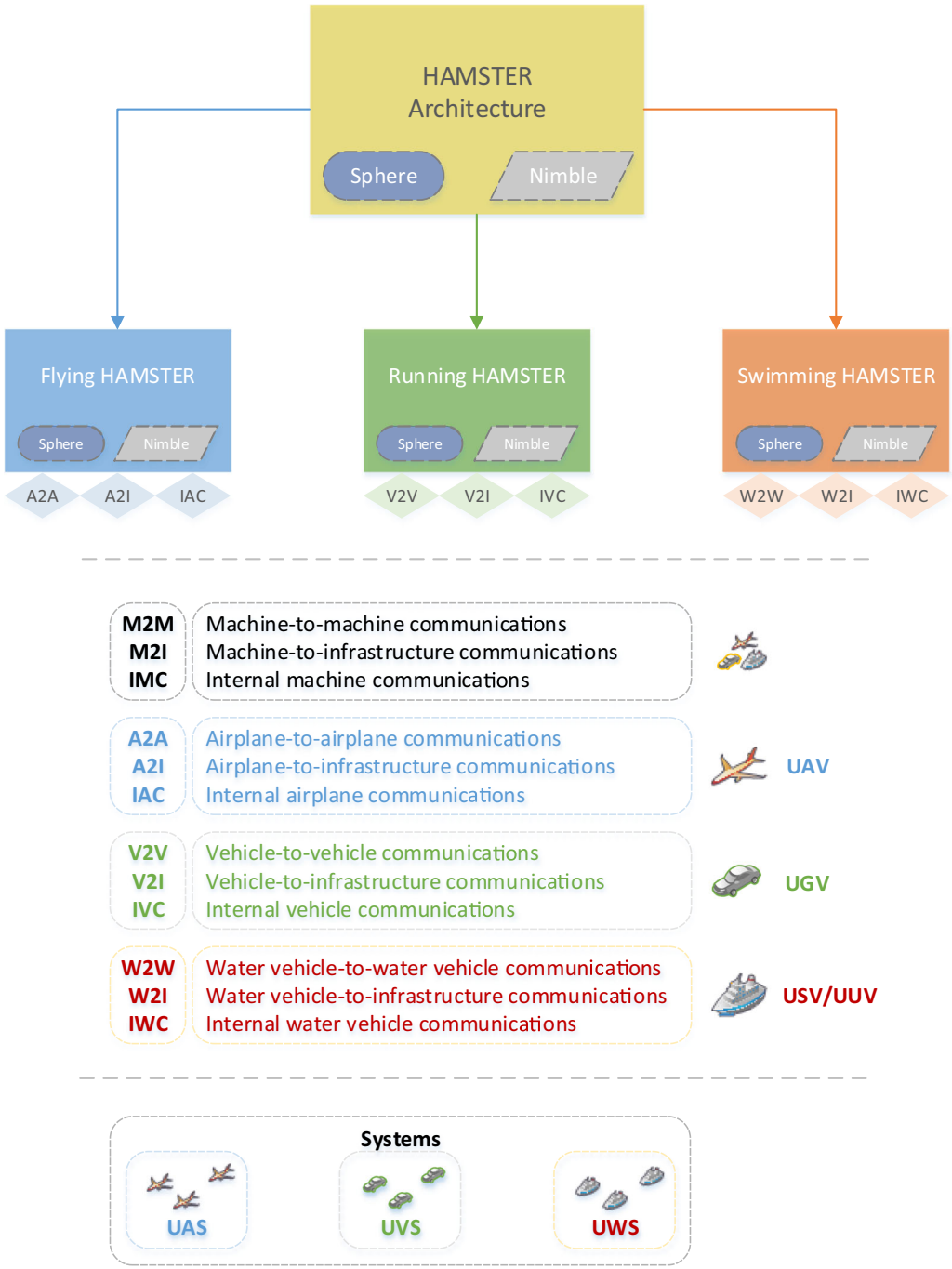



**Fig. 3** HAMSTER versions and the specific modules for mobility (Nimble) and safety & security (Sphere). Flying HAMSTER was designed for aerial systems (UAV and UAS), Running HAMSTER for terrestrial systems (UGV and UGS) and Swimming HAMSTER for aquatic systems (USV, UUV and UWS). Unmanned aircraft systems (UAS) are composed by UAV and every other instance that communicates with the system to perform a mission through A2A, A2I and IAC communications; Unmanned vehicles systems (UVS) are composed by UGV and every other instance that communicates with the system to perform a mission through V2V, V2I and IVC communications; Unmanned water vehicles systems (UWS) are composed by USV/UUV and every other instance that communicates with the system to perform a mission through W2W, W2I and IWC communications. Every established communication is derived from the general concepts represented by M2M, M2I and IMC communications

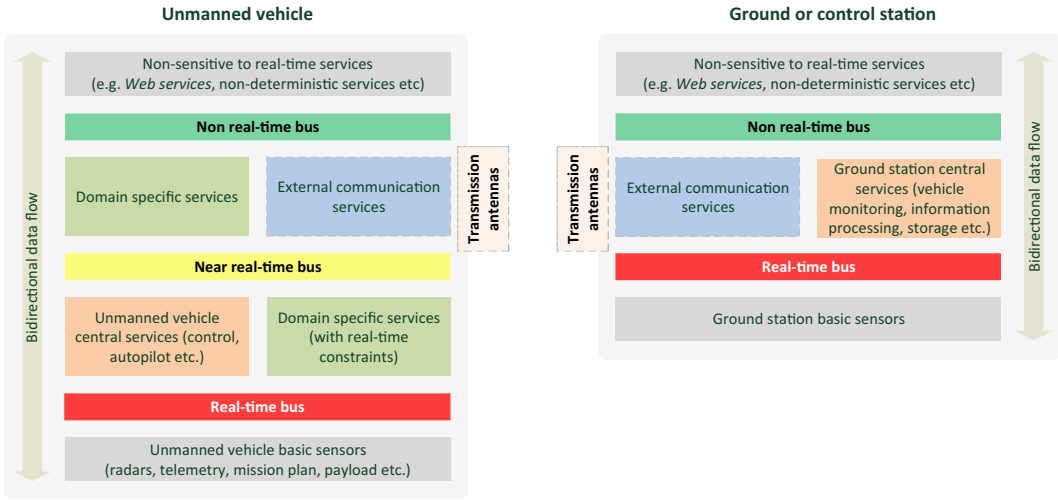



**Fig. 4** Modules of an unmanned vehicle and a ground station with three buses defined according to communication criticality. This figure was adapted from the proposal for unmanned aircraft systems, originally published in [32]

(UAV) and unmanned aircraft systems (UAS). Flying HAMSTER deals specifically with internal airplane communication (IAC), airplane-to-airplane communication (A2A) and airplane-to-infrastructure communication (A2I).

The main applications of UAVs are related to agricultural and environmental monitoring, safety, military and civil defense. The aircraft is usually able to capture images for processing relevant information about a specific field, which may contribute to improve productivity. There are several cases where they might be applied in environmental and borders monitoring, or even applied as aerial sensors in networks for disaster management [36] and multiple UAV applications [5, 22, 28, 29, 48].

FANETs are under A2A communication.

### 3.2 Running HAMSTER: the Terrestrial Systems Architecture

Running HAMSTER deals specifically with vehicles on terrestrial segment. It was defined based on specific characteristics and requirements of unmanned ground vehicles (UGV) and unmanned ground systems (UGS). Running HAMSTER treats internal vehicle communication (IVC), vehicle-to-vehicle communication (V2V) and vehicle-to-infrastructure communication (V2I).

The objective of ground vehicles may vary from driver support in possible dangerous situations with the intention of preventing road accidents, to autonomous driving with no human intervention, which could be used in urban traffic, agriculture, industry and safety applications [13]. The sensor fusion technique is used for integration of multiple sensors such as cameras, digital compasses, and GPS, allowing the vehicle to become autonomous in both urban and rural areas [46].

VANETs are under V2V communication.

### 3.3 Swimming HAMSTER: the Aquatic Systems Architecture

Swimming HAMSTER was designed for vehicles that operate on aquatic environments. It was defined based on specific characteristics and requirements of unmanned surface vehicles (UGV), unmanned undersea vehicles (UUV) and unmanned water vehicles systems (UWS). Swimming HAMSTER is composed by internal water vehicle communication (IWC), water vehicle-to-water vehicle communication (W2W) and water vehicle-to-infrastructure communication (W2I).

The aquatic vehicles have been used for various tasks, especially those related to monitoring of oil exploration and maintenance of hydropower. The current challenges for these vehicles go beyond

autonomy, integrating other areas with the distributed and embedded systems, such as computer networks, artificial intelligence, software engineering, electrical, mechanical and mechatronics engineering, among others. The multiple vehicles tasks are also challenging [1, 50].

UANETs are under W2W communication.

### 3.4 Sphere: Security & Safety Aspects on HAMSTER Architecture

Sphere [34] is the HAMSTER module which will concentrate all the security & safety aspects of the main architecture and all derivative versions. The aim is to define patterns for assuring security & safety that allow every unmanned vehicle derived from HAMSTER to safely share information, even when different scenarios are involved, e. g. to permit the safe communication between an unmanned surface vehicle and an unmanned aerial vehicle. It is also a goal of Sphere to centralize the modules "health" check, which guarantees a safer operation for the vehicle and, consequently, the entire system. This subsection is divided into two parts. The first one addresses a components usage policy and the second one an authentication protocol which will also be responsible by the components "health" checking.

#### *3.4.1 Components Usage Policy*

One of the first steps to ensure the safe operation of a vehicle and to facilitate its integration into the space of actuation (for instance, an UAV into the airspace) must be the redefinition of its components usage policy. Only a few parts of an UV are properly treated to ensure that all connected modules are authentic and have not been replaced or tampered with by a third party. The current policy adopted by most aircraft manufacturers uses a concept of "Accept all" which trusts in all components embedded in an aircraft. This proposal suggests the assumption of an "Almost Deny All" approach, which denies the authenticity of all mechanical components and peripherals attached to the vehicle until the opposite is proved, which may result in safer vehicles.

The categorization of every module is therefore crucial for such a new security model to be applied to UV. There are various peripheral devices embedded in an UV that require different levels of security, which leads to the necessity of a module categorization according to the criticality of their performed functions. The Sphere proposal suggests the modules categorization into primary, secondary, and so on, according to necessity. Bigger and more robust vehicles may have their modules divided into more than two categories, once there is a greater variety of modules criticality to be considered.

Primary modules are those considered essential components for the UV to operate, to be aware of its location and to be able to perform an emergency operation abort safely, even when the mission was not entirely concluded. An autopilot, a GPS receiver, and barometric/inertial units are examples of modules classified as primary, since they might cause a big lost if in failure state. In contrast, modules not considered as essential functions to the UV are classified as secondary modules. Whether abnormal behaviors are detected in any secondary module, the operation of the primary components of the UV is not affected and the secondary module that presented the abnormality should be disabled or isolated. It implies that all primary modules must be authenticated before the operation begins. However, the secondary modules do not necessarily need an authentication before the mission execution.

In addition, to protect the UV against malicious attacks, there is the possibility of identifying anomalies due to usage time. For instance, pressure and collisions suffered by an aircraft may cause natural degradations in components integrity. Therefore, mechanisms to identify the existence of unusual behaviors should help to increase the UV safety, even with a consequent abort of a mission for reasons of physical integrity of the UV. These concepts are strongly connected to Sense & Avoidance area, which are considered as a feature to be integrated to HAMSTER as a future work.

The creation of access profiles for modules is another concept associated with the proposal of authentication. As a mission is assigned to the UV, it must go through an authentication process, which assigns different access permissions to the UV modules. Such concept is similar to the one used in modern operating systems where an administrator user is allowed to install and uninstall software with no restrictions, unlike a visitor user who has access to

the programs, but is not allowed to install/uninstall them. Applied to UV, such concept adds a layer of security that allows blocking the use of selected modules by specific users. Such specification is intended to prevent unauthorized access. Even if there is a single effective user of an UV, no other user (an attacker or not) will have privileged access to modules or their information.

Figure 5 presents the Sphere main modules. Central Security Unit (CSU) is responsible by modules authentication and "health" verification. Based on such results, CSU associates every module to a particular usage profile. The communication security is also addressed by Sphere and directly impacts the respective vehicle communications. The access policy assigned to each user must use cryptographic algorithms suitable for embedded or real-time sensitive environments. Several experiments have been carried out regarding security for critical embedded systems [33, 44].

### *3.4.2 Protocol Structure*

To protect the UV against attacks coming from malicious components, Sphere implements strict security policies. It is necessary to ensure that all modules

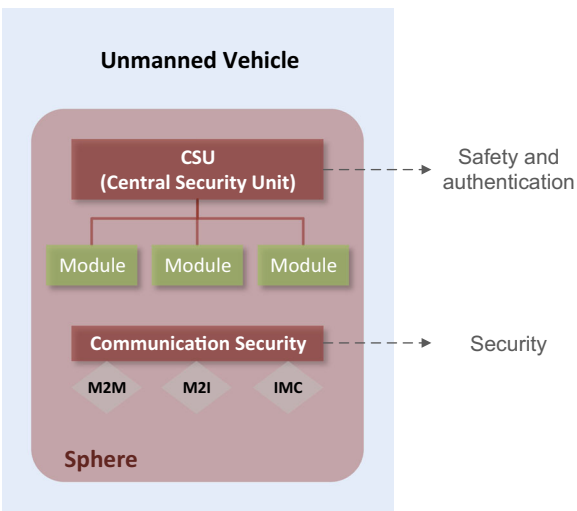



**Fig. 5** Generic composition of Sphere, the Safety & Security aspects module on HAMSTER architecture. Each version of HAMSTER may address several different concepts, according to specific necessities. However, CSU will be able to equally communicate among different HAMSTER versions, allowing heterogeneous vehicles communications

are authentic, so if one of them fails or presents an abnormal behavior, the others must not communicate with it. Furthermore, such policies must be applicable even during vehicle operations, considering that external factors may affect the components behavior e. g. climate or weather changes. In addition, each component must contribute for overall UV safety. In order to apply such methods and requirements, it is assumed that at the system startup or after hardware changes, CSU module remains in an unsafe state and must be authenticated. It will be responsible for storing a table of public keys of all vehicle components, operating similarly to a Certification Authority (CA), which has the goal of ensure that a public key belongs to an entity (module). Each module (or component) will store a hash table of the keys for integrity checking.

During the vehicle start up, a mutual authentication phase should occur with CSU. It checks the database credentials of all modules, their criticality, and even if there is any access restriction. There is also the possibility of deciding whether a module should be initialized or not during the verification stage. The following steps will be authentication and exchange of encrypted messages to establish a secure channel for communication among modules and CSU.

After such handshake, three situations are expected:

- The module that is trying to authenticate and CSU have not been tampered with;
- The module has been tampered with and therefore has not been authenticated:
  - If it is a module of primary type, the UV must not operate;
  - If it is a module of secondary type, communication with it must be interrupted and a notification be sent to a control station.
- The module that is trying to authenticate may notice that CSU is not authentic, and must notify other components about it.

From the point of view of communication security, an ideal situation would be if all modules could authenticate with others. However, this method would cause a system overload, since the increase of modules in the aircraft would cause an exponential increase in the number of exchanged messages. To solve such

problem there exist the e-voting protocols [27]. In case of non-authentic CSU, protocols such as those presented in [21] might be used. Such model can be further expanded according to the needs of the UV, including a negotiation mediated by CSU to create a secure channel of communication among modules. A graphical representation of processes performed during the authentication module with CSU can be seen in Fig. 6.

### 3.5 Nimble: Mobility Aspects on HAMSTER Architecture

Regarding mobility, HAMSTER architecture introduces Nimble. The goal is to allow the creation of a distributed architecture addressing all the elements that compose an Unmanned System (e. g. vehicles, control stations, support vehicles, sensor networks etc.) and to assess every type of communication for each scenario.

The concept of 3-D wireless networks is considered by Nimble. Such networks consider merging the digital world with the physical world allowing the exchange of information among individuals and objects with data services. For instance, in military scenarios modern 3-D wireless networks can be used to connect aircraft, troops and fleets allowing greater exchange of data among them, thus ensuring the security of sensitive information that may be exchanged. This new paradigm introduced by 3-D wireless networks meets the needs of scenarios where UV are applied [26].

Nimble also considers the adoption of IPv6 instead of IPv4, once HAMSTER aims to increase mobility as much as possible. IPv6 naturally increases mobility [31]. It is well known that the increasing number of devices connected to networks nowadays is one of the major reasons that boosted the use of IPv6. In avionics there is an increase of researches that apply the concept of ubiquity, once all the elements involved in a modern Unmanned System often have a specific network address, since every element is connected to the network.

### 3.6 Why HAMSTER Architecture?

Mainly, HAMSTER aims to provide a short way for researchers and developers of Unmanned Vehicles to achieve a flexible and secure solution for implementing communications. It is able to cover both internal and external communications, observing specific environment characteristics and providing robust ways of increasing security, safety and mobility.

Moreover, HAMSTER is aimed at providing an integrated architecture allowing heterogeneous systems to fully communicate. Vehicles should be able to interact with the environment and also other types of vehicles, which would allow more complex and complete missions to be executed. For that, a communication architecture that follows a main structure for communications, mobility and also safety and security will be a better choice. HAMSTER helps achieving such goals once it provides the main modules adapted for each scenario, and still keeps possible the interaction among different HAMSTER versions.

HAMSTER is a data communication architecture that will be available soon. We will present two case studies to demonstrate few of the possibilities

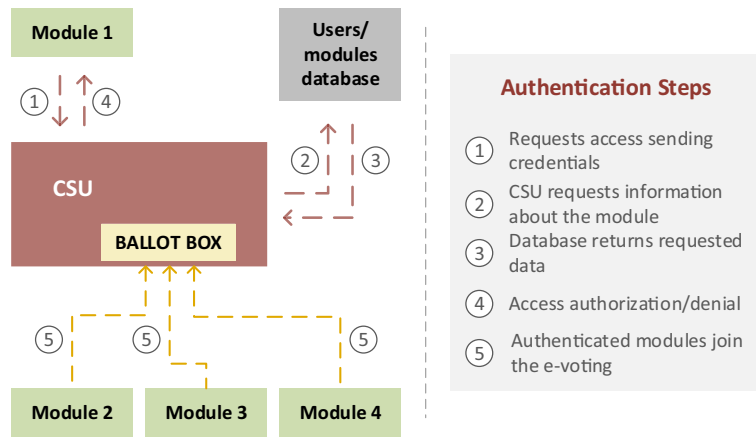



**Fig. 6** Modules and CSU authentication steps. The "Ballot box" is used by every other system modules for authenticating CSU through e-voting protocols

provided by HAMSTER. The case studies (Sections 4 and 5) present results for the aerial segment only. The Unmanned Aircraft Systems were chosen as focus since there are several known challenges still unsolved. Thus, the following sections are focused completely on the aerial segment with Flying HAMSTER (Section 3.1).

## 4 Case Study 1: Fly-by-Wireless with Flying HAMSTER

This study case is focused on Unmanned Aircraft Systems and aims to provide clues for the development of Flying HAMSTER. Thus, it consists on real experiments regarding efforts to the direction of fly-by-wireless on Internal Airplane Communications (IAC as stated in Fig. 3).

### 4.1 Background

There exists nowadays a huge eagerness by the industry towards fly-by-wireless [11, 43], which includes also NASA [45]. Due to that, the aerospace industry and technology providers were motivated to establish: (a) a new emphasis for system engineering approaches to reduce cables and connectors, (b) provisions for modularity and accessibility in the vehicle architecture, (c) a set of technologies that support alternatives to wired connectivity.

Leipold et al. [24] investigated Ultra WideBand technology for in-cabin communication with optimized resource allocation in high performance systems, and average communication latencies have been achieved using simulation. Moreover, Yedavali and Belapurkar [51] present a survey and several recent works on using wireless sensor networks for aircraft control and health monitoring. And in [30], the opportunities and challenges for using wireless inter-connect for safety-critical avionics are discussed. The benefits include weight reduction, increased flexibility and decreased costs and maintenance. However, electromagnetic susceptibility and security still are the main challenges to handle.

The goal of this case study is to present a performance evaluation of five different communication schemes applied to six sensor nodes and a master node embedded on an UAV.

### 4.2 Problem Statement

The paradigm called fly-by-wireless is based on the replacement of wired networks for aircraft control by wireless networks. Thus, it is possible to significantly reduce the weight, fuel consumption and maintenance costs of the aircraft [30]. However, the large number of wireless nodes required for the effective control of the aircraft present major challenges. The protocols used in Wireless Sensor Networks such as IEEE 802.15.4 [16] present a large drop in communication efficiency (ratio between sent and received messages) as the number of nodes increases [35].

Accordingly, the performance of communication protocols suitable for fly-by-wireless technologies must be properly evaluated. Such type of analysis is essential since controlling an aircraft is a critical operation. Therefore, if the control is replaced by wireless nodes, it must have baselines to predict packet delivery efficiency. Thus, one of the main contribution of this case study is to analyze the communication efficiency of wireless nodes (based on IEEE 802.15.4) using classical communication protocols (TDMA, Flurry and Periodic). Our tests were performed on real prototypes.

### 4.3 Materials and Methods

The experiments were defined according to five developed communication schemes, which determine how the sensors communicate with the master module (central module). The sensors were distributed inside and outside of an UAV prototype [7]. The goal of this case study is to find out which communication scheme is more suitable for being applied in a real UAV. The following subsections explain these characteristics of our experiments and the statistical techniques used for evaluating and analysing the results.

#### *4.3.1 Communication Schemes*

For implementing the communication algorithms, three approaches were chosen: TDMA [2], Flurry and Periodic [49].

Although the promising applications enabled by wireless sensor networks are very attractive, there are many system challengesto resolve. First of all,

energy is an essential problem since sensors are usually battery-powered. Second, in some emergency applications, a short time of data collection is also required. Towards such a data gathering sensor networks, TDMA is a good choice to satisfy the above requirements [2]. First, TDMA can save energy by eliminating collisions, avoiding idle listening, or entering inactive states until their allocated time slots. Second, as a collision-free access method, TDMA can bound the delays of packets and guarantee reliable communication.

On the other hand, there are the Periodic protocols. In this case, nodes in a network periodically send messages to the master module. The advantage of using this model is mainly the energy saving, once there is the possibility of deactivating nodes when not communicating [49]. Apart from that, there is also the Flurry protocol which operates in a random way. Since its behaviour cannot be predicted, we opted to evaluate this communication protocol as a parameter for comparisons with the other protocols.

Five communication schemes were implemented based on above mentioned communication approaches. They determine the behaviour of six Arduino boards that were applied to simulate embedded sensors in an UAV. The implemented algorithms are detailed bellow:

1. **TDMA without requests**: the node sends a message each 100 ms; and waits for a resynchronization message each 1200 ms.
2. **TDMA with requests**: the node sends a message each 100 ms only if it receives a request of send type; and waits for a resynchronization message each 1200 ms.
3. **Flurry**: each node has a third of chance of being activated. Once it is active, it sends 12 messages with an interval of 100 ms among them. After that, it waits for 1200 ms.
4. **Periodic without requests**: all nodes send simultaneously a message each 100 ms; and then, for each batch of execution, they wait for a resynchronization message.
5. **Periodic with requests**: all nodes send simultaneously a message each 100 ms; and then, for each batch of execution, they wait for a request message from the master module to start sending messages again.

The schemes number 2 and 5 are dependent on requests sent from the master module. This one is programmed to send a request message each 50 ms. It is also important to point out that the IEEE 802.15.4 protocol was used on the radio modules.

#### *4.3.2 Environmental Setup and Possible Interferences*

The indoor experiments were performed with a prototype of an aircraft. This choice is due to the fact that there is no need to perform such preliminary experiments with an aircraft in flight, since the objective is to evaluate the interference of six modules trying to communicate simultaneously with a central module (the master one).

Therefore, the aircraft chosen was a prototype of Tiriba [7]. The master module ($M$) and four slave modules ($Sn$, where $n$ represents a number identification) were positioned inside the aircraft while the other two slave modules were positioned above the wings (one module per wing). The master module was positioned in the central region of the aircraft, so the distances between master and slave modules could be as small as possible. For a better understanding of modules distribution, see Fig. 7. Figure 8 shows a photo with some modules positioned inside the UAV.

### 4.4 Results and Discussion

Modules were positioned inside the aircraft as it was explained in Environmental setup section (see Fig. 7).

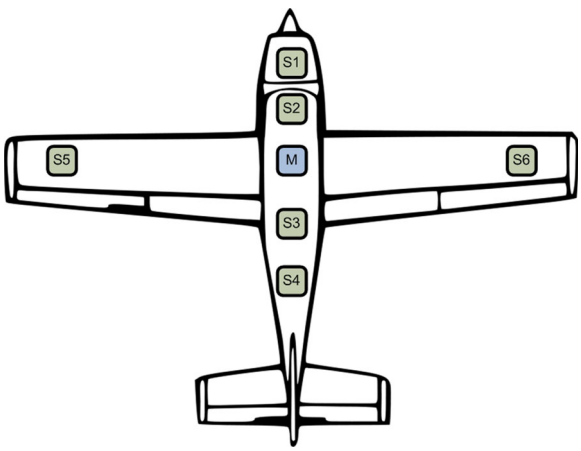



**Fig. 7** The nodes distribution over the airplane: slave modules ($S1$-$S4$) and Master module ($M$) are inside the airplane, while slave modules $S5$ and $S6$ are above the airplane wings

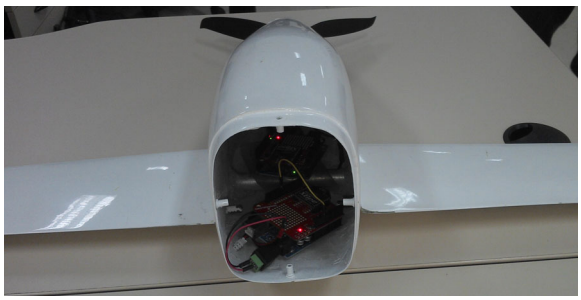


**Fig. 8** Modules positioned inside the aircraft during the setup stage

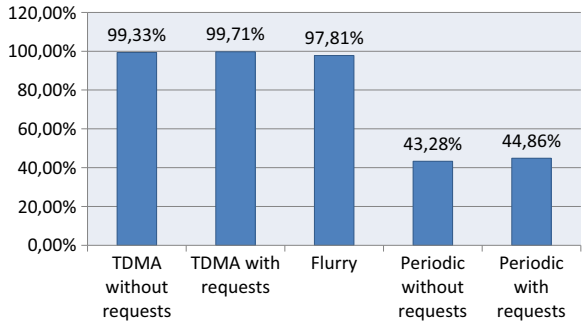



**Fig. 10** The communication efficiency of each communication scheme

The experiments were performed 10 times. Figure 9 shows an inferior performance by TDMA implementations (with and without requests). This is due to the algorithm operation, which is dependent on requests for sending messages. It is also possible to see a random behavior by Flurry communication scheme, which is completely expected, once this implementation forces slave sensors to randomly send messages to the master one. On the other hand, the Periodic communication schemes showed the best overall performance. The Periodic with requests is a bit better than Periodic without requests.

There are some cases where it is possible to visualize that external modules had a bit less success in communication whether compared to internal ones. It was expected due to the fact that external modules have more obstacles to overpass, once they are positioned outside the airplane.

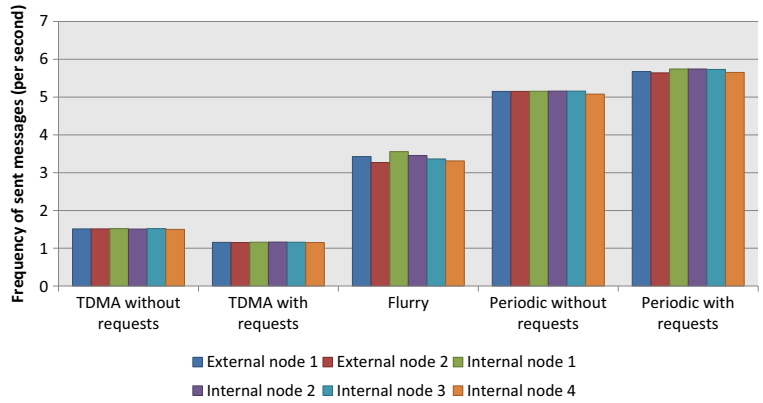
For evaluating the performance in a comparative way, it was calculated the communication efficiency by comparing the number of messages sent by each communication scheme and the number of messages that were in fact received by master module. For each case it was calculated the percentage of delivered messages as shown in Fig. 10.

Figure 10 shows a bad performance by Periodic schemes. It is due to the fact that Periodic schemes send messages with a specific time frequency, however most of them are lost. Comparing Periodic schemes with TDMA schemes, it is important to point out that the second one requests messages, so it is expected that it will be able to receive the prompted message.

On the other hand, Flurry and both TDMA schemes had a great performance: 99,33 % of messages on TDMA without requests have been successfully delivered; 99,71 % of messages on TDMA with requests have been successfully delivered; and on Flurry scheme, 97,81 % of messages have been successfully delivered.

The results of Case study 1 are helpful for the definition of HAMSTER architecture regarding fly-by-wireless, which has been a field of intense research lately. The internal aspects of Flying HAMSTER, the HAMSTER version for avionics, will be more detailed

**Fig. 9** Frequency of messages sent by each module in each communication scheme per second



within the official documentation of the architecture. Although this case study was carried out specifically for UAS, it could also provide clues for internal wireless communications in cars and aquatic vehicles with further adaptations.

## 5 Case Study 2: Evaluation of Elliptic Curve Cryptography for Providing Security

This study case is focused on Unmanned Aircraft Systems and aims to provide clues for the development of Flying HAMSTER. Thus, it consists on real experiments regarding efforts to the direction of security and safety on all the communication types defined for the aerial segment (A2A, A2I and IAC as stated in Fig. 3).

### 5.1 Background

Eissa et al. [10] focused on creating an authentication mechanism in MANETs. For this, four different keys are generated: an identity key, a public key, a private key, and a symmetric key. The first step is to check the level of confidence of every node. For this, neighboring nodes are consulted using the identification key. If at least $n$ nodes confirm the confidence level of the others, the communication starts. Both nodes agree on a session key using the public key and hold in their databases of reliable keys. Thereafter, such nodes use their private key to encrypt messages in communication. As a result, cryptanalysis attacks do not work as it is necessary to have a public key.

Faughnan et al. [12] and Kashikar and Nimbhorkar [20] have shown the possible attacks a UAV might face during operations. Javaid et al. [18] aimed the creation of secure channels for communication among UAV systems, satellites and base stations. After a teste with attacks, the system had some failed components, especially after the denial of service attack.

Raj et al. [37] studied a protocol for nodes admission to a network in a decentralized manner. At the start it was assumed that the network is composed by trusted nodes only. This group of nodes owns a shared secret key. To join the network, a node must request and receive permission from all nodes in the network using a secure communication channel. If a node is approved, to establish a communication, the other nodes create a new shared secret key that will be used for communications between pairs of nodes.

The traditional cryptographic algorithms are usually enough for most of the computing applications. The asymmetric RSA, for instance, is well tested and sometimes considered a synonym of public-key cryptography. However, critical embedded systems with strict limitations would work at better conditions if smaller cryptographic keys were used, still providing high reliability to the result. Our investigation with Case study 2 is the chance of using a different public-key algorithm to replace RSA.

### 5.2 Problem Statement

This case study was carried out to identify the viability of using ECC instead of RSA, since the first one might present several advantages for computational systems with restricted resources. Critical embedded systems are usually implemented with limited resources, which are shared among all the modules that compose the whole system. Thus, a security proposal that considers such limitations would be feasible to be implemented in embedded systems. Next sections present the ECC developed algorithms, results and discussions.

### 5.3 Materials and Methods

#### *5.3.1 Elliptic Curve Cryptography (ECC)*

The three accepted encryption schemes are based on three mathematical problems [19]: Integer factorization problem (IFP), Discrete logarithm problem (DLP), and Elliptic curve discrete logarithm problem (ECDLP). The latter offers a higher level of security, since it operates with smaller key sizes in comparison to RSA and DSA (Digital Signature Algorithm). To achieve an appropriate level of security, RSA and DSA should use 1024-bit key size based on the time needed to break these figures, while the ECC needs to operate with 160-bit keys.

Besides the much smaller key size, ECC algorithm has specific advantages, such as only exponential-time attacks may be applied if the curve is carefully chosen. Even if factoring and multiplicative group discrete logarithms are broken, the ECDL can still be difficult to compute [19]. Establishing a comparison, the security level of an implementation of elliptic curves with 160-bit key is equivalent to RSA 1024-bit key size [25].

*5.3.2 Developed Algorithms*

Two algorithms were developed in this case study [33]. The method chosen for the implementation of ECC is El-Gamal, which combines the properties of the El-Gamal elliptic curve encryption method of exchanging messages. Its operation is given as follows: two users must share the same elliptic curve and a point $P$. Each one must choose a random number that acts as its private key, and multiply the known point, obtaining $aP$, which becomes its public key. At the beginning of the communication, the public key is transmitted to the other user, which has $bP$ as public key and thus the private key $b$. For the exchange of messages the user needs to multiply his/her own private key by the public key of the other user, obtaining $b(aP)$, and then add this result to the message encoded in an $M$ number. Therefore, the message will be $M + b\,(aP)$. When the message is delivered to the receiver, he/she will be able to decode it by multiplying his/her private key by the public key of the other user, $(a(bP))$, and subtracting the total content, $M + b\,(AP) - (bP) = M$.

In the implementation two libraries were used as tools to perform mathematical operations: MIRACL (Multiprecision Integer and Rational Arithmetic C/C++ Library) due to its application in other papers [38], and RELIC 0.2.3 (Library for Efficient Cryptography) [3]. The MIRACL library produced by Shamus Software is proprietary, but free for educational use. It is intended to behave as a tool for developers of encryption systems and offers the necessary operations to handle large numbers and a full support for elliptic curves. RELIC, on the other hand, has been developed by researchers at UNICAMP in order to provide cryptographic tools based on flexibility and efficiency. It has implementations of arithmetic of large integers, arithmetic in binary and prime fields, elliptic curves over prime fields, among others.

The algorithm developed with the MIRACL library operates on blocks of fixed size of 18 characters and the algorithm that uses the RELIC library operates on blocks of 40 characters. Parameters that define the elliptic curve and the point used in common by the users are fixed. These definitions have been established according to some experiments that have proven their efficiency while operating with these block sizes. They have been performed by the authors as a simple comparison between possible block sizes.

The algorithms were developed in C language and run through three actions that must be informed as parameters: key creation, encryption and decryption. The latter two also require the names of input and output files. The implementations of the two algorithms have similar structures with four main functions responsible for creating keys, encrypting, decrypting and calling other functions. In addition, both algorithms have defined structures for the public and private keys.

The function responsible for creating the keys first creates a random integer as the private key and then multiplies a known point of the curve by that number to generate the public key. After being generated, keys are stored in two different files to be exchanged over the network, if necessary.

The function that encrypts a block of text starts transforming the message in a point on the curve. This is the main difference between the developed algorithms in this research. The MIRACL library maps a number of bytes, but it does not work when there is a NULL byte. It is therefore necessary to treat the block previously and indicate the places where this byte is present. After all of these actions, it is necessary to map the sequence of bytes in a number. From this number it is possible to find the point that is part of the curve where the x-axis is closer to that number.

With RELIC library, it is necessary to map the sequence of bytes, with no previous treatment for numbers. Through transactions between the number and the structure of a known point of the curve, the sequence of bytes is transformed into a point.

Using the messages mapped at points, the encryption is performed by multiplying the private key by the receiver's public key and add to the point of the message. In contrast, the function that decrypts the encrypted block subtracts the multiplication of the private key by the sender's public key, obtaining the message encoded at a point. After the reverse operation, it is possible to obtain the original sequence of bytes.

The function that calls other functions reads the parameters of the execution and operations are defined according to the action. When the action is to generate keys, an appropriate function is executed. However, for encryption and decryption, input and output operations must be performed. In encryption, it is necessary to read the bytes from the original file and write the

encrypted blocks at points in an output file. In decryption, the program reads the points of the encrypted file and writes the byte sequences in the output file, restoring the original file.

In the MIRACL library scenario, functions to read and write points are already implemented, but in the case of RELIC it was necessary to create them. The structure of a point was studied to determine the basic data type of each component of the structure and encode functions of writing and reading for each structure.

### 5.4 Results and Discussion

The experiments were set up according to rules to evaluate the performance of computing systems [17]. A variety of terms, such as response variable, factors, and interaction levels is used during the stages of design and analysis of experiments. Response variable represents the result (output) of an experiment and is often the variable selected to measure the systems performance. Factors are the variables that affect the system response; levels are the values that a factor may assume; and interaction indicates the dependency between the factors evaluated [17].

The first steps were the definitions of a response variable to be evaluated, the amount of replication required for the experiments and the testing environment. The environment used to run the tests is a Pentium Dual-Core CPU T4300 2.10 GHz with 2 GB of RAM and Linux Ubuntu 11.04 operating system. To evaluate the efficiency of encryption and compare the results of the developed algorithms, the response variable selected was the average response time. The whole process performed in the experiments is the encryption and decryption of each input file, that is, the average response time refers to the sum of the average response times of each of these two steps. Each experiment was performed 15 times, ensuring a statistical validation since there was no large standard deviation among the results.

There are a few ways to accomplish the design of experiments. In this work we have used the full factorial design (Jain, 1991). In this type of planning, all combinations are used considering all factors and levels. Thus, it is possible to evaluate all factors, determine the effect of each factor on the experiments and verify the interactions between them. Table 1 shows the possible combinations of the experiments. The first factor is the library used, which has two levels: MIRACL and RELIC. The second factor is the size of the message, which may vary between 50 and 100 KB. Finally, the third factor assumed is the key size, also with two variations: 160-bit and 256-bit. Therefore, it is possible to generate eight different combinations for the experiments.

**Table 1** Combinations of experiments

| Exp. | Library | Key size (bits) | Message size (KB) |
|---|---|---|---|
| 1 | MIRACL | 160 | 50 |
| 2 | MIRACL | 160 | 100 |
| 3 | MIRACL | 256 | 50 |
| 4 | MIRACL | 256 | 100 |
| 5 | RELIC | 160 | 50 |
| 6 | RELIC | 160 | 100 |
| 7 | RELIC | 256 | 50 |
| 8 | RELIC | 256 | 100 |

Figure 11 shows the comparison between the average response times achieved by each of the algorithms run in the first message size (smaller), considering the two key sizes. The algorithm based on MIRACL library has a considerably higher time than the one based on RELIC, in both cases. The times obtained are approximately 9 seconds (MIRACL) and 3.3 seconds (RELIC), in which the key size is 160-bit. When using 256-bit key size, the times are 20.9 seconds (MIRACL) and 10.6 seconds (RELIC).

Figure 12 shows the second comparison chart with the performance of algorithms to encrypt and decrypt the second message size (larger). There was a natural elevation in the response time due to increased data to

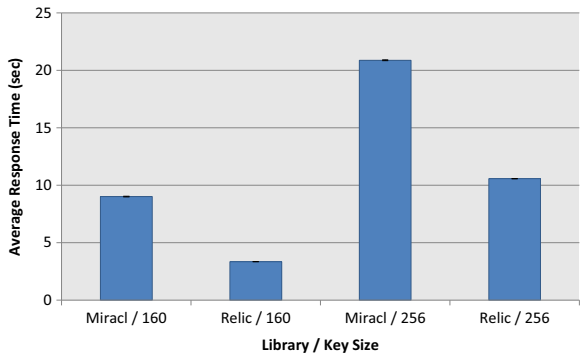



**Fig. 11** Comparison between MIRACL and RELIC libraries with the first message size (50 KB)

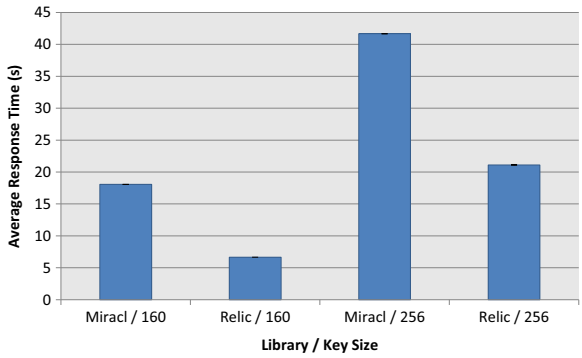



**Fig. 12** Comparison between MIRACL and RELIC libraries with the second message size (100 KB)

be processed while maintaining the same characteristics of the previous comparison. The times obtained are approximately 18.1 seconds (MIRACL) and 6.6 seconds (RELIC), in which the key size is 160-bit. When using 256-bit key size, the times are 41.7 seconds (MIRACL) and 21.1 seconds (RELIC).

The charts show a better performance of the algorithm based on RELIC in both cases (Figs. 11 and 12). We have calculated the percentage of influence of each factor of the performance evaluation, as well as the influence of the associated factors over the response time. The chart in Fig. 13 shows that factor A (algorithm) exerted a 28 % influence on the results, which is relevant to the comparison presented. Factor B (key length) exerted a greater influence on the response variable, with a total of 40 %. Factor C (message size) influenced the results in 23 %. The associated factors exerted small influences: BC influenced 4 %, AC only 3 %, AB only 2 %, and ABC associated exerted no influence.

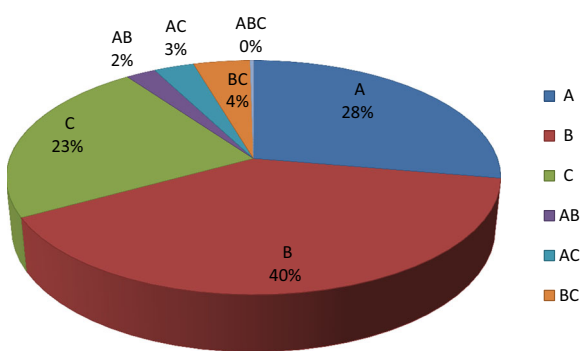



**Fig. 13** Influences of each factor on the response time (A - Algorithm; B - Key Size, C - Size of message)

These results have shown that the message size influences the outcome of the application due to the difference in the amount of data to be encrypted. However, the aim of this article was to show that the influence of the algorithm used is quite considerable. It is also important to point out that the adopted key size of 160-bit was recommended by NIST [25]. As the response time is crucial for critical embedded systems that often work with real-time tasks, RELIC is more suitable for the implementation of the ECC algorithm, considering the conditions of the environment used for the experiments. It was also possible to identify several variations when the experiments were conducted in an environment with similar characteristics to a critical embedded system.

It is important to notice that, initially, the time obtained may be classified as long. However, considering that the assumed key size is relatively large and critical embedded systems require the application of cryptography in most cases to send short commands, such as changing routes or missions, the performance presented has met the expectations, obtaining very short response times, which may be considered a solution for real-time systems, the focus of this work. It is also important to point out that these algorithms should be applied in association with symmetric key algorithms to significantly reduce the response times.

## 6 Conclusions

This paper presented HAMSTER, the HeAlthy, Mobility and Security based data communication archiTEctuRe. It was designed for Unmanned Vehicles and addresses mainly three types of communications (machine-to-machine, machine-to-infrastructure and internal machine communications). The three main versions of HAMSTER were presented in this paper: Flying HAMSTER (for aerial systems), Running HAMSTER (for terrestrial systems) and Swimming HAMSTER (for aquatic systems). Every version of HAMSTER architecture is also equipped with Sphere and Nimble, two special modules that deal with special situations. Sphere covers Safety & Security aspects regarding communication, components "health" and modules authentication. Nimble is aimed at increasing the overall mobility in such scenarios, strongly actuating with inherent communications of each application field.

It was also presented an evaluation of five communications schemes for internal communications in airplanes and a cryptographic evaluation of two Elliptic Curve Cryptography algorithms. Results should help Researchers & Developers to choose the best option for implementing IMC communications and Security into their Unmanned Systems.

**Acknowledgments** The authors would like to thank FAPESP for the financial support through process number 2012/16171-6.

## References

1. Abbott-Mccune, S., Kobezak, P., Tront, J., Marchany, R., Wicks, A.: UGV: Security analysis of subsystem control network. In: Proceedings of SPIE - The International Society for Optical Engineering, vol. 8741. Baltimore (2013)
2. Alba, E., Talbi, E.g., Zomaya, A.Y., Mao, J., Wu, Z., Wu, X.: A TDMA scheduling scheme for many-to-one communications in wireless sensor networks. Comput. Commun. **30**(4), 863–872 (2007)
3. Aranha, D.F., Gouvêa, C.P.L.: RELIC is an Efficient LIbrary for Cryptography (2011)
4. Bekmezci, I., Sahingoz, O.K., Temel, S.: Survey Flying Ad-Hoc Networks (FANETs): A survey. Ad Hoc Netw. **11**(3), 1254–1270 (2013)
5. Bouachir, O., Abrassart, A., Garcia, F., Larrieu, N.: A mobility model for UAV ad hoc network. In: 2014 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 383–388. IEEE (2014)
6. Bovee, B., Nekoui, M., Pishro-Nik, H., Tessier, R.: Evaluation of the universal geocast scheme for VANETs. In: Vehicular Technology Conference (VTC Fall), 2011 IEEE, pp. 1–5 (2011)
7. Branco, K.R.L.J.C., Pelizzoni, J.M., Neris, L.O., Trindade, O., Osorio, F.S., Wolf, D.F.: Tiriba - a new approach of UAV based on model driven development and multiprocessors. In: 2011 IEEE International Conference on Robotics and Automation, pp. 1–4. IEEE (2011)
8. Chung, H., Oh, S., Shim, D.H., Sastry, S.S.: Toward robotic sensor webs: Algorithms, systems, and experiments. Proc. IEEE **99**(9), 1562–1586 (2011)
9. Clapper, J., Young, J., Cartwright, J., Grimes, J.: Unmanned systems roadmap 2007–2032. Tech. rep., Department of Defense (2007)
10. Eissa, T., Razal, S.A., Ngadi, M.A.: Enhancing MANET security using secret public keys. In: 2009 International Conference on Future Networks, pp. 130–134 (2009)
11. Elgezabal Gomez, O.: Fly-by-wireless: Benefits, risks and technical challenges. In: CANEUS Fly by Wireless Workshop 2010, pp. 14–15. IEEE (2010)
12. Faughnan, M.S., Hourican, B.J., MacDonald, G.C., Srivastava, M., Wright, J.A., Haimes, Y.Y., Andrijcic, E., Guo, Z., White, J.C.: Risk analysis of unmanned aerial vehicle hijacking and methods of its detection. In: Systems and Information Engineering Design Symposium (SIEDS), 2013 IEEE, pp. 145–150 (2013)
13. Fernandes, L.C., Souza, J.R., Pessin, G., Shinzato, P.Y., Sales, D., Mendes, C., Prado, M., Klaser, R., Magalhães, A.C., Hata, A., Pigatto, D., Castelo Branco, K., Grassi, V., Osorio, F.S., Wolf, D.F.: CaRINA intelligent robotic car: Architectural design and applications. J. Syst. Archit. **60**(4), 372–392 (2014)
14. Frew, E., Brown, T.: Networking issues for small unmanned aircraft systems. J. Intell. Robot. Syst. Theory Appl. **54**(1-3 SPEC. ISS.), 21–37 (2009)
15. Hwang, J.N.: Wireless MediaNets: Application-driven next-generation wireless IP networks. Multimed. Syst. **17**(4), 251–285 (2010)
16. IEEE Standards Association: IEEE 802.15.4-2011 - IEEE standard for local and metropolitan area networks - Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) (2011)
17. Jain, R.: The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling. Wiley Professional Computing, Wiley (1991)
18. Javaid, A.Y., Sun, W., Devabhaktuni, V.K., Alam, M.: Cyber security threat analysis and modeling of an unmanned aerial vehicle system. In: 2012 IEEE Conference on Technologies for Homeland Security (HST), pp. 585–590 (2012)
19. Jena, D., Jena, S.K.: A novel and efficient cryptosystem for large message encryption. Int. J. Inf. Commun. Techol. **3**(1), 32–39 (2011)
20. Kashikar, M., Nimbhorkar, S.: Designing acknowledgement based MANET using public key cryptography. In: 2013 8th International Conference on Computer Science Education (ICCSE), pp. 228–233 (2013)
21. Kikuchi, H., Nakazato, J.: Modint: A compact modular arithmetic java class library for cellular phones, and its application to secure electronic voting. Secur. Protect. Inf. Process. Syst., 177–192 (2004)
22. Kim, S.W., Seo, S.W.: Cooperative unmanned autonomous vehicle control for spatially secure group communications. IEEE J. Selected Areas Commun. **30**(5), 870–882 (2012)
23. Kuiper, E., Nadjm-Tehrani, S.: Mobility models for UAV group reconnaissance applications. In: ICWMC '06. International Conference on Wireless and Mobile Communications, 2006, p. 33 (2006)
24. Leipold, F., Tassetto, D., Bovelli, S.: Wireless in-Cabin Communication for Aircraft Infrastructure. Telecommunication Systems (2011)
25. Lenstra, A.K., Verheul, E.R.: Selecting cryptographic key sizes. J. Crypt. **14**, 255–293 (1999)
26. Li, P., Pan, M., Fang, Y.: Capacity bounds of three-dimensional wireless ad hoc networks. IEEE/ACM Trans. Netw. **20**(4), 1304–1315 (2012)
27. Liaw, H.T.: A secure electronic voting protocol for general elections. Comput. Secur. **23**(2), 107–119 (2004)
28. Luo, C., Ward, P., Cameron, S., Parr, G., McClean, S.: Communication provision for a team of remotely searching UAVs: A mobile relay approach. In: Globecom Workshops (GC Wkshps), 2012 IEEE, pp. 1544–1549 (2012)

29. Maza, I., Caballero, F., Capitán, J., Martínez-De-Dios, J., Ollero, A.b.: Experimental results in multi-UAV coordination for disaster management and civil security applications. J. Intell. Robot. Syst. Theory Appl. **61**(1–4), 563–585 (2011)
30. Mifdaoui, A., Gayraud, T.: Fly-By-Wireless for next generation aircraft: Challenges and potential solutions. In: 2012 IFIP Wireless Days, pp. 1–8. IEEE (2012)
31. NIC.br: IPv6.br — Portal sobre IPv6 do NIC.br (2013)
32. Pigatto, D.F., Goncalves, L., Pinto, A.S.R., Roberto, G.F., Fernando Rodrigues Filho, J., Branco, K.R.L.J.C.: HAMSTER - Healthy, mobility and security-based data communication architecture for unmanned aircraft systems. In: 2014 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 52–63. IEEE (2014)
33. Pigatto, D.F., Silva, N.B.F.D., Branco, K.R.L.J.C.: Performance evaluation and comparison of algorithms for elliptic curve cryptography with El-Gamal based on MIRACL and RELIC libraries. J. Appl. Comput. Res. **1**(2), 95–103 (2012)
34. Pigatto, D.F., Smith, J., Lucas, K.R., Branco, J.C.: Sphere: A novel platform for increasing safety & security on unmanned systems. In: 2015 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 1059–1066. IEEE (2015)
35. Pinto, A., Montez, C., Araújo, G., Vasques, F., Portugal, P.: An approach to implement data fusion techniques in wireless sensor networks using genetic machine learning algorithms. Inf. Fus. **15**, 90–101 (2014)
36. Quaritsch, M., Kruggl, K., Wischounig-Strucl, D., Bhattacharya, S., Shah, M., Rinner, B.: Networked UAVs as aerial sensor network for disaster management applications. e & i Elektrotechnik und Informationstechnik **127**(3), 56–63 (2010)
37. Raj, E., SelvaKumar, S., Lekha, J.R.: Node admission protocols for secure communications. In: 2011 International Conference on Emerging Trends in Electrical and Computer Technology (ICETECT), pp. 69–73 (2011)
38. Ramachandran, A., Zhou, Z., Huang, D.: Computing cryptographic algorithms in portable and embedded devices. In: IEEE International Conference on Portable Information Devices, 2007. PORTABLE07, pp. 1–7 (2007)
39. Raol, J.R., Gopal, A.K.: Mobile Intelligent Autonomous Systems. Taylor & Francis (2012)
40. Rawat, P., Singh, K.D., Chaouchi, H., Bonnin, J.M.: Wireless sensor networks: A survey on recent developments and potential synergies. J. Supercomput. (2013)
41. Rieke, M., Foerster, T., Broering, A.: Unmanned aerial vehicles as mobile multi-sensor platforms. In: The 14th AGILE International Conference on Geographic Information Science. Utrecht (2011)
42. Sahingoz, O.K.: Networking models in flying ad-hoc networks (FANETs): Concepts and challenges. J. Intell. Robot. Syst. **74**(1–2), 513–527 (2013)
43. Sampigethaya, K., Poovendran, R.: Aviation cyberphysical systems: Foundations for future aircraft and air transport. Proc. IEEE **101**(8), 1834–1855 (2013)
44. Schoaba, V., Sikansi, F.E.G., Pigatto, D.F., Branco, K.R.L.J.C., Branco, L.C.: Digital signature for mobile devices: A new implementation and evaluation. Int. J. Futur. Gener. Commun. Netw. **4**, 23–36 (2011)
45. Studor, G.: "Fly-by-Wireless": A revolution in aerospace vehicle architecture for instrumentation and control (2007)
46. Sun, Z., Wang, P., Vuran, M., Al-Rodhaan, M., Al-Dhelaan, A., Akyildiz, I.b.: BorderSense: Border patrol through advanced wireless sensor networks. Ad Hoc Netwo. **9**(3), 468–477 (2011)
47. Temel, S., Bekmezci, I.: On the performance of flying ad hoc networks (FANETs) utilizing near space high altitude platforms (HAPs). In: 2013 6th International Conference on Recent Advances in Space Technologies (RAST), pp. 461–465 (2013)
48. Verma, A., Fernandes, R.: Persistent unmanned airborne network support for cooperative sensors. In: Proceedings of SPIE - The International Society for Optical Engineering, vol. 8756. Baltimore (2013)
49. Wei, G., Ling, Y., Guo, B., Xiao, B., Vasilakos, A.V.: Prediction-based data aggregation in wireless sensor networks: Combining grey model and Kalman Filter. Comput. Commun. **34**(6), 793–802 (2011)
50. Xiang, X., Liu, C., Lapierre, L., Jouvencel, B.: Synchronized path following control of multiple homogenous underactuated AUVs. J. Syst. Sci. Complex. **25**(1), 71–89 (2012)
51. Yedavalli, R.K., Belapurkar, R.K.: Application of wireless sensor networks to aircraft control and health management systems. J. Control Theory Appl. **9**(1), 28–33 (2011)

**Daniel Fernando Pigatto** is a PhD student in the Institute of Mathematics and Computer Sciences at the University of Sao Paulo. He obtained a M.Sc. in Computer Science in 2012 from the University of Sao Paulo. At ICMC/USP, he has been working in network security, embedded software development and critical embedded applications. His current research interests are Unmanned Aerial Vehicles, Network Security and Mobile Robotics.

**Leandro Gonçalves** received his bachelor's degree in Computer Science from São Paulo State University (2015), and currently is a Master's Student at São Paulo State University in the field of Computer and Network Security. He has experience with embedded systems, wireless sensor networks, computer networks and computer and network security.

**Guilherme Freire Roberto** received his bachelor's degree in Computer Science with emphasis in automated systems and digital control from São Paulo State University (2015), and currently is a Master's Student at São Paulo State University in the field of Mathematics and Computer Intelligence. He has experience in wireless sensor networks, embedded systems and image processing.

**Julio Fernando Rodrigues Filho** is graduated in Computer Engineering from the University of São Paulo (2014).

**Natássya Barlate Floro da Silva** received the Engineering degree in Computer Engineering from University of Sao Paulo (USP), in 2012. Currently, she is a PhD student in Computer Sciences in the Institute of Mathematics and Computer Science (ICMC-USP). Her main areas of research interest are Unmanned Aerial Vehicles (UAVs), Control Systems and Sensor Data Fusion.

**Alex Roschildt Pinto** holds Master's degree in Science from Federal University of Santa Catarina (UFSC) - (2003) and PhD in Electrical Engineering at UFSC (2010). Currently, he is Assistant Professor of UFSC in the Department of Engineering. He has experience in the following areas: embedded systems, wireless sensor networks and Internet of Things. He has published more than 50 international papers (Journal, Conferences and Book Chapters) and also has achieved three IEEE awards.

**Kalinka Regina Lucas Jaquie Castelo Branco** has Master in Computer Science from University of Sao Paulo (1999) and Ph.D. in Computer Science from University of Sao Paulo (2004). She is currently an Associate Professor of the Institute of Mathematics and Computer Science (ICMC-USP), working in the department of Computer Systems. She has experience in Computer Science, with emphasis on Computer Networks, Security, Embedded Systems, and Distributed Computing.