Cybersecurity in Autonomous Systems: Evaluating the performance of hardened ROS

David Mañanes, Francisco Javier Rodríguez Lera, Jesús Balsa and Vicente Matellán,

***Abstract*—As robotic systems spread, cybersecurity emerges as major concern. Currently most research autonomous systems are built using the ROS framework, along with other commercial software. ROS is a distributed framework where nodes publish information that other nodes consume. This model simplifies data communication but poses a major threat because a malicious process could easily interfere the communications, read private messages or even supersede nodes. In this paper we propose that ROS communications should be ciphered. We also measure how this ciphering affects its performance. We have used three different ciphering techniques: DES, AES and RSA. We have evaluated the performance of the system, both from the computing and the communications points of view. Preliminary results show that symmetric ciphers using private keys impose significant delays.**

***Index Terms*—autonomous systems, cybersecurity, robotics, performance, cyber-physical systems, ciphers**

I. Introduction

AUTONOMOUS systems are spreading not just in the virtual world (Internet, software systems), or in science-fiction movies, but in our ordinary real world. Currently we can find driverless cars in the streets, autonomous vacuum cleaners in our homes, museum guides, hotel assistants, etc. These cyber-physical systems, as any computer-based system, can suffer different types of vulnerabilities, and the need of cybersecurity [5] is required.

ROS (Robotic Operating System) [6] has become the most popular framework for developing robotic applications. It started in the research environment, but currently most of manufacturers of commercial platforms use ROS as the *de facto* standard for building robotic software. For example, object-manipulation robots like Baxter (by Rethink robotics)[poner REF] or service robots as our RB1 (by Robotnik)[poner REF] are ROS based platforms.

Our research group is developing assistant robot [3] for the elderly. When we initiated experiments involving potential users, caregivers have asked us about the security of our robot, and about the privacy of its communications [1]. If an assistant robot carrying a camera is deployed in a home, the access to the camera should be secured. Even more when the robot is managing medical information. We have developed all our software for the autonomous behaviour of the robots using ROS, so we need to consider its security.

All authors are with the Robotics Group (http://robotica.unileon.es) and the Research Institute on Applied Sciences to Cybersecurity (http://riasc.unileon.es) at Universidad de León (Spain).

Corresponding author: vicente.matellan@unileon.es

A. ROS security assestment

ROS provides specific libraries for robotics as well as classical operating system services such as hardware abstraction (for sensors and actuators), low-level device control, and inter-process communication. Inter-process communication is based on a graph architecture where computation takes place in ROS processes named nodes. These nodes can receive and send messages, but no security was considered in its design.

ROS framework is basically a message-passing distributed system. Its architecture is based on processes that publish *messages* to *topics*. For instance, a process (*node*) can be in charge of accessing a sensor, making the basic processing of the information, and publishing it as a structure information on a named topic. Another process can *subscribe* to this topic, that is, it can read that information, make a decision about the movement of the robot. These commands will be sent to the motors in another topic. These nodes can be running in the same computer or in different computers.

This approach is very convenient for developers but it is very easily tampered by malicious hackers. For instance in [4] an experiment involving a ROS-based honeypot is described. The honeypot was a radio model truck with two cameras and a compass as sensors and controlled from a remote ROS node written in Javascript and hosted in a remote enterprise grade web server. Vulnerabilities described in the paper range from plain-text communications, unprotected TCP ports to unencrypted data storage.

These vulnerabilities are a subset of the security problems threatening any computing system: No sé si merece la pena, demasiado generalista. Igual es mejor hablar ms de cosas especficas de robtica y de ROS en particular

1) Availability: Interruption
2) Confidenciatily: Interception
3) Integrity: Modification
4) Authenticity :Fabrication

Figure 2 graphically summarizes these problems in our robot.

The first step to solve some of these problems is to secure the communication channels by using a ciphering mechanism. but, how does this systems impact on the performance of a robotic system? This is the goal of this paper, characterize and evaluate different alternatives to secure ROS communication mechanism.

Next section describes the testbed we have designed to measure the performance of the encrypted ROS system. Third section evaluates the data obtained in the experiments and in the last section some conclusions and further work are

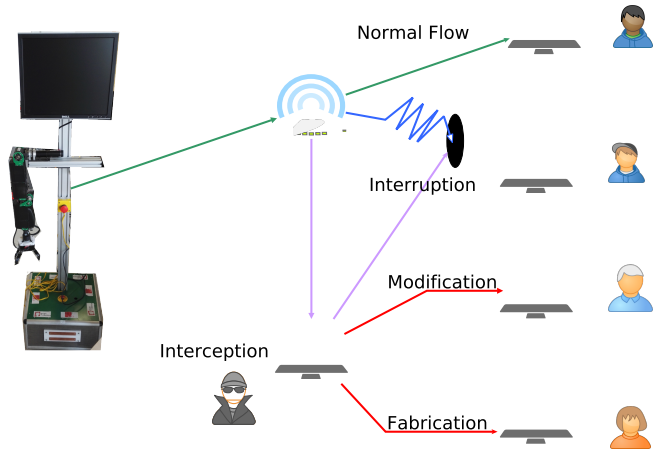

Fig. 1. Conceptual model of the security attacks.

presented.

II. Testbed description

We want to evaluate whether ciphering communications would affect the performance of ROS.

A. Simulated testedbed

We have installed ROS Jade in two computers connected through a wired Ethernet 10/100 switch (model XXXX). In the first computer we have connected a Xtion camera and a Hokuyo laser. In the second computer have run a node visualizing the information from the sensors. Figure **??** shows this environment.

Then we modified the standard ROS implementation. We changed the TCP/IP sockets based implementation by ciphered ones.

B. Robotic testbed

In the second experiment we changed the first computer for a RB1 robot and the XXX switch by a wireless one. This robot was also running ROS Jade.

III. Experimental Measuments

Figure **??** shows the maximum rate that can be reached both in the laser and the camera visualization according to `rviz` information.

The same measurements were made in the second environment to see if the use of wireless systems and a real robot would have any influence.

The absolute values of the frame rates is obviously different, as shown in figure **??**. But the interesting part is the relative different when using clear communications or ciphered ones.

Table **??** compares the relative reduction of speed when using ciphered protocols vs clear ones in both environments as well as the relative increase of CPU usage.

We have addded a funtion to our program in order to measure the time spent in each encrypt and decrypt call. The function is a python method presented as a decorator pattern

```
def fn_timer(function):
  @wraps(function)
  def function_timer(*args, **kwargs):
      v_time_0 = time.time()
      result = function(*args, **kwargs)
      v_time_1 = time.time()
      return result
  return function_timer
```

A. Camera

Encrypt

PC: 234 process

Total length 971790 frames, min time 0.001309 seconds, max time 0.026909 Cifrado mean: 0.010948 Cifrado stdev: 0.000004 El valor moda de la fase de cifrado fue: (array([0.010571]), array([415.]))

Descifrado length: 969295.000000 Descifrado min: 0.001288 Descifrado max: 0.039130 Descifrado mean: 0.008828 Descifrado stdev: 0.000003 El valor moda de la fase de descifrado fue: (array([0.008183]), array([593.]))

PC descifrador: 242 process (ps -ef — wc -l)

Cifrador [INFO] Encrypter node: elasped time: 66755.17 [INFO] Encrypter node: approx. FPS: 14.56

Descifrador [INFO] Decrypter node: elapsed time: 66659.74 [INFO] Decrypter node: approx. FPS: 14.54

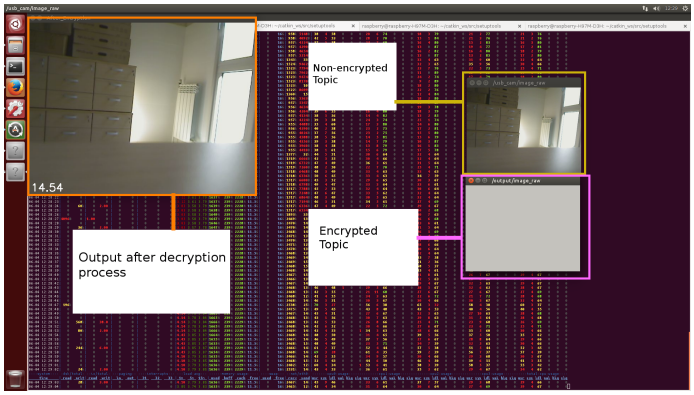

Fig. 3. Time spent in each call to encryption/decryption function.

IV. Conclusion and Futher Work

We have evaluated the influence of cyphering in the performance of ROS based robotic systems.

As we commented in the introduction, we think that securing communications is just one dimension in the cybersecurity of Autonomous Systems. If we want to see autonomous systems working in our homes we need to secure the navigation abilities, the interaction mechanisms, etc.

Some works have been sketched in this area, as for instance in [2].

Acknowledgment

The authors would like to thank the Spanish Ministry of Economy and Competitiveness for the partial support to this work under grant DPI2013-40534-R and to the Spanish National Institue of CyberSecurity (INCIBE) under grand Adenda21 ULE-INCIBE.

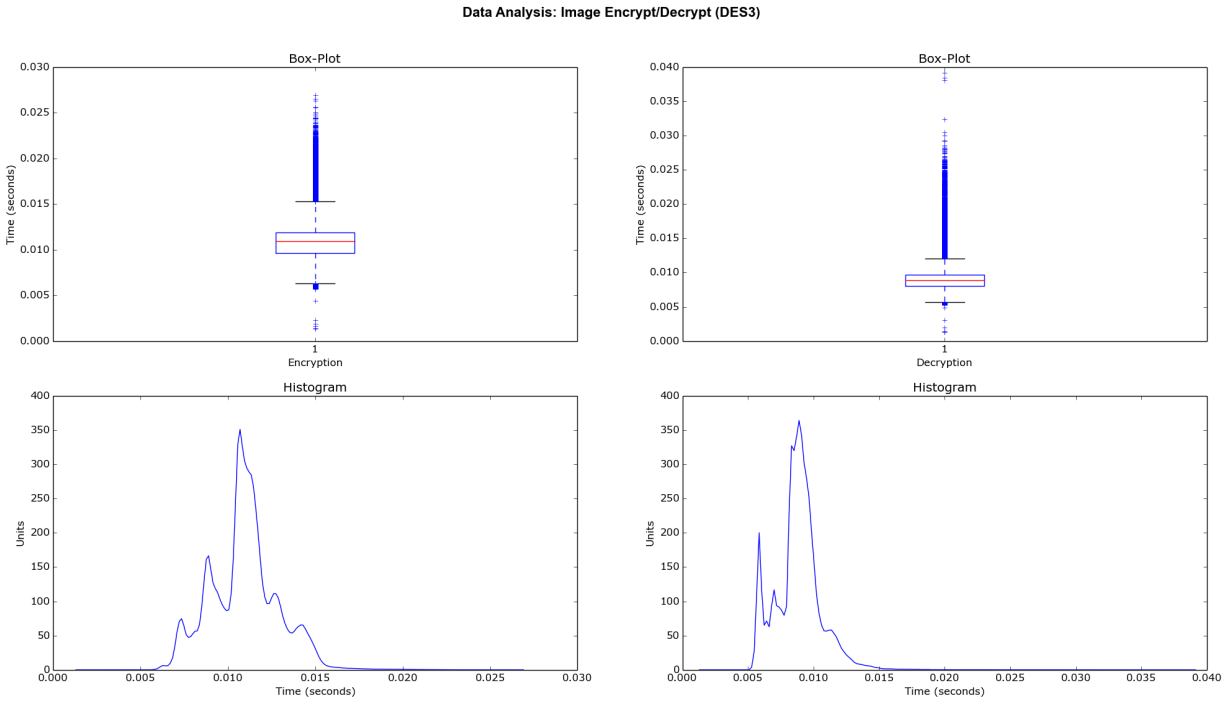

Fig. 2. Time spent in each call to encryption/decryption function.

REFERENCES

[1] Tamara Denning, Cynthia Matuszek, Karl Koscher, Joshua R. Smith, and Tadayoshi Kohno. A spotlight on security and privacy risks with future household robots: Attacks and lessons. In *Proceedings of the 11th International Conference on Ubiquitous Computing*, 2009.

[2] Jérémie Guiochet. Hazard analysis of humanrobot interactions with HAZOPUML. *Safety Science*, 84:225–237, 2016.

[3] Francisco Martn, Jos Mateos, Francisco Javier Lera, Pablo Bustos, and Vicente Matelln. A robotic platform for domestic applications. In *XV Workshop of Physical Agents*, 2014.

[4] Jarrod McClean, Christopher Stull, Charles Farrar, and David Mascareñas. A preliminary cyber-physical security assessment of the Robot Operating System (ROS). *SPIE Defense, Security, and Sensing*, 8741:874110, 2013.

[5] Santiago Morante, Juan G Victores, and Carlos Balaguer. Cryptobotics: Why robots need cyber safety. *Frontiers in Robotics and AI*, 2(23):1–4, 2015.

[6] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.