Poster: Address Resolution Protocol Based Attacks for Multi-Robot Systems

Blake E Todorowski*, Michael Lane Fox*, Harris E Laing*, Kirthan Gaddam*, Anosh Mian*, Rohit Eagala*, Jair Ferrari*, Md Tanvir Arafin*

***Abstract*—Robot Operating System (ROS) is a popular open-source middleware that provides a standard robotic application development framework using commercial-off-the-shelf hardware. As the use of ROS becomes widespread, security vulnerabilities in ROS pose critical concerns for the next generation of robotics. For example, when ROS is used for deploying a multi-robot system (MRS), an attacker can target the weaknesses of ROS to compromise the complete multi-robot infrastructure. This work presents a proof-of-concept hardware demonstration of such an attack to examine how networking vulnerabilities can be exploited to compromise a ROS-based MRS. The experiment setup and demonstration code for this work are available at https://github.com/SPIRE-GMU/MRS_Security.git**

***Index Terms*—Robot Operating System (ROS), Address Resolution Protocol (ARP) Spoofing, Multi-Robot Systems (MRS).**

I. Introduction

RECENT advancements in machine learning and artificial intelligence have unfolded many opportunities for solving long-standing mobility, operation, and control problems in robotics. One such problem is designing intelligent multi-robot systems (MRS) for transportation, manufacturing disaster response, and warehousing. Interestingly, these solutions also require a standard platform on which progress and expertise from different domains can be integrated for real-world deployment. The Robot Operating System, *a.k.a.* ROS, provides a platform for prototyping and integrating robotic applications designed by multiple developers [1]. The design of ROS is modular and allows the developers the flexibility to custom-make the software tools based on the use cases. As a result, ROS has become a cornerstone for the open-source development of robotic applications that can be rapidly prototyped and deployed using commercial-off-the-shelf hardware.

Although the use of ROS has become widespread in modern robotics, the security of ROS did not receive similar attention [2], [3], [4]. As a result, common security vulnerabilities exist in ROS, and security oblivious implementation of ROS will create critical security threats in the future. In this work, we present a demonstration of how a standard cyber security vulnerability can cripple a ROS-based MRS. Here are the key contributions of this work.

- We have developed a ROS-based experimental framework for realizing a multi-robot system using Turtlebots [5].
- We demonstrate an ARP poisoning attack on the MRS framework to derail a leader-follower scheme.
- We have open-sourced the MRS framework and the attack codes and published them at https://github.com/SPIRE-GMU/MRS_Security.git

We plan to present the complete hardware set-up and the attack demonstration at the conference.

*Authors are affiliated with the Cyber Security Engineering Department, George Mason University, Fairfax, VA 22030, USA.
Corresponding Author: Md Tanvir Arafin, email: marafin@gmu.edu

II. Multi-Robot System Design

This research focuses on developing an MRS solution using Turtlebot-3 teleoperation in a live ROS environment. The foundation of ROS MRS relies on a talker-and-listener scheme in which nodes communicate via topics to achieve a goal using the data given to them.

To implement a Teleop MRS, we first leveraged the talker and listener scheme to allow for communication across ROS namespaces, which can be thought of as a unique name to identify the nodes of a robot. We then created teleoperation nodes for two follower robots that take the pre-calculated velocity data of the leader robot and use it for itself, allowing all three robots to move in tandem, as shown in Figure 1.

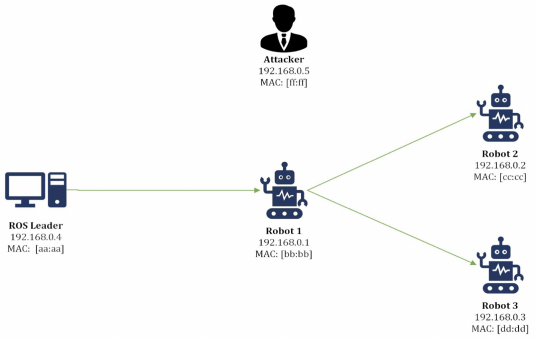

Fig. 1. Multi-robot system setup with three turtlebots for the demonstration. In the pre-attack scenario, a ROS leader node sends a user command to Robot 1 (*i.e.,* the leader robot). Two follower robots (*i.e.,* Robots 2 and 3) follow commands issued by the leader robot.

III. Demo Setup

For our demonstration setup, three robots will be connected to the leader computer (our host computer), as shown in Figure 1. Our host computer will start a ROSCORE server, and the three robots will connect (bring up) to the leader. Once that has been done, the leader will give the teleoperation command to the head robot. Teleoperation is used for remote control in ROS1.

The two follower robots will subscribe to the communication channel where the leader robot publishes its velocity data and utilizes the information. The leader (host computer) will give the head velocity data by selecting control directions, and the lead robot will move accordingly. Simultaneously, the follower robots will listen to the information the leader robot is publishing, interpret it, use the data for themselves, and move alongside it.

IV. Address Resolution Protocol (ARP) Attack

Our demonstration utilizes the Address Resolution Protocol (ARP) protocol's vulnerability for the robots' network connection. ARP lacks reply verification when it comes to receiving packets. This means you can repeatedly send ARP reply packets to a device that uses ARP and the device will treat the packet as if it has sent a response packet already and just received the reply packet late. Once the device receives an ARP reply, it will then update its own ARP table and modify necessary communication paths. Our attack aims to have the targeted robot believe that the threat actor computer is the leader (persona that gives these robots all their commands).

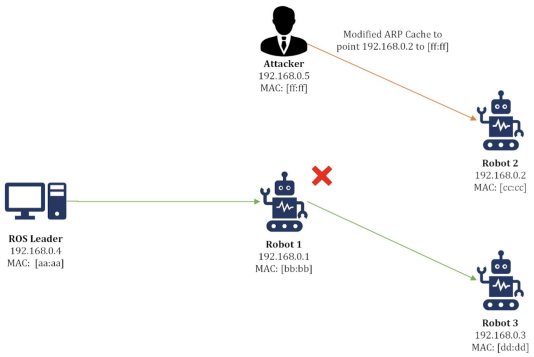

Fig. 2. Demonstration setup for the ARP spoofing attack on a simple multi-robot system.

In the pre-attack scenario, all three robots move simultaneously, per the leader's direction; however, as soon as ARP poisoning targets one of the robots, that robot will no longer be able to receive data from the original leader or other robots. This robot has now been disconnected and is set to point to the threat actor's computer as the leader, as shown in Figure 2. After this attack is launched, the targeted robot will continue to utilize the last data it has heard. Still, it will no longer be able to capture further data from the leader or head robot it previously listened to. This attack can be scaled to multiple numbers of robots, thus temporarily or permanently disconnecting multiple robot nodes from the MRS infrastructure.

Ettercap, a man-in-the-middle tool, is used to run this attack. This tool acts as a proxy on the network and can filter or stop network traffic. In the case of ARP poisoning, Ettercap can redirect traffic from a device to an attacker and simultaneously turn off communications between two devices on the network. Once the attack is run, a robot on the network will no longer receive data from the leader robot and will stay in the current state it is in until it is either turned off or it receives new data from either the leader or an attacker masked as a leader.

Passive Detection of this attack is hard to detect. This is because the robot can return to its normal state once the attack is finished. An attacker can briefly compromise a robot on the network and can immediately remove his connection. This can leave monitoring systems unaware of anything more than a blip in network traffic between two devices. Furthermore, being able to stop communication between the two robots can have a devastating effect. A robot can become entirely out of sync with a system and create unnecessary system maintenance in the future if the sync is not recoverable automatically. Depending on how the robots move in the system, they could be damaged and ultimately break the system, causing it to go down. This could be if the robot were mobile and suddenly lost contact with the leader robot. Instead of stopping, the robot may keep moving from the last command, run into something, and break itself or another object.

To defend against this attack, device communication and network traffic should be monitored for long-term data blockages. If a robot no longer receives data from a leader robot and monitoring shows the leader robot is still sending correct data to the listener robot, a reboot of the compromised robot would be necessary, with a potential system reboot being required.

These tests focus on multi-robot systems running ROS Kinetic and do not cover systems running other versions of ROS or ROS2. Future research is necessary for security on other system versions.

V. Conclusions

This work demonstrates how targeted attacks on the ROS networking protocol can severely compromise a multi-robot infrastructure. Therefore, with the increase in MRSs designed for automation and safety-critical applications, further research and development are warranted for securing ROS-based applications deployed in the wild.

VI. Acknowledgement

This work was supported by NSF Award Number 2245156. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng *et al.*, "ROS: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[2] B. Dieber, B. Breiling, S. Taurer, S. Kacianka, S. Rass, and P. Schartner, "Security for the robot operating system," *Robotics and Autonomous Systems*, vol. 98, pp. 192–203, 2017.

[3] A. Botta, S. Rotbei, S. Zinno, and G. Ventre, "Cyber security of robots: A comprehensive survey," *Intelligent Systems with Applications*, p. 200237, 2023.

[4] J.-P. A. Yaacoub, H. N. Noura, O. Salman, and A. Chehab, "Robotics cyber security: Vulnerabilities, attacks, countermeasures, and recommendations," *International Journal of Information Security*, vol. 21, no. 1, pp. 115–158, 2022.

[5] R. Amsters and P. Slaets, "Turtlebot 3 as a robotics education platform," in *Robotics in Education: Current Research and Innovations 10*. Springer, 2020, pp. 170–181.