Poster: HoneyBot- A Honeypot for Robotic Systems

Celine Irvene, *Member, IEEE,* David Formby, *Member, IEEE,* Sam Litchfield, *Member, IEEE,*
and Raheem Beyah, *Senior Member, IEEE*

Abstract—Historically, robotics systems have not been built with an emphasis on security. Their main purpose has been to complete a specific objective, such as deliver the correct dosage of a drug to a patient, perform a swarm algorithm, or safely and autonomously drive humans from point A to point B. As more and more robotic systems become remotely accessible through networks, such as the Internet, they are more vulnerable than ever. To investigate remote attacks on networked robotic systems we have leveraged HoneyPhy, a physics-aware honeypot framework, to create the HoneyBot. The HoneyBot is the first software hybrid interaction honeypot specifically designed for networked robotic systems. By simulating unsafe actions and physically performing safe actions on the HoneyBot we seek to fool attackers into believing their exploits are successful, while logging all the communication to be used for attribution and threat model creation.

1 Introduction

With the prevalence of robotics growing in all facets of everyday life, robots are becoming a more crucial part of our ecosystem. We rely on them for military purposes on the war front, we rely on them for assisting doctors in the healthcare industry, and even first responders and the police use them. This is not an exhaustive list and if we don't take steps to secure them they will become serious safety threats. In computer security, the first step to securing a resource is traditionally the development of a threat model. A threat model can help assess the probability and the potential harm, which can be useful in minimizing or eradicating the threat. Historically, security in robotics has not been an eminent concern, so to determine a valid threat model these systems must be studied and monitored to learn the scope of attacks they could face. We propose that this should be accomplished with a honeypot specially designed for robotic systems.

Since their inception, honeypots have primarily focused on the traditional IT computing domain, seeking to monitor attackers that aim to compromise company and government workstations/servers. The first honeynet (a network of honeypots) for CPSs (Cyber Physical Systems)/SCADA (Supervisory Control and Data Acquisition) was created by Venkat Pothamsetty and Matthew Franz of the Cisco Infrastructure Assurance Group (CIAG) in 2004 [1]. The goal was to simulate a few popular PLC (Programmable Logic Controller) services to help researchers better understand the risks of exposed control system devices. This work has laid the foundation for many other CPS honeypots [2] [3], none of which are directly applicable to the robotics domain. With the prevalence of robotic systems on the rise, it is critical that advanced monitoring techniques, such as honeypots, be extended to defend them.

Honeypot evasion is, as with most security, a cat and mouse game. There almost always exist configuration fingerprints for any honeypot, and they are corrected as attackers discover them and defenders improve. One way of detecting arbitrary honeypots is to observe a machine's role in a network. Honeypots are, by definition, only interacted with by attackers. If machines in a network exchange no traffic with surrounding hosts, and appear to be unused by regular users, it becomes obvious the machine is a honeypot. The HoneyBot attempts to address these concerns for robotic systems. Firstly, the robotic system exists and is in use, which remedies the traditional lack of context. Secondly, it is not virtualized, but implemented on actual hardware. Finally, the robotic system fully implements the presented services, so all system responses are in line with the HoneyBot devices. The HoneyBot is the first software hybrid interaction honeypot specifically designed for networked robotic systems. By simulating unsafe actions and physically performing safe actions on the HoneyBot we seek to fool attackers into believing their exploits are successful, while logging all the communication to be used for attribution and threat model creation.

2 Robotics

The field of robotics is always changing, but the components that unite almost every class of robots are sensors, actuators, and controllers.

Sensors are the robot's eyes and ears; they enable the robot to understand the world around it and judge features of the environment. The HoneyBot must be able to "spoof", through simulation, any sensor values the robot produces such that an attacker is unaware the commands are not actually performed. To do this we developed device models that provide realistic system responses given an "unsafe/indeterminate" input from an attacker.

The next component common to every class of robots is actuators. Actuators enable the robot to modify the environment and move (or actuate). Since our proof of concept HoneyBot implementation lives in a robot that operates

- *C. Irvene, D.Formby, S. Litchfield, and R. Beyah are with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332.*
E-mail: $\{cirvene3, djformby, slitchfield3, rbeyah\}$ *@gatech.edu*

in networked environments and users/attackers have no physical or visual access to it, this work does not focus on modeling the underlying control system of the actuators. Instead, we emphasize accurately simulating the timing of system responses and how user commands will change the state of the overarching system.

The third, and arguably most important, component of a robot is the control system or controller. The controller, also known as the brain of the robot, enables the robot to parse commands, send signals to different devices, and communicate with other robots as well as the user. The HoneyBot is software that will live in the robot's controller so that it can easily access all data commands and signals to and from the robot's brain allowing it to make decisions accordingly. Figure 1 shows the HoneyBot system architecture. A user/attacker remotely connects to the networked robotic system through the Internet and can send commands. All commands received by the robot will be logged and passed to the *Input Verification* module, which as stated earlier is flexible in structure and can be very comprehensive in the evaluation of commands or relaxed depending on application needs. If the *Input Verification* module deems a command safe, the action will be performed as usual and the system response will be returned to the user/attacker. If, on the other hand, the *Input Verification* module deems a command unsafe the command will be simulated in real time and the ”spoofed” system response will be generated and returned.

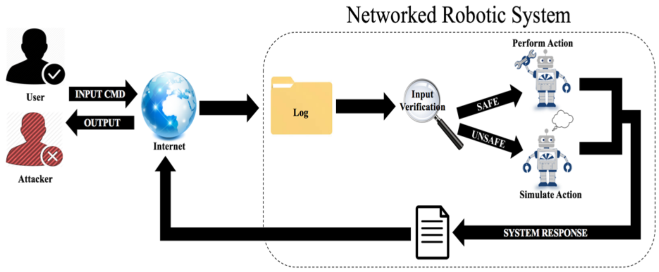

Fig. 1. HoneyBot system architecture

3 Honeypot for Robotic Systems

Given the growing amount of malware targeting CPSs being found, some are beginning to postulate that industrial control systems are the new frontier for cyberattacks. In the past, CPS honeypots have been designed to mimic specific CPS components on a network to protect resources. However, most existing CPS honeypots neglect certain aspects of these systems that can alert an attacker to the nature of the honeypot, namely the physics of the devices that interact with the process [4]. That is why the HoneyBot is based on our previous work, HoneyPhy. HoneyPhy is a physics-aware honeypot framework that accurately models software and protocol fingerprints which are then used to simulate the CPS and fool attackers who access the honeypot [4]. The HoneyPhy framework is composed of three main modules: the *Internet Interface*, the *Process Model*, and the *Device Model* modules. As applied to the HoneyBot, the *Internet Interface Module* is used for opening ports or interfaces on the robotic system so that it can connect to a network. In other words, the *Internet Interface* is the user facing interface, the front end that the attacker can see. The *Process Model* is triggered by an ”unsafe” or ”indeterminate” command, and the action will be simulated in real time by querying the appropriate device model rather than sent to be deterministically performed on the robot. The *Device Model* is different from the other modules in that it contains a model representative of each device found within a robot. These models are built from real data gathered from a given device.

4 Experimentation and Model Building

For model development, we used a combination of techniques including experimentation and physical process modeling to simulate device behavior. The models built are queried at runtime to generate ”spoofed” responses which are sent back to attackers when they perform malicious or otherwise unsafe actions. Suppose the HoneyBot was implemented in a military drone used for finding IEDs (Improvised Explosive Devices) and it received a command directing it to go through a no-fly zone. Clearly, there is something suspicious so the *Input Verification* module flags the action as unsafe. Then, the drone queries its GPS device model and sends back false coordinates, all while maintaining its position in an unrestricted area, but leading the user to believe it is elsewhere. Device models must not only provide realistic state aware data, but they must also reflect the correlation between sensors. For example, a distance sensor must corroborate the reported velocity data, and the velocity data must be in line with encoder readings.

5 Conclusion

The HoneyBot is the first honeypot for robotic systems. Existing honeypots fail to deceive intelligent attackers because they do not accurately model device physics. The HoneyBot addresses this by leveraging HoneyPhy and techniques from traditional honeypots. Device models were built for common robotic sensors and queried to provide convincing system state updates and responses. By simulating unsafe actions and physically performing safe actions on the HoneyBot we can fool attackers into believing their exploits are successful, while logging all the communication to be used for attribution and threat model creation. The HoneyBot is a hybrid interaction honeypot specifically designed for robot systems and should be the de facto standard for robot security as the prevalence of robots grow in society. More details and future updates to HoneyBot can be found at honeybot.gatech.edu.

References

[1] V. Pothamsetty and M. Franz, ”SCADA HoneyNet Project: Building Honeypots for Industrial Networks,” 20 march 2004. [Online]. Available: http://scadahoneynet.sourceforge.net/.

[2] L. Rist, J. Vestergaard, D. Haslinger, A. Pasquale and J. Smith, ”Conpot,” Conpot Development Team, 11 May 2013. [Online]. Available: conpot.org.

[3] K. Wilhoit and S. Hilt, ”The GasPot Experiment: Unexamined Perils in Using Gas-Tank-Monitoring Systems,” Trend Micro, 2015.

[4] S. Litchfield, D. Formby, J. Rogers, S. Meliopoulos and R. Beyah, Re-thinking the Honeypot for Cyber-Physical Systems, 5 ed., vol. 20, Atlanta: IEEE Internet Computing Magazine, 2016, pp. 9-16.