Poster: Using Honeypots to Understand Attacks to Industrial Control Systems

Marcello Pogliani, Mattia Barlocco, Mirko Pesatori, and Stefano Zanero
Dipartimento di Elettronica, Informazione e Bioingegeneria, Politecnico di Milano, Italy
marcello.pogliani@polimi.it, {mattia.barlocco, mirko.pesatori}@mail.polimi.it, stefano.zanero@polimi.it

***Abstract*—We describe our ongoing efforts toward the development of an advanced honeypot that simulates a complex distributed control system (DCS) used in industrial settings such as chemical, oil and gas, water treatment, and food processing plants. Indeed, while anecdotally it is known that ICS are targets of attacks, the details of most incidents are not publicly released (with the exception of high profile cases such as Stuxnet or TRITON). Thus, we believe that, by deploying a honeypot that replicates a real-world deployment of a DCS, we will be able to capture the attempts of attacks toward complex control systems and gain useful insights for the research community. We recently deployed the honeypot in the network of a multinational company that uses the DCS in the course of their business. As a long term goal, we aim to deploy the honeypot on multiple network vantage points, and to collect a repository of ICS attack techniques, as well as ICS malware, to be shared with the security community.**

I. Introduction

Industrial Control Systems are nowadays distributed, integrated with corporate IT systems, and connected (directly or indirectly) to the Internet. This opens a wide array of attack surfaces, compounded by the fact that ICS security is a relatively recent concern, and many systems were originally designed to reside in an air-gapped network. The effects of an attack against an ICS range from data confidentiality issues (e.g., intellectual property stealing) to safety issues, to the interruption of essential services in critical infrastructure, such as water distribution, oil and gas pipelines, and electricity generation systems. Except for a few high profile incidents, such as Stuxnet, the 2015 Ukraine power grid attack, and the 2017 discovery of TRITON, most ICS attacks are not routinely made public. Thus, the research community lacks essential data to study the extent and impact of ICS threats. We propose honeypots as a means to collect ICS threat data.

Collecting threat data using honeypots has been explored in various domains (e.g., worms [1], IoT attacks [2], social networks [3]), including industrial control systems. The goal of most existing ICS honeypots is either scalability [4], ease of deployability (hence, the use of cloud platforms [5]), or the detection of attacks against single devices, such as a single PLC, making it difficult to attract sophisticated attacks against SCADA and DCS networks. Indeed, although ICS are usually deployed on premise, research honeypots often use research or public cloud IP addresses, which could signal that the system is a honeypot; most ICS deployments feature basic security measures, while some honeypots assume directly Internet-addressable control systems using default or easy to guess credentials; some proposals simulate a single device, such as a single PLC, lacking the simulation of a complete network, or use components that are not realistic in a production scenario. To date, publicly available ICS honeypots *lack the realism needed to capture more interesting attacks, particularly in the case of complex systems such as SCADA and DCS networks*.

Thus, as a means to study the behaviour of moderately sophisticated attackers, we design a ICS honeypot by instrumenting a commercial distributed control system (DCS) with OS-level, network-level and ICS-level data collection agents, and we deploy it in a "realistic" IP address space[1].

II. Our Proposal

We implemented and deployed a "pure production" honeypot, which simulates a basic distributed control system (DCS) network loosely following the ISA 99 (IEC 62443) standard.

a) Simulation vs. real DCS: A honeypot must resemble the production system, so that attackers are lured into exploring the target post-compromise, showing attack techniques and goals. We implement a complete (albeit small) ICS network, by instrumenting a commercial distributed control system (DCS) featuring, for instance, engineering workstation software, a HMI with graphical pages adapted from those of a real plant, an historian, and a control processor simulator.

b) Real IP Address: Research honeypots can often be identified from their unrealistic IP address. Thanks to industry collaboration, we deployed our honeypot in the IP address space of a company who use, in their production network, the same DCS software we instrumented.

c) Complexity: We instrument a DCS composed of multiple nodes (virtual machines), and we expose to the Internet only a remote access server to be used by maintenance personnel. We designed our honeypot so that it is not straightforward to compromise, in order to catch also moderately targeted attacks, rather than low-skilled mass attacks.

III. Design

Figure 1 depicts the overall architecture of our honeypot from the network point of view. The honeypot is composed of the DCS network (DMZ, plant and control network), which

[1] To avoid hindering our experimental campaign, we do not mention the name of the DCS software and any detail about the used IP address space.

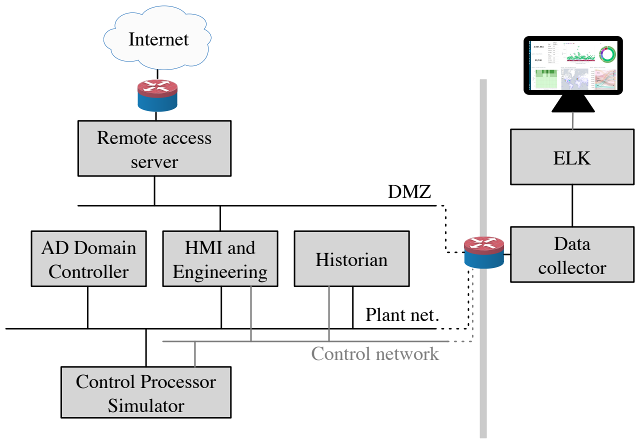

Fig. 1. High-level schema of the instrumented DCS network.

contains an entry point for an attacker and multiple nodes, a monitoring system (divided into a host agent, a network agent and a ICS agent), and a data collection and querying system.

a) DCS network: The DCS runs on a set of Microsoft Windows virtual machines, and is composed of an Internet-exposed "remote access" server, and an internal "plant" network. The plant network contains the HMI, the engineering workstation and the historian; we also deployed an Active Directory domain controller in the plant network. A control network connects the DCS nodes together and with a control processor simulator; the communication along the control network employs a proprietary Ethernet-based protocol.

b) Host Agent: The host agent runs on every honeypot node, collects about OS-level events, and sends them to the data collector. We implemented the agent in user space, extending ossec[2]. Our agent collects operating system logs and changes to registry keys; using Microsoft's WMI subsystem, it monitors executed processes, and file and network socket accesses, and stores the content of any created or modified file; it collects recordings of remote access sessions.

c) Network Agent: We capture and store the full network traffic on all the (virtual) networks the honeypot is composed of, and we use functionalities from the Suricata project[3] to extract netflow information, and to decode application-layer metadata of common protocols.

d) ICS Agent: To study cyber-physical attacks, we collect the state of physical variables of the system that the DCS monitors. We collect this data by analysing the control network traffic. To this extent, we reverse engineered the DCS proprietary protocol to decode packets directed to the historian, which contains the values of historian-monitored data points.

e) Data Collection and Display: All collected data is stored and indexed by Elasticsearch; they can be visualised and queried using our Kibana dashboards. As our goal is to analyse moderately sophisticated attacks, the analysis process is mostly manual, and is triggered by intrusion signals such as a successful login or the creation of an unexpected process. However, to reduce the amount of data shown to the analyst, we filter out a set of OS-level events pertaining to the "normal state" of running the DCS. We tuned this filter by leaving the DCS running for two days, without any interaction.

IV. Preliminary Results

We deployed our honeypot in the IP address space of the production network our partner company. To assess the effectiveness of our monitoring system in reconstructing an attack, we evaluated whether we could reconstruct an attack performed by an external professional penetration testing team. After about a month, we detected more than $500{,}000$ login attempts to the remote access server, with $34{,}000$ unique usernames. Notably, while the most common username was *Administrator*, the third most common username is its the translation in the language of the country of the honeypot IP address—suggesting mass scans with dictionaries targeted to the IP geolocation,— while the username of the remote access user was attempted only 164 times. So far, the honeypot has never been compromised. Besides the short time the honeypot was running, this may be due to the fact that we still left non-trivial barrier to entry: We used relatively weak, but not default, usernames and passwords, and we did not introduce other known vulnerabilities or misconfigurations on purpose.

V. Limitations and Future Work

Instrumenting a real DCS goes a long way with achieving realism. However, it is still possible to distinguish our system from a real plant. First, due to size and licensing constraints, our system is still quite small with respect to a real plant. Second, we were not able to implement a completely realistic simulator using the vendor's provided control processor simulator; to overcome this issue, future work may complete the reverse engineering of the proprietary protocol used in the control network and develop a control processor simulator more suitable for this application.

At the time of writing, the honeypot has been deployed for a month, and we plan to leave it running to collect attack data. In the near future, we plan to experiment with different access levels to the remote access server (e.g., introducing vulnerabilities and misconfigurations, or lowering the guessing effort for the login credentials) to better balance exploitability with our goal of catching sophisticated attacks, and to deploy the honeypot to IP addresses belonging to different countries.

References

[1] D. Dagon *et al.*, "Honeystat: Local worm detection using honeypots," in *Intl. Workshop on Recent Adv. in Intrusion Detection*. Springer, 2004.

[2] P.-A. Vervier and Y. Shen, "Before toasters rise up: A view into the emerging iot threat landscape," in *Proc. Intl. Symp. Research in Attacks, Intrusions, and Defenses*. Springer, 2018.

[3] E. De Cristofaro *et al.*, "Paying for likes?: Understanding facebook like fraud using honeypots," in *Proc. 2014 Internet Measurement Conference*. ACM, 2014.

[4] A. V. Serbanescu, S. Obermeier *et al.*, *A Scalable Honeynet Architecture for Industrial Control Systems*. Springer, 2016, pp. 179–200.

[5] K. Wilhoit, "Who's really attacking your ics equipment?" *Trend Micro Inc., White Paper*, 2013.

[2] https://www.ossec.net/

[3] https://suricata-ids.org/