Evaluating ICS Honeypot Attractiveness: The Role of Interaction Level, Network Type, and Geolocation*

Frederik Ondrikov[a], Denis Donadel[b], Francesco Lupia[c], Massimo Merro[b], Daniel dos Santos[d], Emmanuele Zambon[a], Nicola Zannone[a,*]

[a]*Eindhoven University of Technology, PO Box 513, Eindhoven, 5600 MB, The Netherlands*
[b]*University of Verona, Strada le Grazie 15, Verona, 37134, Italy*
[c]*University of Calabria, Via Pietro Bucci, Rende CS, 87036, Italy*
[d]*Forescout Technologies, John F. Kennedylaan 2, Eindhoven, 5612 AB, The Netherlands*

Abstract

Honeypots are employed in Industrial Control Systems (ICSs) to detect and analyze attacker behavior, but their deployment involves trade-offs between realism, operational cost, and exposure environment. Despite increasing interest in ICS threat monitoring, there is limited empirical evidence on how deployment choices influence the effectiveness of ICS honeypots in attracting meaningful interactions. This work investigates the effect of interaction level, network type, and geographic location on the attractiveness of ICS honeypots. We deploy 16 honeypots, a mix of low- and high-interaction emulations and a physical PLC, across a corporate network and cloud networks in multiple geographic regions, and collect HTTP, S7Comm, and Modbus traffic over a three-month period. Analyzing multiple protocols allows us to capture differences in attacker behavior across web-based and ICS-specific traffic and to assess how protocol diversity influences the volume, complexity, and nature of observed interactions. To evaluate traffic nature, we fingerprint recurring Modbus and S7Comm interactions using protocol-specific features and associate them with known tools or actors. Our results show that, for ICS traffic, network type has the largest influence, while interaction level and

*This work is an extended and revised version of [35].

*Corresponding author

Email addresses: f.ondrikov@student.tue.nl (Frederik Ondrikov), denis.donadel@univr.it (Denis Donadel), francesco.lupia@unical.it (Francesco Lupia), massimo.merro@univr.it (Massimo Merro), daniel.dossantos@forescout.com (Daniel dos Santos), e.zambon@tue.nl (Emmanuele Zambon), n.zannone@tue.nl (Nicola Zannone)

geographic location have a limited impact. We also find that low-interaction honeypots capture traffic comparable to high-interaction setups, supporting their use for general threat intelligence collection. While most traffic consists of automated reconnaissance, filtering it reveals more complex activities, such as multi-stage campaigns, cross-environment scans, and device manipulation.

Keywords: Industrial Control Systems, Honeypots, Data analysis, Fingerprinting.

1. Introduction

Industrial Control Systems (ICSs) are physical and engineered systems whose operations are monitored, coordinated, and controlled by interconnected computing and communication components [40]. ICSs include Operational Technology (OT), such as Supervisory Control and Data Acquisition (SCADA), and embedded devices, such as Programmable Logic Controllers (PLCs), which are foundational for process control in critical infrastructures. Modern ICSs are increasingly integrating Information Technology (IT) capabilities into legacy OT environments. While this integration improves connectivity and enables remote access, it also expands the attack surface and introduces new cybersecurity risks. ICSs usually have long operational lifespans, often without significant upgrades, leaving them vulnerable to emerging cyber threats. Notable incidents such as the Stuxnet worm, which targeted Iran's nuclear program [10], and the Industroyer malware, deployed against Ukraine's power grid [24], illustrate the severe impact that ICS vulnerabilities can have on real-world critical infrastructure.

To better understand and mitigate these threats, honeypots are increasingly deployed as decoy systems designed to attract attackers, allowing analysts to capture and analyze malicious behavior. They serve as early warning systems and deception mechanisms that divert attacks from critical assets [39]. Honeypots can differ in their level of interaction. Low-interaction honeypots expose limited functionality, are easier to deploy with minimal risk, but are often easily fingerprinted and ignored by advanced attackers or scanners such as Shodan [25, 30, 42, 45, 49]. High-interaction honeypots, on the contrary, mimic real systems more closely, making them harder to detect and better suited to observe complex attacker behavior [23, 39]. However, they require greater operational effort and introduce a higher risk if compromised. In practice, operators must therefore balance the expected value of collected data against deployment complexity and operational overhead.

In addition to interaction level, deployment environment and geographic location may also influence honeypot effectiveness. ICS infrastructures are tradi-

tionally hosted within corporate networks, whereas cloud-based deployments are increasingly used to reduce operational risk and simplify management. However, differences in perceived realism, exposure patterns, and network characteristics may affect how attackers engage with these systems. Geographic location may also influence a honeypot's attractiveness, due to regional targeting preferences, localized threat actors, or geopolitical factors. Despite their practical relevance, the relative importance of these deployment factors remains unclear.

Prior research has examined individual aspects of ICS honeypot design, including interaction level [13, 27, 28, 44], deployment environment [8, 21, 43, 46], and geographic distribution [5, 13, 44]. However, these aspects are typically studied in isolation, making it difficult to assess their relative impact or to derive guidance for cost-effective deployment strategies. Systematic comparisons of ICS honeypots with varying levels of interaction, including physical devices, remain largely absent. Reported differences in deployment environments [5, 8] emerge from IT-focused analyzes or lack the controlled conditions necessary to isolate the influence of the network environment. Similarly, although geographic location has been shown to affect attack patterns in IT environments [9, 13, 53], its impact in ICS remains largely underexplored. Moreover, existing analyses frequently rely on packet-level inspection and coarse filtering heuristics [9, 17, 21, 27], which limits the ability to distinguish automated scanning activity from more complex or targeted interactions. As a result, current evidence provides limited insight into how deployment choices influence not only the quantity but also the nature and complexity of the collected ICS traffic. A systematic evaluation of how key deployment characteristics influence ICS honeypot attractiveness is therefore needed to support evidence-based deployment decisions and to guide the design of cost-effective monitoring infrastructures.

Contributions. This work investigates how interaction level, network type, and geographic region influence the attractiveness of ICS honeypots, with the aim of identifying cost-effective deployment strategies. To this end, we deploy 16 ICS honeypots using the data collection infrastructure of Forescout Technologies, a cybersecurity company specialized in threat detection and operational technology security. Our setup includes a physical PLC (Siemens Simatic S7-1200) and both low- and high-interaction honeypots emulating the same device, distributed across two network types (corporate vs. cloud) and three geographic regions (Europe, North America, Asia). The honeypots are exposed to the Internet for three months, during which we collect HTTP, Modbus, and S7Comm traffic. We evaluate honeypot attractiveness by traffic volume and complexity. We also perform a qualitative analysis to assess the nature of the captured ICS traffic. To this end, we fingerprint

recurrent Modbus and S7Comm interaction patterns using protocol-specific features and associate them with known tools or actors. This allows us to evaluate the extent to which observed ICS traffic is generated by automated tools such as scanners. We then manually analyze the interactions not matching existing fingerprints to infer the potential intent and identify novel or targeted behavior. The main contributions are as follows:

- Our study shows that network type has a stronger influence on ICS honeypot traffic than geographic location or interaction level. Corporate deployments attracted significantly more S7Comm traffic and sophisticated behavior, highlighting the importance of protocol-specific analysis and deployment environment in designing and deploying cost-effective ICS honeypots.
- Our analysis shows that most ICS traffic consists of automated reconnaissance, with S7Comm interactions largely generated by open-source tools and Modbus interactions showing greater tool diversity. Filtering out this automated activity reveals the presence of sophisticated, targeted behaviors, including multi-stage campaigns, cross-environment scans, reverse-engineered clients, and device manipulation. These findings show the value of systematic filtering and fingerprinting for uncovering complex interactions hidden by background scanning traffic.
- Our results demonstrate the need to align the honeypot type with its operational goals. We observed that high- and low-interaction ICS honeypots capture similar traffic for general threat intelligence, making low-interaction setups a cost-effective choice. However, we also observed occasional complex interactions for which high-interaction honeypots could better support the investigation of attack methods and post-exploitation activities.

Outline. The remainder of this paper is organized as follows. Section 2 reviews relevant background and prior research on ICS honeypots. Section 3 introduces our research questions. Section 4 describes the deployed infrastructure and the methods used to address the research questions. Section 5 presents the results, and Section 6 discusses the key findings. Finally, Section 7 concludes the paper.

2. Background and Related Work

This section introduces background on ICS honeypots and reviews related work, identifying research gaps.

2.1. ICS Honeypots

ICS honeypots serve multiple purposes, including diverting attackers from critical assets, studying adversary behavior, and gathering threat intelligence. Like

traditional honeypots, they are categorized by interaction level. Low-interaction honeypots simulate basic network behavior without replicating the full device functionality, relying on scripted actions that attackers may eventually recognize [25]. Despite this, they remain popular for their ease of deployment, simplicity, and lower operational risk. High-interaction honeypots, on the other hand, offer richer emulation by supporting full process control and interaction with simulated or physical environments, though they require more complex setup and pose greater security risks. Recent frameworks like HoneyICS [26] have emerged to provide high-interaction capabilities by virtualizing PLCs and linking them to simulated industrial processes. Built on OpenPLC [4], a widely used open-source virtual PLC, HoneyICS supports Modbus natively and integrates Snap7 [31] to enable S7Comm communication. It allows interaction with a simulated industrial process, offering higher realism compared to low-interaction honeypots like Conpot [28]. Understanding the trade-offs between these honeypot types is crucial for designing effective experiments and studying attacker behavior in realistic ICS scenarios.

2.2. Related Work

Research on ICS honeypots spans several dimensions, including the comparison of interaction levels, the influence of network type and geographic location, and the analysis of network traffic and payloads. Table 1 summarizes the literature in this domain.

Interaction Level. The attractiveness of honeypots with different interaction levels has been explored primarily in IT and IoT environments. Early work by Pouget et al. [38] and Alata et al. [2] examined high- and low-interaction honeypots in IT environments. They found high-interaction honeypots valuable for gaining deeper insight into attacker behavior, with most intrusions attributed to inexperienced attackers using automated tools. While foundational, these studies are dated and might not reflect current attack automation or advancements in honeypot technologies. More recent research [23] confirmed that high-interaction honeypots produce richer behavioral data, despite their complexity. In IoT, Guarnizo et al. [17] observed that high- and low-interaction honeypots attracted similar attack patterns, suggesting that simulating realistic environments does not necessarily increase the frequency or sophistication of attacker interactions. Whether these findings extend to ICS environments remains unclear. In the ICS domain, several studies investigate the effectiveness of honeypots but did not compare interaction levels. Jicha et al. [20] and Bieker et al. [5] deployed low-interaction honeypots like Conpot and GridPot, but do not systematically analyze the collected traffic. On the other hand, Dodson et al. [13] and Lupia et al. [27] examined the attractiveness of various high-interaction

Table 1: Overview of related work

	Domain	Interaction Level				Deployment Environment		Traffic Analysis	
		Low-Interaction Honeypot	High-Interaction Honeypot	Physical PLC	Level of Interaction Comparison	Network Type Comparison	Region Comparison	Payloads Analysis	Automated Tools
Pouget et al. [38]	IT	◐	●	○	◐	○	○	●	○
Alata et al. [2]	IT	◐	●	○	◐	○	○	●	○
Bloomfield et al. [8]	IT	◐	●	○	◐	●	○	○	○
Sochor et al. [43]	IT	●	○	○	○	●	○	◐	○
Bove et al. [9]	IT	●	○	○	○	●	●	◐	◐
Kelly et al. [21]	IT	●	○	○	○	◐	●	◐	◐
Kocaogullar et al. [23]	IT	●	●	○	●	○	○	●	○
Zou et al. [53]	IT	●	○	○	○	◐	◐	◐	○
Guarnizo et al. [17]	IoT	●	●	○	●	◐	◐	◐	◐
Tambe et al. [46]	IoT	○	●	○	○	◐	◐	●	◐
Jicha et al. [20]	ICS	●	○	○	○	○	○	○	○
Dodson et al. [13]	ICS	○	●	○	○	○	◐	●	◐
You et al. [50]	ICS	◐	●	●	●	○	◐	○	○
Bieker et al. [5]	ICS	●	○	○	○	●	●	●	○
Maesschalck et al. [28]	ICS	●	○	●	◐	○	○	○	○
Lupia et al. [27]	ICS	○	●	○	○	○	○	●	◐
Conti et al. [11]	ICS	○	●	○	○	●	○	○	◐
Mesbah et al. [29]	ICS	●	○	○	○	○	○	○	○
Vasilatos et al. [48]	ICS	○	●	○	○	○	○	◐	○
Shan et al. [41]	ICS	○	●	○	○	○	○	○	○
Srinivasa et al. [44]	IoT/ICS	●	●	○	●	●	●	●	◐
This work	ICS	●	●	●	●	●	●	●	●

Legend: ○= Not Discussed, ◐= Partially Discussed, ●= Fully Discussed.

ICS honeynet configurations. Only a few studies directly compared interaction levels in ICS honeypots. Srinivasa et al. [44] compared RioTPot and Conpot and observed that the high-interaction honeypot attracted more traffic. However, differences in protocol support weaken these findings. Some studies, such as [28, 50], also considered physical devices. You et al. [50] deployed a Siemens S7-300 PLC to assess function code coverage and briefly noted differences in interaction levels, but the device lacked a physical process, making it relatively static and less attractive to attackers. Moreover, their study did not include an evaluation of attacker behavior. Recent work further indicates that the presence of a realistic physical process plays a key role in attracting ICS-relevant interactions [11, 48]. For instance, Conti et al. [11] show that realistic physical-process ports in high-interaction environments increase honeypot believability and produce more targeted interaction sequences consistent with ICS reconnaissance and manipulation. Similarly, Shan et al. [41] show that ICS honeypots with physical process simulation attracted longer interactions compared to honeypots without physical process simulation. This suggests process-level realism, beyond interaction level alone, may influence attacker engagement. Maesschalck et al. [28] tested different Conpot setups along-

side a physical PLC, though the device was not connected to the Internet, limiting insights into their differences in attractiveness. Overall, although high-interaction honeypots, especially those incorporating physical devices, have been hypothesized to yield more realistic and detailed insights, few ICS studies have systematically assessed how interaction level influences honeypot attractiveness. The comparative benefits of different types of ICS honeypots thus remain largely unexplored.

Network Type. Several studies have examined how network type affects honeypot interactions. Bloomfield et al. [8] compared high-interaction IT honeynets in corporate and small-to-medium enterprise networks, finding that corporate setups attracted more attacks. They attributed this difference to greater network visibility and the higher likelihood that attackers perceive corporate infrastructure as more valuable. Sochor et al. [43] demonstrated that attackers differentiate between IP address ranges associated with academic and commercial networks. Kelly et al. [21] reported the highest attack volume on Google Cloud among IT honeypots deployed on AWS, Azure, and Google Cloud, although geographic factors may have influenced the outcome. Other studies deployed honeypots across different network types without analyzing their impact. For instance, Guarnizo et al. [17] and Tambe et al. [46] used local networks with cloud-based forwarders for IoT honeypots, but did not assess how the environment influenced honeypot activity. Bove et al. [9] observed little variation in SSH intrusion patterns between cloud and internet-connected systems. Bieker et al. [5] observe lower ICS traffic in cloud honeypots than academic networks, attributing this to the scarcity of ICS devices in the cloud, although their sequential deployment limits confidence in their findings. Evidence from ICS deployments also shows qualitative differences between cloud-hosted and local environments. For instance, Conti et al. [11] report that public cloud setups generate more generic scanning, while cautious or targeted adversaries more often engage with field-like industrial contexts. Their comparison of cloud infrastructure and local exchange points shows that, although physical-process emulation encourages engagement, traffic volume and composition depend on network location. In contrast, Srinivasa et al. [44] conducted a concurrent deployment of honeypots in corporate and cloud networks, finding more malicious ICS traffic targeting the cloud-based instance. However, they interpreted scanning behavior as inherently malicious, which might not be accurate, as some scans could originate from benign sources. Although these studies suggest that network type influences attacker behavior, systematic evaluations in ICS settings are lacking. Most prior work focuses on IT systems and lacks controlled, parallel deployments that isolate the role of network type.

Region. Prior research has emphasized the value of deploying honeypots across diverse locations to capture a wider range of attack strategies [13]. Building on this, Zou et al. [53] deployed high-interaction IT honeypots in East Asia, Western Europe, and the US East via Microsoft Azure, observing notable regional differences in traffic volume and unique IP addresses. Similar observations have also been reported in other works. Bove et al. [9] and Kelly et al. [21] both reported regional variation in attack activity. Srinivasa et al. [44] found that Europe attracted the most malicious IoT/ICS traffic, followed by the US and Asia, while Bieker et al. [5] observed higher ICS traffic in Asia than in the US. Despite these insights, most studies rely on IT honeypots, leaving gaps in our understanding of how geographic location influences attacks in operational technologies like ICS. Additionally, the lack of standardized experimental setups across regions makes it difficult to isolate geographic effects from other confounding variables.

Traffic Analysis. Several studies have analyzed network traffic collected using honeypots, particularly focusing on the inspection of payloads to classify and understand attacker behavior. A number of works [9, 17, 21, 27] employed Suricata to generate rule-based alerts, which were used to guide their analysis of honeypot traffic. However, this tool operates at a packet level, missing more complex behaviors that originate from network flows. Other approaches involved the usage of Zeek to extract interactions [27], which are more meaningful when analyzing network traffic. Some studies have explored the presence of automated tools, particularly in IT [9, 21] and IoT contexts [17, 46], though without distinguishing between individual tools or examining their specific usage patterns. In the ICS domain, Dodson et al. [13] conjecture that network traffic is largely automated, but without providing a systematic analysis of the tools used. Lupia et al. [27] revealed a higher prevalence of automated interaction in the IT traffic than in ICS traffic, though only a few tools were explicitly identified. Srinivasa et al. [44] excluded traffic from known scanners (e.g., Shodan) and categorized the remaining interactions into five types based on their apparent goals (e.g., brute-force, poisoning), but without conducting a systematic analysis of the tools used. Recent work further highlights the value of protocol- and port-level behavioral signals for distinguishing opportunistic automated activity from more deliberate ICS-oriented interaction. For instance, Mesbah et al. [29] show that frequencies of ICS protocol requests and use of physical-process ports differentiate automated scanners from targeted actors, complementing payload inspection by revealing behavioral intent.

2.3. Research Gaps

Although prior work has advanced our understanding of ICS honeypots, several key gaps remain. Only a limited number of studies compare interaction levels in ICS environments, and existing comparisons rely on heterogeneous configurations that hinder interpretation. For example, Srinivasa et al. [44] compared honeypots with different protocol support, while Maesschalck et al. [28] evaluated a physical PLC that was not Internet-exposed. These design differences prevent a systematic assessment of how interaction level influences honeypot attractiveness in ICS settings. As a result, the trade-off between deployment complexity and effectiveness in attracting meaningful interactions remains insufficiently understood.

The influence of the deployment environment also remains unclear. Prior studies report differences between cloud and non-cloud deployments, but they either focus on IT systems [8, 21, 43] or rely on sequential or non-controlled ICS deployments [5]. Although Srinivasa et al. [44] conducted concurrent deployments, their classification of scanning activity as malicious limits conclusions regarding attractiveness. These limitations make it difficult to isolate the role of network type in ICS honeypot deployment. Geographic location is another underexplored dimension in ICS environments. While regional differences have been observed in IT honeypots [9, 21, 53], ICS-specific evidence remains limited and inconsistent [5, 44]. Moreover, prior studies typically vary multiple deployment factors simultaneously, preventing a clear assessment of geographic effects.

The relationships between deployment factors, particularly network type and geographic location, are rarely investigated. Existing work typically evaluates interaction level, network type, or geographic location in isolation, limiting understanding of their combined influence on honeypot attractiveness and attacker engagement in real-world ICS environments. A broader, systematic evaluation is therefore needed to assess how these factors jointly affect honeypot attractiveness and to guide the design of cost-effective, targeted deployments.

To understand the benefits of different honeypot configurations, it is also essential to assess their ability to attract meaningful threats and generate actionable insights. However, analyzing network traffic from ICS honeypots presents significant challenges. The data volume is typically large, making manual inspection impractical. Moreover, this traffic is often dominated by automated and opportunistic activity, introducing noise that complicates the identification of targeted or sophisticated attacks. To this end, several studies rely on rule-based systems such as Suricata for traffic characterization [9, 17, 21, 27]. While these systems can flag known patterns, they operate at the packet level, offer limited support for ICS-specific protocols, and provide little contextual information, hindering the

reconstruction of attack sequences or the understanding of attacker intent. Other studies identify automated activity but do not systematically analyze the tools used or their behavioral patterns [13, 27, 44], thereby causing much traffic to remain for manual analysis. Overall, these limitations hinder the identification of meaningful or targeted interactions within large volumes of traffic.

Addressing these gaps is necessary to support evidence-based decisions on ICS honeypot deployment that balance operational cost with the ability to capture and analyze relevant threats. This requires a systematic analysis of how deployment factors, such as interaction level, network type, and geographic location, influence ICS honeypot attractiveness, as well as moving beyond basic traffic classification to identify interactions that may signal more targeted or meaningful threats.

3. Research Questions

Previous studies have examined various honeypot types and configurations, often focusing on isolated aspects (cf. Section 2.2). However, attacker engagement may depend on a broader, potentially interdependent set of factors, including interaction level, network type, and geographic location. This work systematically investigates how these factors influence the attractiveness of ICS honeypots.

Our first research question examines how an ICS honeypot's interaction level affects its attractiveness. High-interaction honeypots, including those involving physical PLCs, offer the potential to capture more detailed and realistic attacker behavior. However, they are also more complex to deploy and operate. Understanding their impact helps assess under which conditions the added complexity is justified.
RQ1: *To what extent does the level of interaction supported by an ICS honeypot influence the attractiveness of the honeypot?*

Realistic IP addresses are key for honeypots to avoid detection [13]. While cloud setups offer convenience and isolation, ICS devices are typically tied to physical infrastructure, making corporate deployments appear more authentic. This raises the question of whether corporate-based ICS honeypots are more attractive than cloud-hosted ones.
RQ2: *To what extent does the network type on which an ICS honeypot is deployed influence the attractiveness of the honeypot?*

Prior work suggests that honeypot attractiveness and attacker behavior can vary by geographic region, motivating the deployment of distributed honeypots [13]. To investigate this, ICS honeypots should be deployed in diverse locations to assess whether geographic location affects attractiveness and reveals potential regional differences in attackers' engagement.

RQ3: *To what extent does the geographic location of an ICS honeypot influence the attractiveness of the honeypot?*

To assess the trade-off between different honeypot configurations, it is not enough to quantify traffic volume; it is also essential to evaluate whether they attract meaningful threats and provide insights into attacker behavior. This remains challenging in ICS contexts due to high data volumes dominated by automated, low-value interactions [13, 27], which hinder the identification and analysis of more targeted or sophisticated activity. Current work often relies on packet-level, rule-based systems like Suricata, which provide only coarse filtering and lack contextual awareness—leaving much traffic unclassified and requiring manual inspection. To address these limitations, we pose two research questions:

RQ4: *Which tools are commonly used for ICS interactions, and what actions are associated with them?*

Identifying specific tools and their behavior helps differentiate between routine scanning and more targeted or harmful activity, offering a clearer view of the capabilities of different honeypot configurations.

RQ5: *What patterns or characteristics emerge from ICS interactions that cannot be attributed to known tools or exploits?*

By filtering out known tool-based activity, we can focus on potentially novel or manual interactions, gaining insight into higher-value threats and attacker intent.

4. Methodology

An overview of our methodology is shown in Figure 1. After defining the experiment conditions (Section 4.1), we deploy the data collection infrastructure, including honeypots, networks, and storage systems, and collect data over a 90-day period (Section 4.2). To answer the research questions, we evaluate protocol-specific attractiveness across honeypots by comparing the captured traffic based on interaction level (RQ1), network type (RQ2), and geographic region (RQ3). For RQ4 and RQ5, we identify and fingerprint recurrent Modbus and S7Comm interaction patterns based on protocol-specific features and associate them with known tools or actors. This analysis allows us to evaluate the extent to which observed ICS traffic is generated by automated tools (RQ4). Finally, we manually examine interactions not covered by the fingerprints to identify targeted behaviors and infer their potential intent (RQ5).

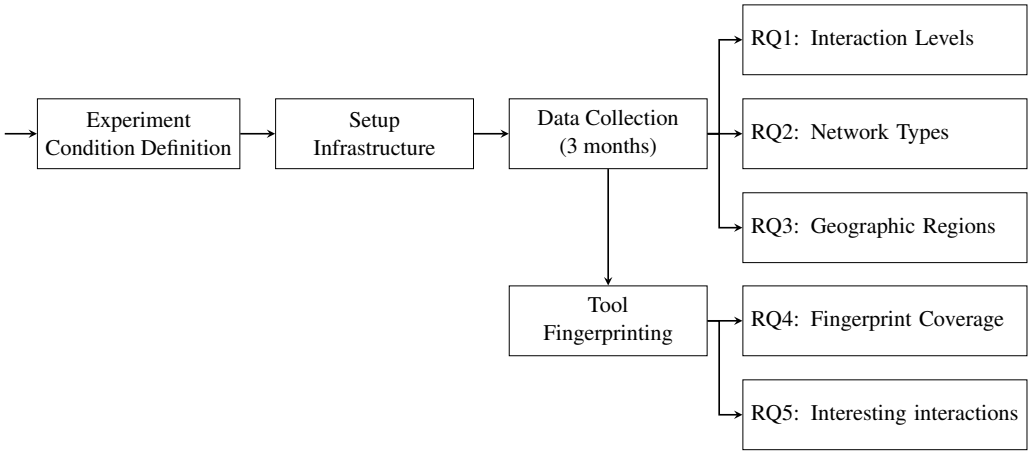


Figure 1: Methodology overview

4.1. Experiment Conditions

We define 12 experiment conditions in our study. Each condition corresponds to a specific configuration of the honeypot, determined by three variables: the *interaction level* it supports, the *type of network* in which it is deployed, and the *geographic region* of deployment. We now discuss the choices made for each of these variables.

Interaction Levels. To assess how interaction level affects honeypot attractiveness (RQ1), we deploy three setups: a *low-interaction ICS honeypot*, a *high-interaction ICS honeypot*, and a *physical ICS device*. To allow meaningful comparison, all honeypots simulate the same device type and expose the same protocols. In particular, we focus on two widely used ICS protocols: Modbus and S7Comm. These protocols are selected due to their broad adoption in industrial environments and their relevance in existing threat intelligence studies. We select the S7-1200 as the physical PLC due to native support for Modbus and S7Comm and wide ICS adoption. We use Conpot [28] for low interaction and HoneyICS [26] for high interaction. Both can simulate a Siemens S7-1200 PLC, support Modbus and S7Comm, and host a static web server. All setups appear similar externally but differ in interaction level: Conpot gives static responses, HoneyICS offers process interaction via Modbus only, and the physical PLC offers process interaction via both protocols and full device configuration via S7Comm. Table 2 details the different interaction levels of the three setups.

Table 2: Function coverage for S7Comm (left), Modbus (center), and HTTP (right)

S7Comm Function	Conpot	HoneyICS	PLC
Setup Communication	◐	◐	●
CPU Information	○	◐	●
CPU State	◐	◐	●
Start PLC	◐	◐	●
Stop PLC	◐	◐	●
Read Data Block	○	◐	●
Write Data Block	○	◐	●
Project Upload	○	○	●
Project Download	○	○	●

Modbus Function Code	Conpot	HoneyICS	PLC
01 (Read Coils)	◐	●	●
02 (Read Discrete Inputs)	◐	●	●
03 (Read Holding Registers)	◐	●	●
04 (Read Input Registers)	◐	●	●
05 (Write Single Coil)	◐	●	●
06 (Write Single Register)	◐	●	●
15 (Write Multiple Coils)	◐	●	●
16 (Write Multiple Registers)	◐	●	●
17 (Report Slave ID)	◐	○	○
43 (Report Device Information)	◐	○	○

HTTP Permission	Conpot	HoneyICS	PLC
CPU Diagnostics	◐	◐	●
Flash LEDs	○	○	●
Change Operating Mode	○	○	●
CPU Maintenance	○	○	●
Tag Access	○	○	●
User-Defined Web Pages	○	○	●
Filebrowser Access	○	○	●

Legend: ○= Not Supported, ◐= Partially Supported, ●= Fully Supported.

Network Types. To assess whether network type affects honeypot attractiveness (RQ2), we consider a *corporate network* and a *cloud-based network*. The corporate network reflects a realistic ICS setup, with on-premise devices accessible via public IPs. In contrast, cloud networks are commonly used only for honeypot deployments [5, 9, 21]. By comparing the two, we aim to determine whether they are perceived differently in terms of attractiveness. To control for geographic bias, one cloud deployment is placed in the same region as the corporate network (Europe); see Section 4.2 for details.

Geographic Regions. To assess the influence of geographic region (RQ3), we consider three global regions: *Europe*, *North America*, and *Asia*. This selection follows prior research (cf. Section 2.2) and Shodan scans showing these areas host the most publicly accessible ICS devices. According to Shodan, the US has the highest number of exposed ICS devices using Modbus and S7Comm, making it the best representative for North America. While China ranks second, infrastructure restrictions prevent server deployment there; thus, we choose Singapore, ranked third, to represent Asia. The choice for Europe reflects the location of the corporate network available for our study (see Section 4.2).

4.2. Data Collection

4.2.1. Infrastructure

Based on our experiment conditions (Section 4.1), we set up an infrastructure to enable data collection (Figure 2). The infrastructure encompasses 16 ICS honeypots.[1] All honeypots expose HTTP, S7Comm, and Modbus services. They are deployed on Forescout's corporate network and exposed on the cloud network

[1]Note that we deploy two instances of HoneyICS, resulting in 16 honeypots across the 12 experiment conditions in Section 4.1. This setup allows us to estimate the variability in the collected traffic.

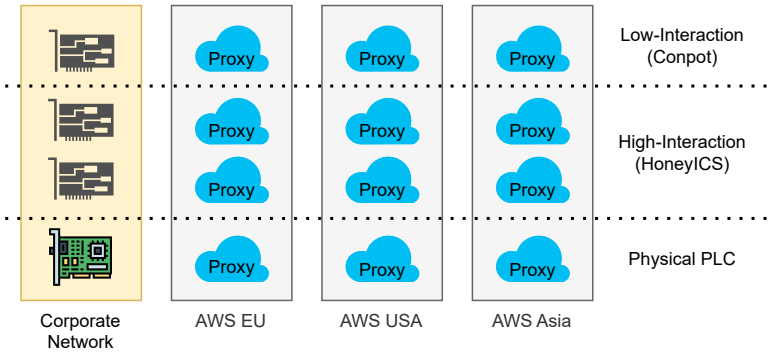


Figure 2: Architecture of the honeypots and AWS proxies

using proxies. This ensures consistent behavior across network types and regions. We now describe key details of the infrastructure.

Honeypots. We deploy Conpot as a Docker container and customize it to mimic a Siemens S7-1200 PLC. Customization involves modifying Conpot's default templates to update its web interface and the scripted S7Comm and Modbus responses. We deploy two HoneyICS instances in a Docker environment that simulates a production process controlled by virtualized S7-1200 PLCs. Each virtualized PLC exposes the same services (Modbus and S7Comm) and a web interface. The physical Siemens S7-1200 PLC interacts with HoneyICS via Modbus and controls part of the same (simulated) physical process. To this end, its ladder logic is built and loaded using Siemens TIA Portal software. To ensure continuous operation under attack, an automated script monitors the PLC integrity and restarts it when needed (e.g., if forced into STOP mode).

Deployment. We deployed our honeypots using the data collection infrastructure of Forescout Technologies, including a subscription to Amazon Web Services (AWS), which enables controlled exposure of the deployments across environments. Specifically, the honeypots are hosted on a corporate network in the Netherlands, each with a unique public IPv4 address leased from a Dutch ISP that supports enterprise and ICS environments to ensure realistic deployment conditions. For the cloud instances, we leverage Forescout's subscription to AWS. Based on our experiment conditions (Section 4.1), we selected three AWS regions: US East (North Virginia), Asia Pacific (Singapore), and EU Central (Frankfurt), which is the AWS region closest to the corporate network. Instead of duplicating instances, we deploy lightweight proxies on AWS that forward traffic to the corporate-hosted honeypots. Each proxy has a dedicated Elastic IP to ensure stable public access

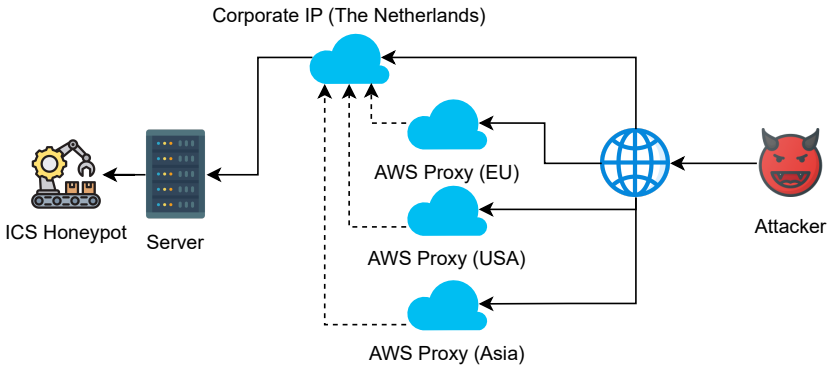


Figure 3: Honeypot accessible from its corporate network or via one of its cloud proxies.

across reboots. This infrastructure also supports centralized monitoring and logging of interactions across deployments.

Proxies. Since the physical PLC cannot be deployed in the cloud, proxies are used to forward requests from the cloud network to the PLC. This setup creates the illusion of direct communication with the PLC. For consistency, the same approach is also used for the virtualized honeypots, avoiding the need to deploy the honeypots in the cloud. As shown in Figure 3, honeypots can be accessed either directly via their corporate public IPs or indirectly through a proxy. The proxies are implemented as lightweight (t2.micro) AWS compute instances running Ubuntu 22.04, with packet forwarding configured using iptables [32]. Each proxy has a dedicated Elastic IP to ensure stable public access across reboots.

Packet Capture and Storage. All traffic to and from the honeypots on ports 80 (HTTP), 102 (S7Comm), and 502 (Modbus) is captured as PCAP files using tcpdump [47]. The packet capture setup varies depending on the network environment. In the corporate network, the packet capture leverages Forescout's infrastructure, which includes a pre-configured mirroring system. All traffic is mirrored to an external analysis system and stored in an AWS S3 bucket. For AWS deployments, a challenge arises as requests are proxied, so honeypots log correct payloads but proxy IPs. To address this, we capture traffic on the proxies. Tcpdump filters ensure proxy traffic is excluded from honeypot captures and vice versa, so each system records only its own interactions. PCAP files are saved on separate virtual drives for each honeypot and proxy to protect against system failures. File sizes are limited to 100 MB to reduce data loss in case of corruption.

Table 3: Observed requests per honeypot

Honeypot	Network	Region	HTTP	Modbus	S7Comm
Conpot	Corporate	EU	27032	1172	1336
HoneyICS1	Corporate	EU	26307	776	1682
HoneyICS2	Corporate	EU	25829	643	1632
Physical PLC	Corporate	EU	15329	695	1658
Conpot	Cloud	EU	123332	403	761
HoneyICS1	Cloud	EU	119915	453	845
HoneyICS2	Cloud	EU	117983	460	904
Physical PLC	Cloud	EU	60915	411	901
Conpot	Cloud	US	73377	415	811
HoneyICS1	Cloud	US	77890	429	875
HoneyICS2	Cloud	US	84779	475	920
Physical PLC	Cloud	US	77429	441	889
Conpot	Cloud	Asia	87613	450	810
HoneyICS1	Cloud	Asia	105403	466	979
HoneyICS2	Cloud	Asia	84149	425	961
Physical PLC	Cloud	Asia	67718	436	841
Total			1175000	8550	16805

4.2.2. Dataset

Data collection took place over three months, between December 21, 2024, and March 21, 2025. In this period, all incoming and outgoing packets for each honeypot are captured and stored as PCAP files. This format allows us to extract all relevant traffic details, such as source and destination addresses, protocols, and payload content (e.g., Modbus function codes). It is worth noting that not all captured traffic originated from interactions with the honeypots. Some packets were generated by background processes on the honeypot and proxy hosts, such as DNS queries and automatic operating system updates. To avoid introducing bias into our analysis, we filtered out such packets, as well as traffic generated from IPs under our control, prior to conducting further processing. Table 3 summarizes the distribution of HTTP, Modbus, and S7Comm requests across honeypots, with further details on Modbus and S7Comm function codes provided in Tables 4 and 5, respectively.

4.3. Data Analysis

To address our research questions, we use both quantitative and qualitative analyses. RQ1 to RQ3 are examined using quantitative methods to assess the attractiveness of honeypot configurations. RQ4 and RQ5 are explored through qualitative analysis, focusing on the nature and intent of interactions to understand malicious behavior and tool usage. The methods and metrics used are detailed in the following subsections.

4.3.1. Quantitative Analysis

To assess the attractiveness of different honeypot configurations, we characterize and compare interactions across different honeypot deployments. To this

Table 4: Observed Modbus requests

Modbus Function Code	Count
01 (Read Coils)	44
02 (Read Discrete Inputs)	17
03 (Read Holding Registers)	1131
04 (Read Input Registers)	391
05 (Write Single Coil)	4
17 (Report Slave ID)	938
43 (Report Device Information)	5995
0 (Unknown function)	26
66 (Unknown function)	2
100 (Unknown function)	2
Total	8550

Table 5: Observed S7Comm requests

S7Comm Function	Count
Setup Communication	5721
[Read SZL] ID=0x0011	5652
[Read SZL] ID=0x001c	5402
[Read SZL] ID=0x011c	21
[Read SZL] ID=0x0424	4
[Read SZL] (no ID specified)	2
[Message Service] SubscribedEvents=(MODE)	3
Total	16805

end, we first introduce the notion of interaction and define the metrics for assessing honeypot attractiveness.

Interactions. We adapt the definition of interaction proposed in [27]: a sequence of requests that (1) originate from the same IP, (2) target the same destination IP and port, and (3) occur within a specified maximum time interval.[2] Following [27], we analyze inter-request intervals to define appropriate time thresholds, considering potential delays introduced by the use of proxies, particularly in HTTP traffic. For HTTP, we select a 30-second threshold, which captures 95% of HTTP follow-up requests (from the same IP). For S7Comm and Modbus, we inspect packet sequences, ensuring continuity of TCP connections. Based on this analysis, we select a 15-second threshold across all deployments for both S7Comm and Modbus, which captures 99.96% and 100% of subsequent requests, respectively.

Attractiveness. We assess attractiveness by interaction volume and complexity. The complexity is measured as the number of requests per interaction, providing an indication of their sophistication. A honeypot is deemed more attractive if it receives a larger number of interactions with higher complexity.

Analysis Methods. We aggregate the data according to experiment conditions to assess attractiveness across different interaction levels (RQ1), network types (RQ2), and geographic regions (RQ3), analyzing interaction volume and complexity in

[2]Note that, differently from [27], we exclude the source port. This is to account for tools that open multiple TCP connections for a single logical task.

each case. To address RQ1, we aggregate interactions by honeypot type (Conpot, HoneyICS1, HoneyICS2, and the physical PLC) across all deployment environments (corporate network, AWS EU, AWS US, and AWS Asia). To address RQ2, we separately aggregate interactions from the honeypots on the corporate network and on the AWS EU cloud. To address RQ3, we focus on cloud deployments and aggregate interactions for each region (AWS EU, AWS US, AWS Asia). Additionally, we analyze the geographic origin of IP addresses interacting with the honeypots using GreyNoise [16] and, where necessary, IP-API [19]. Grouping IP origins by continent helps assess whether the deployment region influences the geographic distribution of interactions.

4.3.2. Qualitative Analysis

While the quantitative analysis is meant to reveal which honeypot configurations attract higher volumes and more complex interactions, it does not capture the underlying intent of these interactions. To gain deeper insights, we complement this with a qualitative analysis focused on the tools employed, the actors involved, and how interaction patterns vary across different ICS honeypot configurations. To this end, we fingerprint recurrent interaction patterns and attribute interaction patterns to known ICS tools, as described in Section 4.3.3.

To address RQ4, we apply the identified fingerprints to determine how frequently ICS tools are used across honeypot configurations. By analyzing the corresponding interaction patterns and cross-referencing IP addresses with GreyNoise, we infer the likely intent behind each tool and associate them with known actors. To address RQ5, we analyze interactions that are not matched by any fingerprint. These interactions are manually inspected to infer intent, identify potential manual activity, and, where possible, associate them with known actors using GreyNoise.

4.3.3. ICS Interaction Fingerprinting

To gain deeper insights into the collected ICS interactions, we identify and fingerprint recurring patterns in Modbus and S7Comm traffic. The fingerprinting process, illustrated in Figure 4, follows a structured and iterative methodology. A pattern is considered recurrent if it appears at least five times across all honeypot deployments. This allows us to assess the occurrence of similar interactions across multiple IPs, deployment environments, and regions. These interactions are analyzed based on protocol-specific characteristics and associated with specific tools or actors when possible. We repeat the process for the remaining unmatched traffic until no additional patterns can be identified. Next, we provide an overview of the fingerprinting process for Modbus and S7Comm.

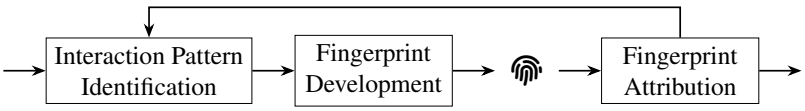


Figure 4: Fingerprinting process for ICS tools.

Table 6: Example of honeypot-specific patterns

Honeypot	**Packet**	**TransID**	**Unit ID**	**Modbus Function**
Conpot	1	$x \in [0, 65535]$	1	43/1
	2	$x+1$	1	43/1
	3	$x+2$	1	43/1
	4	$x+3$	1	17
HoneyICS	1	$x \in [0, 65535]$	1	43/1
	2	$x+1$	1	17
Physical PLC	1	$x \in [0, 65535]$	1	43/1

Modbus. We identify 17 distinct interaction patterns in Modbus traffic, based on features such as TransIDs, UnitIDs, Function Code, and sequences of Modbus requests. For example, while the TransID field may take any value between 0 and 65535, we observe several requests using Function Code 43 (Read Device Identification) with Unit ID 0 and a consistent TransID of 23111. This regularity suggests the use of a specific tool.[3] Among the identified patterns, two exhibit port scanning behavior, targeting not only Modbus but also additional services such as Portmap and HTTP. Some patterns are honeypot-specific, due to differences in Modbus function support across honeypot implementations (see Table 2). Table 6 shows an example of these patterns, all originating from IPs associated with the same actor. For Conpot, the tool sends three subsequent device identification requests, each with an incremented TransID, followed by a slave ID request, again with an incremented TransID. For HoneyICS, the sequence is shorter, consisting of a single device identification request followed by a slave ID request. In the case of the physical PLC, only the device identification is requested.

To reduce fragmentation and ensure consistency in analysis, we apply grouping criteria to consolidate related patterns into broader fingerprints. We merge inter-

[3]An inspection of the Zgrab2 source code revealed that its default Request ID is the 16-bit unsigned integer `0x5A47`, which corresponds to 23111 in decimal. Moreover, executing Zgrab2 reproduces interactions identical to those observed, allowing us to confidently attribute this pattern to Zgrab2.

action patterns with similar characteristics into a single fingerprint when one of the following conditions is met: *(i)* the interactions originate from a single actor identified by GreyNoise (in which case the fingerprint is named after the actor); *(ii)* the interactions originate from IPs associated with the same ISP (in which case the fingerprint is named after the ISP). Applying these criteria, we consolidate the 17 interaction patterns into 12 distinct fingerprints.

To attribute the resulting fingerprints to specific tools or actors, we compare our fingerprints against those identified in [3], which systematically label recurring Modbus interactions occurring in the dataset collected in [26] and associate them with known tools and actors whenever possible. This comparison allows us to match six of our fingerprints with previously identified fingerprints. In particular, two fingerprints correspond to `Zgrab2` [51] and `Nmap modbus-discover.nse` [33]. The remaining four fingerprints were not associated with previously documented tools or actors and are therefore labeled Tool A–D. These fingerprints resemble Zgrab2-style device identification probing behaviors that share the same Modbus operation (Function Code 43) but differ in observable request parameters, most notably the Transaction Identifier (TransID) and Unit Identifier (UnitID). For instance, Tools A-C can be differentiated by TransID. On the other hand, the UnitID distinguishes Tool D from Tools A–C: Tool D targets UnitID 1 while the others target UnitID 0.

For the remaining fingerprints, we use GreyNoise to investigate the origin of the associated IP addresses. When a fingerprint consistently originates from IPs linked to a known actor in GreyNoise, we attribute it accordingly. This is the case for Tools Bitsight, Driftnet, Crowdstrike, and Ningxia. Bitsight follows a structured probing sequence, while Crowdstrike performs a sequence with fixed TransID and incrementing UnitIDs. Ningxia conducts a two-request interaction targeting specific UnitIDs. Driftnet performs device identification and slave ID queries with environment-dependent length; its interaction with physical PLCs is indistinguishable from Tool D based on packet content alone, and attribution therefore relies on source IP. In cases where the IPs belong to the same ISP but cannot be linked to a specific actor or tool, we label the fingerprint after the ISP. For instance, one port scan originated from BinaryEdge.io, while another was observed exclusively from Ucloud IPs. The BinaryEdge.io scan performs multi-protocol probing on port 502, including Modbus Function Code 1 requests embedded within a broader service discovery sequence. The UCLOUD scan exhibits structured Modbus probing with reused Transaction Identifiers, with the number and combination of requests varying by honeypot interaction level. An overview of the 12 fingerprints and their associated tools or actors is provided in

Table 7: Modbus fingerprints

Tool	Operation
`Zgrab2` [51]	Read Device Identification
`Nmap modbus-discover.nse` [33]	Read Device Identification and Report Slave ID
`Tool A`	Read Device Identification
`Tool B`	Read Device Identification
`Tool C`	Read Device Identification
`Tool D`	Read Device Identification
`Tool Bitsight` [7]	Read Device Identification
`Tool Driftnet` [14]	Read Device Identification and Report Slave ID
`Tool Crowdstrike` [12]	Read Holding Registers
`Tool Ningxia`	Read Device Identification
`Port Scan BinaryEdge.io` [6]	Read Coils
`Port Scan Ucloud`	Read Device Identification and Read Coils and Unknown Function Request

Table 8: Example of S7Comm fingerprints

Fingerprint	**Packet**	**Function**	**SZL ID**	**SZL Index**
`Zgrab2`	1	Setup communication	–	—
	2	Read SZL	0x0011	0x0001
	3	Read SZL	0x001c	0x0001
`Nmap s7-info.nse`	1	Setup communication	–	—
	2	Read SZL	0x0011	0x0001
	3	Read SZL	0x0011	0x0001
	4	Read SZL	0x001c	0x0001

Table 7. Their complete definition is presented in Appendix A (Table A.18).

S7Comm. We identify five recurrent patterns in the observed S7Comm interactions. Each interaction begins with a setup communication request, which initializes the S7Comm session and is common to all observed interactions. Once the connection is established, clients issue various requests, including Read SZL (System-Zustands-Listen) operations that retrieve diagnostic or configuration information based on specific list IDs and indexes. To enable fingerprinting, we analyze these requests by extracting distinctive features such as the targeted list IDs, associated indexes, and the sequence and frequency of request types. Table 8 provides two examples of the fingerprints we extracted. The first, attributed to `Zgrab2`, comprises three S7Comm messages: a setup communication request followed by Read SZL

Table 9: Keywords for searching S7Comm tools

List 1:	[“s7comm”, “s7comm-plus”, “siemens s7”, “s7 protocol”, “s7 PLC”, “s7”]
List 2:	[“”, “client”, “tool”, “scan”, “exploit”, “vulnerability”, “script”, “discover”, “test”, “penetration”, “sniffer”, “fuzzer”, “hack”, “attack”, “security”, “info”, “banner grabbing”]

Table 10: Inclusion criteria for S7Comm tool

A tool must …
1. Focus on Siemens S7 PLCs.
2. Be executable (with minimal modifications).
3. Use S7Comm to communicate with PLCs.
4. Be downloaded, forked or starred at least 10 times.

requests for IDs 0x0011 and 0x001c. The second, linked to `Nmap`'s `s7-info.nse` script, exhibits a similar pattern but includes an additional Read SZL request for ID 0x0011.

To assess whether the identified S7Comm fingerprints correspond to known tools, we search multiple online databases. GitHub serves as a primary repository for custom-developed tools, while ExploitDB [34], 0day.today [1], and Packet Storm Security [37] are recognized sources for publicly available exploits. To capture tools that may be hosted elsewhere, we also include the first page of results from DuckDuckGo for each query. This multi-source strategy increases the likelihood of discovering relevant tools. We build search queries using a two-list approach (see Table 9): the first list contains terms related to the Siemens S7Comm protocol, and the second includes common keywords associated with tool functionality. By combining one term from each list (e.g., “s7comm scan”), we generate 102 search queries and apply them across the selected sources. We filter the results based on the inclusion criteria defined in Table 10. These criteria help us filter relevant tools, ensuring the inclusion of widely used and functional tools that actors are likely to employ. Specifically, we exclude tools that do not use the S7Comm protocol or require extensive bug fixing.

After applying the selection criteria outlined in Table 10, we identify six S7Comm tools. By inspecting their behavior and source code, we are able to associate four of these tools with the extracted fingerprints. One fingerprint could not be linked to any of the identified tools. However, based on our data and information retrieved from GreyNoise, we observe that this pattern was only associated with an actor identified as ONYPHE [36]. Accordingly, we attribute the fingerprint to

Table 11: S7Comm fingerprints

Tool	Description
`Zgrab2` [51]	Retrieves module and component identification.
`PLC Device Scanner` [15]	Retrieves module and component identification.
`Nmap s7-info.nse` [18]	Retrieves module and component identification.
`s7comm_scan.py` [52]	Retrieves module identification.
`Tool ONYPHE` [36]	Setup S7Comm connection.

Table 12: Interaction Level: volume and complexity

Honeypot	Protocol	Min	Q1	Median	Q3	Max	Count
Conpot	HTTP	1	1	1	2	19994	42321
	S7Comm	1	3	3	3	8	1318
	Modbus	1	1	1	1	241	1487
HoneyICS1	HTTP	1	1	1	2	12534	40408
	S7Comm	1	3	3	3	16	1535
	Modbus	1	1	1	1	10	1393
HoneyICS2	HTTP	1	1	1	2	8021	41870
	S7Comm	1	3	3	3	16	1529
	Modbus	1	1	1	1	16	1348
Physical PLC	HTTP	1	1	1	2	7365	38381
	S7Comm	1	3	3	3	4	1484
	Modbus	1	1	1	1	10	1314

this actor. An overview of the discovered fingerprints and their associated tools or actors is provided in Table 11.

5. Results

This section presents the results of our study, structured around the research questions. We refer to Appendix B for a detailed overview.

5.1. Quantitative Analysis

RQ1: To what extent does the level of interaction supported by an ICS honeypot influence the attractiveness of the honeypot? Table 12 reports the number of interactions (Count) and their complexity (Min, Q1, Median, Q3, and Max) for each protocol across different levels of interaction (Conpot, HoneyICS, and physical PLC). The table shows notable differences between protocols. We now discuss the results for each protocol.

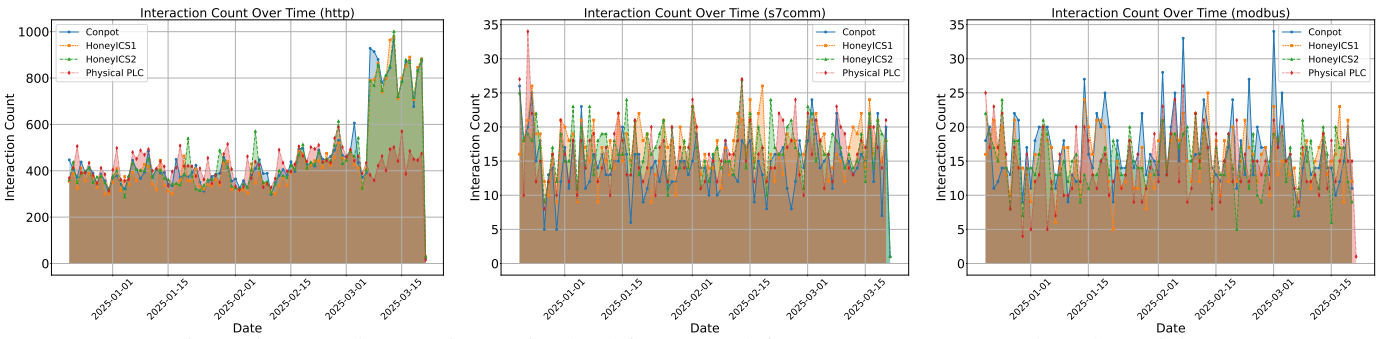


Figure 5: Interactions per interaction level for HTTP (left), S7Comm (center), and Modbus (right)

HTTP. Figure 5 (left) shows the number of HTTP interactions per day for each interaction level. For a large portion of the data collection period, we observe no notable differences between the honeypots. However, we observe a sharp increase in March for Conpot and both HoneyICS instances, while the physical PLC maintains a stable trend. This shift may be attributed to differences in the web server technologies used by the honeypots. Conpot and HoneyICS both employ Lighttpd, whereas the physical PLC uses a jQuery-based server. A detailed analysis shows that a single IP address is responsible for the increase (on both Conpot and HoneyICS), repeatedly issuing HTTP GET requests to `/login.cgi/cgi_main.cgi` with admin/admin credentials. The Lighttpd servers generate responses that appear to encourage repeated attempts, while the jQuery-based server returns responses that did not trigger further activity. This suggests that the observed increase in interactions is primarily driven by server behavior rather than by the honeypot's interaction level. Considering the overall number of interactions (see Table 12), the physical PLC receives the fewest interactions, followed by HoneyICS, while Conpot receives the most. Even among honeypots with the same interaction level (HoneyICS1 vs. HoneyICS2), we observe a noticeable variation in the number of interactions, further suggesting that interaction level alone does not determine HTTP attractiveness. Moreover, Table 12 shows that the Min, Q1, Median, and Q3 values are identical across honeypots, indicating that at least 75% of interactions have similar complexity regardless of the honeypot's interaction level. The main difference lies in the maximum complexity: Conpot exhibits the most complex interaction, followed by HoneyICS and then the physical PLC. Again, notable variation between the two HoneyICS instances reinforces the conclusion that the interaction level does not consistently influence the attractiveness or complexity of HTTP interactions in ICS honeypots.

S7Comm. Figure 5 (center) reports the S7Comm interactions per day for each interaction level. The curves are closely aligned, indicating minor differences in interac-

tion frequency across levels. Conpot consistently receives the fewest interactions. HoneyICS shows the highest activity, with its two instances differing by only six interactions (Table 12). The physical PLC follows, with approximately 50 fewer interactions (3%), indicating a comparable level of engagement. The distribution of interaction complexity is nearly identical across all honeypots (Table 12). Outliers are scarce for Conpot and both HoneyICS instances, while absent for the physical PLC.

Modbus. Figure 5 (right) reports the Modbus interactions per day for each interaction level. All interaction levels show similar daily trends, but Conpot exhibits the most frequent and highest peaks, while HoneyICS and the physical PLC follow a comparable but lower pattern. This trend is consistent with the total interaction counts shown in Table 12, where Conpot receives approximately 100 more interactions (7%) than HoneyICS and about 170 more than the physical PLC (12%). These findings suggest that Conpot attracts more Modbus interactions compared to the higher-interaction counterparts. Across all interaction levels, most Modbus interactions have a complexity of 1. For HoneyICS and the physical PLC, the few interactions with higher complexity comprise 6 and 10 requests, while for Conpot, they mostly comprise 2 or 3 requests (Table 12). An in-depth analysis shows that this difference in complexity is influenced by how Conpot responds to a tool used by Crowdstrike. This suggests that interactions with low-interaction honeypots tend to be less complex than those with high-interaction honeypots or physical PLCs. However, Conpot was involved in one interaction comprising 241 requests, while they were at most 10 and 16 for HoneyICS and 10 for the physical PLC. This suggests that while Conpot typically attracts simpler interactions, it can also elicit highly complex behavior.

Answer to RQ1. The interaction level does not consistently determine the attractiveness of an ICS honeypot. For HTTP, server configuration had a greater impact on interaction patterns than interaction level. In the case of S7Comm, HoneyICS was slightly more attractive, while for Modbus, Conpot drew more interactions. Overall, attractiveness appears to depend more on protocol-specific factors than on interaction level alone.

RQ2: To what extent does the network type on which an ICS honeypot is deployed influence the attractiveness of the honeypot? Table 13 reports the number of interactions (Count) and their complexity (Min, Q1, Median, Q3, and Max) for each protocol across network types (corporate vs. cloud). We now discuss the results for each protocol.

HTTP. Figure 6 (left) shows the HTTP interactions per day for both the corporate

Table 13: Network Type: volume and complexity

Network	Protocol	Min	Q1	Median	Q3	Max	Count
Corporate	HTTP	1	1	1	2	8021	30700
	S7Comm	1	3	3	3	7	2258
	Modbus	1	1	1	1	241	1585
Cloud	HTTP	1	1	1	2	7269	44365
	S7Comm	1	3	3	3	8	1183
	Modbus	1	1	1	1	10	1313

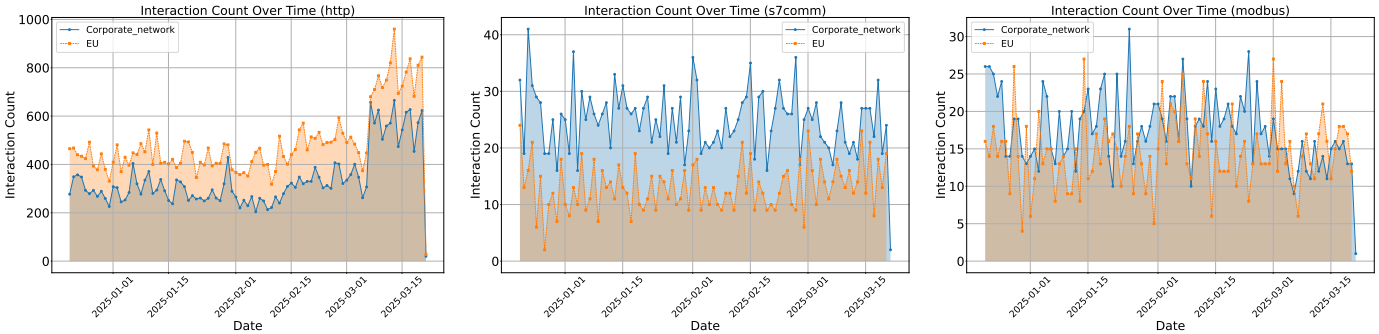


Figure 6: Interactions per network type for HTTP (left), S7Comm (center), and Modbus (right)

network and the cloud network (AWS EU). Across the entire observation period, the cloud network consistently receives more HTTP interactions than the corporate network (45%), suggesting that it is more attractive for HTTP-based activity. The distribution of HTTP interaction complexity is largely consistent across both networks, with only minor differences. As shown in Table 13, the Min, Q1, Median, and Q3 values are identical, indicating similar interaction patterns regardless of network type. The only notable distinction is in the maximum observed complexity, which is approximately 750 higher on the corporate network. This suggests that while HTTP interactions are largely comparable across networks, the corporate network occasionally attracts more complex requests.

S7Comm. Figure 6 (center) reports the S7Comm interactions per day for both the corporate and cloud networks. Over the entire observation period, the corporate network consistently receives more S7Comm interactions than the cloud network. This trend is confirmed by Table 13, which shows that the corporate network receives approximately twice as many S7Comm interactions (91%), suggesting it is more attractive for this protocol. The overall complexity is almost constant in the two networks, which shows a maximum size of interaction of 7 and 8 for the corporate and cloud networks, respectively.

Modbus. Figure 6 (right) shows the Modbus interactions per day for the corporate

Table 14: Geographic Region: volume and complexity

Location	Protocol	Min	Q1	Median	Q3	Max	Count
EU	HTTP	1	1	1	2	7269	44365
	S7Comm	1	3	3	3	8	1183
	Modbus	1	1	1	1	10	1313
US	HTTP	1	1	1	2	19994	41386
	S7Comm	1	3	3	3	16	1182
	Modbus	1	1	1	1	16	1303
Asia	HTTP	1	1	1	2	12534	46529
	S7Comm	1	3	3	3	16	1243
	Modbus	1	1	1	1	10	1341

and cloud networks. While both exhibit fluctuations, the corporate network consistently receives more interactions. As shown in Table 13, the corporate network receives about 270 more Modbus interactions (a 21% increase), suggesting it is a more attractive target for this protocol. Table 13 shows that the overall distribution is largely the same on both networks, with most interactions having a complexity of 1. The difference in maximum complexity is due to a single outlier from Conpot on the corporate network, as noted in the RQ1 analysis. The maximum complexity for all other honeypots on the corporate network is 10, matching that of the cloud network. This suggests that, overall, interaction complexity for Modbus does not meaningfully differ between network types.

Answer to RQ2. Network type affects the volume of interactions for specific protocols but not their overall complexity. HTTP traffic is more frequent in the cloud environment, whereas S7Comm and Modbus interactions are more common on the corporate network. This suggests that while network type influences protocol-specific attractiveness, it has a limited impact on the sophistication of interactions.

RQ3: To what extent does the geographic location of an ICS honeypot influence its attractiveness? Table 13 reports the number of interactions (Count) and their complexity (Min, Q1, Median, Q3, and Max) for each protocol across regions (Europe, North America, and Asia). We now discuss the results for each protocol, along with the origin of iterations per region.

HTTP. Figure 7 (left) reports the HTTP interactions per day across different regions. The trends are relatively similar across regions, although Asia exhibits the highest peaks, followed by the EU and then the US. Table 14 confirms this pattern, showing that Asia receives over 2000 more HTTP interactions than the EU (5%) and over 5000 more than the US (12%). This suggests that Asia is the most attractive region

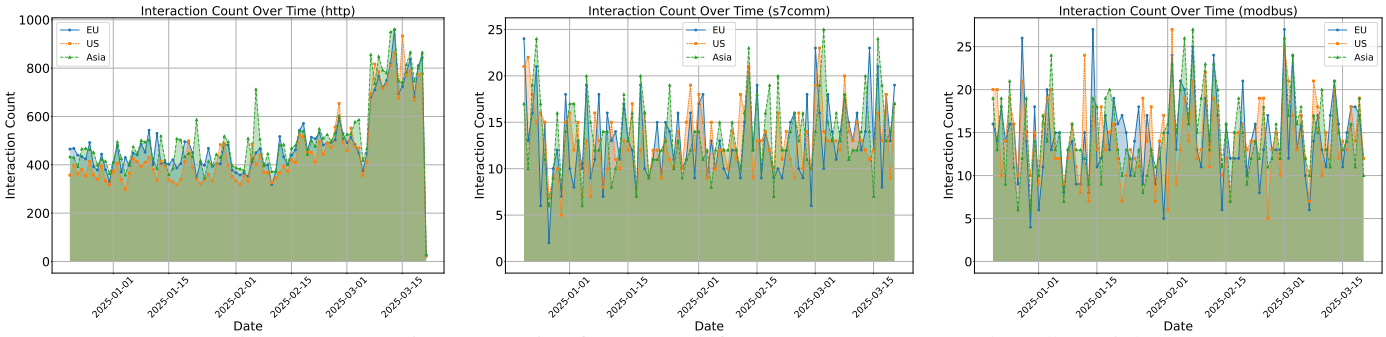


Figure 7: Interactions per region for HTTP (left), S7Comm (center), and Modbus (right)

for HTTP. The complexity is consistent across all regions, with identical Min, Q1, Median, and Q3 values, as shown in Table 14. The only notable variation lies in the maximum values: the US shows the highest (19,994), followed by Asia (12,534) and the EU (7,269). These, however, appear to be outliers; most complex interactions for the US and Asia typically fall between 7,100 and 7,400. Overall, these findings suggest that HTTP interaction complexity is not strongly influenced by geographic region.

S7Comm. Figure 7 (center) reports the S7Comm interactions per day across different regions. The trends are relatively similar, with no substantial differences in the number of S7Comm interactions. As shown in Table 14, Asia receives approximately 60 more interactions than both the EU and the US (5%), while the difference between the EU and US is minimal, just one interaction. This may indicate that Asia is marginally more attractive for S7Comm, though the variation is minor. The distribution of interaction complexity is nearly identical across all regions, as shown in Table 14. The only notable difference is in the maximum complexity, with both the US and Asia showing interaction values roughly twice as complex as those observed in the EU.

Modbus. Figure 7 (right) reports the Modbus interactions per day across different regions. The trends are relatively similar across all regions, with no substantial differences observed. Table 14 supports this observation, indicating that Asia receives approximately 30 more Modbus interactions than the EU (2%) and 40 more than the US (3%). Similarly, the distribution of interaction complexity is nearly identical across regions as shown in Table 14. The only notable deviation is the slightly higher maximum complexity observed in the US compared to the other regions.

Origin of Interactions. Tables 15, 16, and 17 present the geographic origin of interactions for each protocol, with bold values indicating the regions where the honeypots were deployed. Across all protocols, we observe only minor differences

Table 15: Origin of HTTP interactions

Region	Africa	Asia	Europe	North America	Oceania	South America	Total
EU	86 (0.19%)	8045 (18.13%)	**24201 (54.55%)**	11424 (25.75%)	170 (0.38%)	439 (0.99%)	44365 (100%)
US	84 (0.20%)	5868 (14.18%)	23403 (56.55%)	**11455 (27.68%)**	154 (0.37%)	422 (1.02%)	41386 (100%)
Asia	313 (0.67%)	**8852 (19.02%)**	24539 (52.74%)	12185 (26.19%)	169 (0.36%)	471 (1.01%)	46529 (100%)
Total	483 (0.37%)	22765 (17.21%)	72143 (54.54%)	35064 (26.51%)	493 (0.37%)	1332 (1.01%)	132280 (100%)

Table 16: Origin of S7Comm interactions

Region	Africa	Asia	Europe	North America	Oceania	South America	Total
EU	0 (0.00%)	92 (7.78%)	**227 (19.19%)**	817 (69.06%)	0 (0.00%)	47 (3.97%)	1183 (100%)
US	0 (0.00%)	115 (9.73%)	183 (15.48%)	**817 (69.12%)**	0 (0.00%)	67 (5.67%)	1182 (100%)
Asia	0 (0.00%)	**136 (10.94%)**	214 (17.22%)	847 (68.14%)	0 (0.00%)	46 (3.70%)	1243 (100%)
Total	0 (0.00%)	343 (9.51%)	624 (17.29%)	2481 (68.76%)	0 (0.00%)	160 (4.43%)	3608 (100%)

Table 17: Origin of Modbus interactions

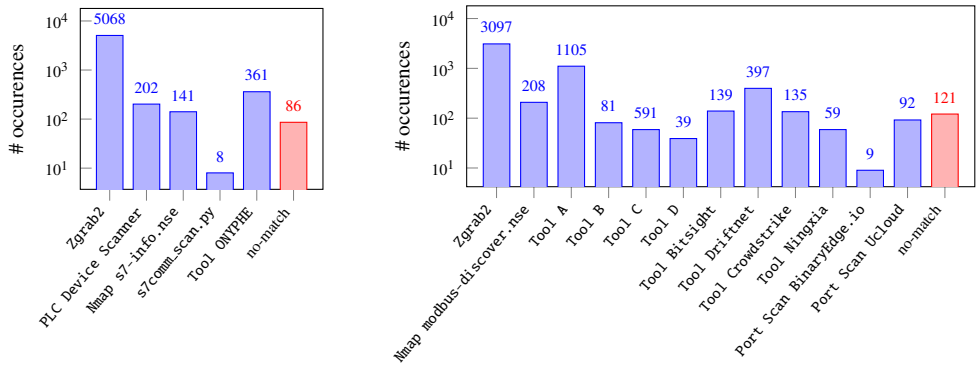
Region	Africa	Asia	Europe	North America	Oceania	South America	Total
EU	2 (0.15%)	163 (12.41%)	**218 (16.60%)**	887 (67.56%)	0 (0.00%)	43 (3.27%)	1313 (100%)
US	3 (0.23%)	147 (11.28%)	200 (15.35%)	**903 (69.30%)**	0 (0.00%)	50 (3.84%)	1303 (100%)
Asia	5 (0.37%)	**162 (12.08%)**	229 (17.08%)	900 (67.11%)	0 (0.00%)	45 (3.36%)	1341 (100%)
Total	10 (0.25%)	472 (11.93%)	647 (16.35%)	2690 (67.98%)	0 (0.00%)	138 (3.49%)	3957 (100%)

in the distribution of interaction origins based on deployment region. While the region in which a honeypot is hosted tends to show a slightly higher share of interactions for that protocol, the difference is minimal, never exceeding 2.7%. These results suggest that geographic deployment has limited influence on the origin of incoming interactions, indicating that honeypot attractiveness is largely independent of deployment region.

Answer to RQ3. Geographic location has limited impact on honeypot attractiveness. For each protocol, interaction count and complexity are mostly consistent across regions, with only minor differences. The origin of interacting IPs also appears unaffected by deployment region, suggesting ICS honeypots are not systematically targeted based on location.

5.2. Qualitative Analysis

RQ4: ***Which tools are commonly used for ICS interactions, and what actions are associated with them?*** Our analysis shows that a large portion of the observed ICS



(a) S7Comm

(b) Modbus

Figure 8: Interactions (no)matching fingerprints for S7Comm and Modbus.

traffic is automated through tool usage. Based on aggregate data from our deployments (see Tables B.20 and B.21 in Appendix B.2), 98.5% of S7Comm and 97.8% of Modbus interactions were associated with the identified ICS tool fingerprints (cf. Tables 7 and 11). Figure 8 illustrates the distribution of the interactions matching these fingerprints across both S7Comm and Modbus protocols. Interestingly, `ZGrab2` emerges as the most frequently observed tool for both protocols. We also noticed that Modbus interactions involved a more diverse set of tools compared to S7Comm. This likely reflects Modbus's nature as an open protocol that has been deployed across many different vendors and industries for decades, whereas S7Comm is proprietary to Siemens and used mainly within their ecosystem.

A closer examination of S7Comm interactions reveals particularly high fingerprinting coverage, typically exceeding 96% and often reaching over 99% across deployments (Table B.20). As shown in Figure 8a, `Zgrab2`, which is mainly used to gather module and component information, accounted for the largest volume of fingerprinted interactions. This pattern was consistent across all honeypot types and geographic locations. The second most common tool is `Tool ONYPHE`, which establishes S7Comm connections and appears to be operated by a specific, identifiable actor rather than being a widely-used public tool. Interestingly, `Tool ONYPHE`'s activity varied between different honeypots and regions, suggesting potentially targeted scanning practices. Other tools included `PLC Device Scanner` and the `Nmap s7-info.nse` script, both aimed at device identification but observed less frequently; the latter appeared slightly more often in the corporate network. The `s7comm_scan.py` script was seen only sporadically. Overall, these results suggest

that S7Comm interactions are predominantly reconnaissance-oriented, aimed at discovering devices and understanding their capabilities.

In the case of Modbus interactions, our fingerprints achieved high coverage across most deployments. However, Conpot instances consistently showed slightly lower coverage (e.g., 90.8% in the corporate network) compared to HoneyICS and physical PLC deployments, which often reached nearly complete coverage in cloud environments (Table B.21). As with S7Comm, `Zgrab2` was the most frequently observed tool (Figure 8b), primarily issuing *Read Device Identification* requests across all honeypot configurations. The second most prevalent was `Tool A`, particularly observed in cloud deployments. Notably, several fingerprints were linked to known commercial security vendors and scanning services (Table 7). `Tool Bitsight`, `Tool Driftnet`, `Tool Crowdstrike`, and `Tool Ningxia` were regularly observed performing device identification. Port scanning activities associated with `BinaryEdge.io` primarily targeted Conpot, while `Port Scan Ucloud` (which performed a mix of device identification, coil reads, and unknown function requests) was observed more broadly but less frequently. The `Nmap modbus-discover.nse` script was most prominent in the corporate network, with minimal presence in cloud environments, again suggesting potential targeted practices. Several other tools (`Tools B`, `C` and `D`) associated with device identification were observed occasionally. Modbus interactions, therefore, are also predominantly characterized by device identification attempts, originating from both general-purpose scanners and actor-specific tools.

Answer to RQ4. Observed ICS interactions are dominated by automated reconnaissance activities, primarily scanning and device identification. `ZGrab2` emerges as the most frequently used tool across both S7Comm and Modbus traffic. While S7Comm interactions also involve actor-specific tools such as `Tool ONYPHE`, Modbus traffic displays greater tool diversity—likely due to its open specification and broader industrial adoption—including general-purpose scanners, tools from security vendors (e.g., `Tool Bitsight`, `Tool Driftnet`), and targeted port scans. These results indicate that observed activity largely centers on information gathering across both protocols rather than exploitation.

RQ5: ***What patterns or characteristics are observed from ICS interactions that cannot be identified as common tools or exploits?*** The previous analysis revealed that the vast majority of ICS traffic is generated by automated tools, primarily performing routine reconnaissance such as scanning and device identification. While these interactions dominate the dataset, filtering them out allows us to

isolate a smaller set of interactions that deviate from typical automated behavior. Specifically, 86 S7Comm interactions (1.5% of the overall S7Comm interactions) and 121 Modbus interactions (2.2% of the overall Modbus interactions) could not be matched to the identified tool fingerprints. These unmatched interactions appeared across all honeypot deployments, with a slightly higher occurrence in the corporate network. These remaining cases deserve additional investigation, as they may reflect more sophisticated, targeted, or manually driven activities. Below, we discuss the most representative examples.

Targeted Reconnaissance across Deployments. A notable case involved an IP address from China that, at the time of analysis, was not flagged by standard threat intelligence platforms. While the activity aligns with the behavior of `s7comm_scan.py`, it exhibited an unusual level of sophistication. The campaign unfolded in distinct phases across several days and different network environments, which points toward a targeted intelligence-gathering campaign rather than typical opportunistic scanning patterns. The sequence began with a port scan, conducted about one hour prior to ICS-specific reconnaissance, followed by targeted S7Comm scans focused on specific honeypot deployments (Conpot and HoneyICS2) across different cloud regions. Two days later, the attackers expanded their scope to include both HoneyICS1 and HoneyICS2 within the corporate network, revealing either evolving target selection criteria or a methodical expansion of reconnaissance scope. The evolution of this attack campaign raises important questions about whether the actor possessed prior knowledge of the infrastructure or was systematically mapping specific ICS deployments throughout the IPv4 address space.

Metasploit-Based Multi-Stage Cross-Protocol Attack with Human-Driven Interaction. The most complex attack sequence we observed was a three-day campaign that unfolded over five calendar days originating from three residential IP addresses located in different cities in Iran. This coordinated, multi-stage attack, primarily leveraged Metasploit modules[4] and systematically targeted our Conpot instance deployed in the corporate network. The main steps are summarized in Figure 9.

The campaign began with initial reconnaissance on the first day, followed by intensive exploitation activities on day three. These activities included various

[4]We attributed the activity to Metasploit modules based on fingerprints reported in [3]. These modules were not fingerprinted in our study because their interaction patterns occurred less than five times across all honeypot deployments during the data collection period, and thus did not meet our threshold for recurrent pattern analysis.

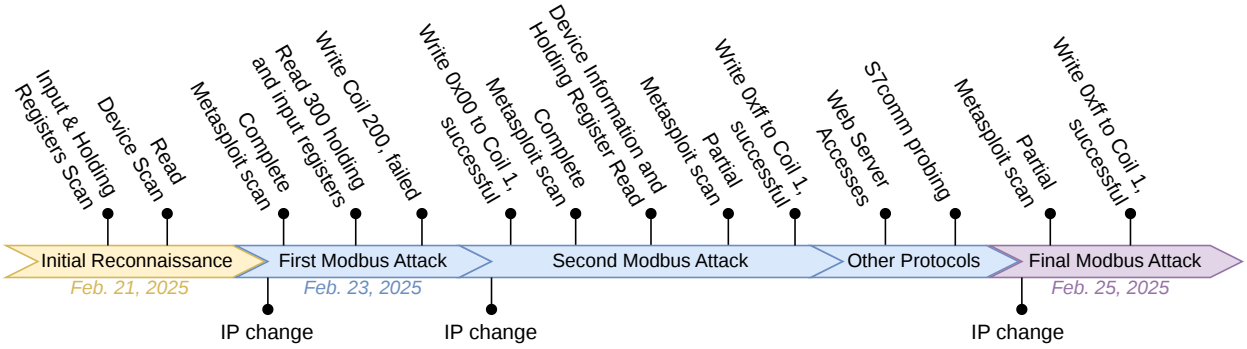


Figure 9: Main steps of a manual attack received in the Conpot instance.

exhaustive scans that methodically probed every possible device identifier (all 254 UnitIDs) in the Modbus address space to detect active devices, together with various failed and successful writing operations. The final stage occurred on day five, where the attacker used a fresh IP address to repeat the device identification scan, although this last scan was interrupted after only 32 UnitIDs, and after which the attacker executed a final coil write operation. Throughout the campaign, the attackers thoroughly explored Modbus functionalities, performing register reads for data acquisition, issuing device identification queries, and attempting multiple coil writes aimed at altering control outputs—actions capable of disrupting physical processes in real environments. Notably, the attackers exhibited a clear progression between protocols, transitioning from Modbus operations to HTTP web server reconnaissance, which lasted approximately 1.5 hours, before shifting the focus to S7Comm communications, including the use of unsupported S7Comm-plus packets.

In conclusion, the presence of significant pauses between stages (ranging from minutes to nearly two hours), coupled with operational errors such as reusing Modbus address 200 for both scanning and coil writing, strongly suggests direct human involvement rather than fully automated scripts.

Custom Tools Development and Reverse Engineering. We also identified activity indicative of actors developing custom tools or attempting to reverse engineer protocol clients. The most notable example involved two IP addresses from France and the Netherlands, both belonging to the cloud provider SCALEWAY S.A.S. and flagged as suspicious by threat intelligence platforms. These actors targeted Honey-ICS deployments located in Asia and the US, as well as a Conpot instance in Europe, with interactions separated by nearly a month yet exhibiting identical patterns.

The interactions featured a custom, reverse-engineered S7Comm client that incorrectly attempted to initiate S7Comm-plus communication using static session identifiers. Typically, such session identifiers are dynamically generated for each connection to prevent replay attacks. After failing with S7Comm-plus attempts, the actor reverted to standard S7Comm operations, notably subscribing to PLC Mode-Transition events using the username USER1. This is a pattern consistent with Siemens TIA Portal behavior but implemented incorrectly.

Subsequently, the client issued malformed Read SZL requests, using indices that appeared to reference previously received PLC data rather than valid protocol parameters. This strongly indicates that the attackers were unsuccessfully attempting to parse and reutilize data extracted from PLC responses. Additionally, the French IP engaged in attempts to establish MMS/IEC 61850 protocol connections, reinforcing the conclusion that these interactions were part of broader ICS protocol experimentation. Overall, these patterns reflect an active (but flawed) effort to reverse engineer ICS client software.

Targeted Infrastructure Mapping and Web Interface Abuse. In another case, an attacker, originating from Germany, showcased sophisticated operational security awareness and advanced infrastructure mapping capabilities. This actor systematically identified the relationship between our geographically distributed proxies and the physical PLC, possibly using TLS certificate analysis, as evidenced by Shodan data which shows targeted scans on port 443 exclusively toward the physical PLC and specific proxy instances in Europe and the US, while notably excluding the Asia-based proxy. In particular, the attacker methodically navigated through the identified proxy chain, accessing first the European proxy, then the US proxy within two minutes, before reaching the actual PLC, clearly demonstrating an in-depth understanding of the infrastructure topology. In the subsequent step, remarkably, the attacker avoided traditional ICS protocol reconnaissance techniques (for both S7Comm and Modbus). Instead, the actor logged directly into the PLC's web administrative interface and executed a STOP command, effectively halting the simulated physical process. This approach demonstrates how a determined operator can achieve process disruption through legitimate administrative interfaces, bypassing traditional ICS security measures that focus on protocol-level attacks.

Modbus Interactions with Unusual TransID Patterns. We also discovered two Modbus interactions characterized by unusual TransID patterns that stood out from typical automated scanning behavior. In the first case, an attacker operating from a Vodafone Germany IP address targeted our Conpot instance in the corporate

network. The attacker repeatedly read the same set of 100 holding registers but incremented the TransIDs from 1 to 12 with each request. In the second case, another attacker used an Amazon-hosted IP address to connect to HoneyICS2 US instance and first browsed the web interface, then switched to the Modbus protocol to read exactly 16 discrete input registers. What made this interaction unusual was that the attacker aligned the TransIDs with the specific discrete input bits he/she was querying. In Modbus protocol, the TransID is typically just a sequence number used to match requests with responses and it has no functional relationship to the data being read. Most tools simply increment it or use random values. Both of these interaction patterns appeared exactly once in our dataset and do not match any known tool. This behavior indicates manual exploration or the use of custom scripts designed to probe how the honeypots respond to unconventional queries.

Answer to RQ5. Interactions not matching known tool fingerprints often reveal more sophisticated, likely manual attacker behavior that goes beyond standard automated scanning. These include cross-environment reconnaissance campaigns (e.g., the Chinese IP using `s7comm_scan.py` in a targeted manner) and complex, multi-stage human-guided attacks involving tool-chaining and protocol pivoting, as seen in the three-day Iranian campaign using Metasploit. Evidence also points to custom tool development or flawed reverse-engineering of clients (e.g., the Scaleway S7Comm client with static session IDs and incorrect indices), and advanced operational security awareness, such as attackers identifying proxy-PLC relationships and using web interfaces for direct process interruption (e.g., the German actor stopping the PLC). Furthermore, unusual protocol field manipulations, like non-standard Modbus TransID usage, suggest manual exploration or custom scripting.

6. Discussion

The results of this study provide several important insights into ICS honeypot deployments. These findings highlight key considerations for designing and deploying ICS honeypots more effectively in future research.

For ICS honeypots, network type shows larger differences than geographic region or interaction level. Our results show that the interaction level (RQ1) and geographic location (RQ3) of ICS honeypots have limited observable impact on the volume or complexity of traffic across protocols. In contrast, the network type affects the amount of ICS traffic (RQ2): honeypots in the corporate network received 91%

more S7Comm traffic than the cloud-host honeypots (Table 13).[5] This suggests that IPs associated with cloud infrastructure may be selectively avoided by attackers targeting ICS devices, while still being scanned for exposed web applications. Our qualitative analysis (Section 5.2) supports this interpretation, showing that interactions indicative of more sophisticated, targeted, or manually guided behavior occur more frequently in the corporate network. Our results partially align with prior work. Zou et al. [53], using an IT honeypot, found more IT traffic in Asia. While we found similar results for our IT traffic (HTTP), the same behavior cannot be seen in the OT world (Modbus and S7Comm protocols), where geographical differences appear limited in our data. This contrasts with Bieker et al. [5] and Srinivasa et al. [44], who reported higher ICS traffic in Asia and Europe, respectively. This discrepancy likely stems from methodological choices in prior work, such as aggregating results across protocols and interpreting scanning behavior as inherently malicious. In contrast, our protocol-specific analysis avoids strong assumptions about attacker intent, offering a more cautious interpretation that may more accurately reflect the nature of unsolicited ICS traffic. Our findings have two key implications. First, they highlight the importance of protocol-level analysis when designing honeypot studies or interpreting their results, as attacker behavior is strongly influenced by the specific services exposed. Second, within the deployment configurations studied here, corporate-network environments appear to be associated with higher levels of ICS-relevant interaction. However, confirming whether this effect generalizes across geographic regions requires deployments that vary network type and region independently. Future work should analyze how different combinations of ICS and IT services influence honeypot attractiveness to guide the design of more effective and targeted honeypot deployments and should deploy comparable honeypot configurations across multiple regions and network types to enable the analysis of interaction effects. This study evaluates the attractiveness of honeypot deployments based on interaction level, network type, and geographic location. However, broader geopolitical factors, such as regional conflicts, nation-state activities, or cyberespionage, may also influence attacker behavior. While certain patterns may

[5]These findings should be interpreted in light of a limitation of our experimental design. Corporate-network deployments were available only in Europe, whereas deployments in Asia and the United States were limited to non-corporate environments. Consequently, network type and geographic region are partially confounded, and the present study does not provide the factorial coverage required to estimate interaction effects between these variables. The observed differences therefore reflect associations within the available deployment configurations rather than fully isolated effects of network type or region.

suggest such influences, confirming them requires additional infrastructure, data, and attribution methods beyond the scope of this work. Future research could integrate external threat intelligence, geopolitical indicators, or attribution frameworks to assess the impact of political context on ICS-targeted activity.

Most ICS interactions are simple and automated, but filtering them enables the identification of more targeted and sophisticated behavior. Our analysis shows that most ICS interactions stem from automated reconnaissance, primarily scanning and device identification (RQ4). S7Comm activity is largely driven by open-source tools, likely due to the protocol's proprietary nature, which limits broader tool development. In contrast, Modbus, with its open specification and wider adoption, shows a greater variety of tools, including those linked to specific actors or custom-developed tools. Identifying and filtering these automated interactions is essential: it enables analysts to focus on the smaller, more meaningful portion of traffic that enables analysts to focus on targeted, sophisticated, or manually guided behavior. Our analysis of such interactions (RQ5) uncovered several notable cases, including multi-stage campaigns (e.g., the Metasploit-based attack from Iranian IPs), cross-environment reconnaissance (e.g., using `s7comm_scan.py`), and reverse-engineering attempts (e.g., the flawed S7Comm client from Scaleway IPs). We also observed deliberate tampering, such as a PLC being disabled through the web interface, and unusual protocol manipulations, such as non-standard Modbus TransIDs. These findings have two main implications. First, while most ICS traffic may appear noisy and low-value, filtering it reveals a smaller but more significant subset of interactions that offer insights into tactics, tooling, and exploitation strategies. This has practical relevance for both defenders and researchers: by isolating the most meaningful traffic, analysts can more effectively prioritize investigations, develop targeted detection rules, and assess emerging threat trends. Second, they highlight the importance of expanding the set of known fingerprints through systematic efforts, not just analyzing traffic, but actively surveying and dissecting publicly available tools (e.g., Metasploit modules). Broader fingerprint coverage allows for more accurate filtering, enabling faster identification of novel or high-risk behavior. Overall, our findings call for more refined analytical approaches that separate routine scanning from targeted ICS activity, improving the prioritization of monitoring and incident response.

For general threat intelligence, realistic ICS honeypots offer insights similar to simpler setups but can reveal more about occasional complex and targeted interactions. Our analysis indicates that both low- and high-interaction honeypots capture similar ICS traffic, regardless of deployment context (cloud vs. corporate)

or geographic region. This suggests that for gathering general threat intelligence, the added realism of high-interaction honeypots offers limited benefit. For this purpose, low-interaction honeypots provide a more cost-effective and operationally simpler choice. However, the occasional appearance of more complex interactions–even on low-interaction honeypots as seen with Modbus in our study–indicates that high-interaction honeypots may still play a role in capturing and analyzing rare but potentially insightful behavior as they are better suited for targeted investigations of attack methodologies and post-exploitation behaviors. This finding underscores the importance of aligning the honeypot type with its intended purpose, as advocated in [22]. Future work should aim to identify concrete criteria for when high-interaction honeypots are justified, enabling more informed trade-offs between realism, operational costs, and threat intelligence insights.

6.1. Threats to Validity

Internal Validity: Our analysis of interaction origins relies on the source IP addresses of incoming connections. The use of anonymization techniques (e.g., VPNs, TOR) can mask the true geographic origin of interactions and affect attribution accuracy. As a result, regional comparisons may be biased by misattribution, introducing confounding factors that limit causal interpretation. To assess this potential bias, we crosschecked the source IP addresses with publicly available lists of Tor exit nodes[6] and common VPN endpoints[7], using snapshots corresponding to the time of packet collection. Our investigation identified only 0.25% and 0.01% of TOR and VPN addresses, respectively. Moreover, we noticed that these IPs generated exclusively HTTP traffic, with no industrial protocols involved. In addition, cloud deployments are exposed through proxies that forward traffic to the underlying ICS environment. While this design enables consistent deployment across platforms, the use of proxies may introduce patterns in network characteristics (e.g., latency), which a very sophisticated attacker could theoretically notice. However, since the deployments are exposed on the internet, traffic already traverses a number of intermediate nodes with variable latency, making the additional overhead of a lightweight proxy negligible in comparison and unlikely to be meaningful for the attacker.

External validity: The data collection period for our study spans three months. While a longer period might have captured additional behaviors, our results indicate stable traffic patterns over time. Additionally, our deployment is centered on a sin-

[6] `https://github.com/alireza-rezaee/tor-nodes`

[7] `https://github.com/az0/vpn_ip`

gle PLC model and supports only the Modbus and S7Comm protocols. Future work could investigate the attractiveness of ICS environments featuring a broader range of PLC brands and ICS-specific protocols. Our study relies on AWS for cloud-based deployments. While prior research has reported variability in honeypot attractiveness across different cloud providers [21], these findings are based on emulated IT environments. It remains unclear whether similar differences apply in ICS contexts, highlighting the need for further investigation using multiple cloud platforms.

7. Conclusion

In this study, we examined how ICS honeypot configurations affect their attractiveness, focusing on interaction level (low vs. high), network type (corporate vs. cloud), and geographic region (Europe, North America, and Asia). Our infrastructure included low- and high-interaction honeypots emulating a Siemens Simatic S7-1200 PLC, along with a physical device, deployed across 16 setups and monitored for three months. We assessed attractiveness based on traffic volume, complexity, and nature. To support traffic analysis, we fingerprinted recurring Modbus and S7Comm patterns, enabling attribution to known tools and identification of notable ICS behavior. Results show that network type more strongly affects ICS traffic than geographic location or interaction level, with corporate deployments attracting more S7Comm interactions. While most traffic reflects automated reconnaissance, filtering reveals the presence of sophisticated and targeted behaviors. We find that low-interaction honeypots suffice for general threat intelligence, though high-interaction setups may add value in complex cases. Future work should define criteria for selecting honeypot types based on deployment goals and broaden fingerprint coverage through systematic analysis of publicly available ICS tools.

References

[1] 0day Today. `https://0day.today`. Accessed: 2025-06-06.

[2] E. Alata, V. Nicomette, M. Kaâniche, M. Dacier, and M. Herrb. Lessons learned from the deployment of a high-interaction honeypot. In *European Dependable Computing Conference*, pages 39–46. IEEE, 2006.

[3] V. Asanache, L. Lahaije, and K. A. Schlett. Fingerprinting automated industrial control system scanning tools. IST Seminar Report, Eindhoven University of Technology, 2024.

[4] Autonomy. https://autonomylogic.com/. Accessed: 2025-04-17.

[5] M. C. Bieker and D. Pilkington. Deploying an ICS honeypot in a cloud computing environment and comparatively analyzing results against physical network deployment. Master's thesis, Naval Postgraduate School, Monterey, CA, 2020.

[6] BinaryEdge. https://www.binaryedge.io. Accessed: 2025-06-06.

[7] Bitsight. https://www.bitsight.com. Accessed: 2025-06-06.

[8] R. Bloomfield, I. Gashi, A. Povyakalo, and V. Stankovic. Comparison of empirical data from two honeynets and a distributed honeypot network. In *International Symposium on Software Reliability Engineering*, pages 219–228. IEEE, 2008.

[9] D. Bove. Using honeypots to detect and analyze attack patterns on cloud infrastructures. Master's thesis, Friedrich-Alexander University, 2018.

[10] T. M. Chen and S. Abu-Nimeh. Lessons from stuxnet. *Computer*, 44(4):91–93, 2011.

[11] M. Conti, F. Trolese, and F. Turrin. Icspot: A high-interaction honeypot for industrial control systems. In *International Symposium on Networks, Computers and Communications (ISNCC)*, pages 1–4. IEEE, 2022.

[12] CrowdStrike. https://www.crowdstrike.com. Accessed: 2025-06-06.

[13] M. Dodson, A. R. Beresford, and M. Vingaard. Using global honeypot networks to detect targeted ICS attacks. In *International Conference on Cyber Conflict*, pages 275–291. IEEE, 2020.

[14] Driftnet. https://driftnet.io. Accessed: 2025-06-06.

[15] D. Efanov. PLC Device Scanner. https://packetstormsecurity.com/files/119726/PLC-Device-Scanner.html. Accessed: 2025-06-06.

[16] GreyNoise. https://www.greynoise.io. Accessed: 2025-06-06.

[17] J. D. Guarnizo, A. Tambe, S. S. Bhunia, M. Ochoa, N. O. Tippenhauer, A. Shabtai, and Y. Elovici. SIPHON: Towards Scalable High-Interaction Physical Honeypots. In *Workshop on Cyber-Physical System Security*, pages 57–68. ACM, 2017.

[18] S. Hilt. Nmap s7-info.nse. https://nmap.org/nsedoc/scripts/s7-info.html. Accessed: 2025-06-06.

[19] ip-api. IP Geolocation API. https://ip-api.com. Accessed: 2025-06-06.

[20] A. Jicha, M. Patton, and H. Chen. SCADA honeypots: An in-depth analysis of Conpot. In *Conference on Intelligence and Security Informatics*, pages 196–198. IEEE, 2016.

[21] C. Kelly, N. Pitropakis, A. Mylonas, S. McKeown, and W. J. Buchanan. A comparative analysis of honeypots on different cloud platforms. *Sensors*, 21(7):2433, 2021.

[22] S. Kempinski, S. Ichaarine, S. Sciancalepore, and E. Zambon. ICSvertase: A Framework for Purpose-based Design and Classification of ICS Honeypots. In *International Conference on Availability, Reliability and Security*, pages 1–10. ACM, 2023.

[23] Y. Kocaogullar, O. Cetin, B. Arief, C. Brierley, J. Pont, and J. Hernandez-Castro. Hunting high or low: Evaluating the effectiveness of high-interaction and low-interaction honeypots. In *Socio-Technical Aspects in Security*, page 14–30. Springer-Verlag, 2025.

[24] P. Kozak, I. Klaban, and T. Šlajs. Industroyer cyber-attacks on Ukraine's critical infrastructure. In *International Conference on Military Technologies (ICMT)*, pages 1–6. IEEE, 2023.

[25] E. López-Morales, C. Rubio-Medrano, A. Doupé, Y. Shoshitaishvili, R. Wang, T. Bao, and G.-J. Ahn. Honeyplc: A next-generation honeypot for industrial control systems. In *SIGSAC Conference on Computer and Communications Security*, pages 279–291. ACM, 2020.

[26] M. Lucchese, F. Lupia, M. Merro, F. Paci, N. Zannone, and A. Furfaro. HoneyICS: A high-interaction physics-aware honeynet for industrial control systems. In *International Conference on Availability, Reliability and Security*. ACM, 2023.

[27] F. Lupia, M. Lucchese, M. Merro, and N. Zannone. ICS Honeypot Interactions: A Latitudinal Study. In *International Conference on Big Data*, pages 3025–3034. IEEE, 2023.

[28] S. Maesschalck, V. Giotsas, and N. Race. World wide ICS honeypots: A study into the deployment of Conpot honeypots. In *Industrial Control System Security Workshop*, 2021.

[29] M. Mesbah, M. S. Elsayed, A. D. Jurcut, and M. Azer. Analysis of ICS and SCADA Systems Attacks Using Honeypots. *Future Internet*, 15(7), 2023.

[30] S. Morishita, T. Hoizumi, W. Ueno, R. Tanabe, C. Gañán, M. J. Van Eeten, K. Yoshioka, and T. Matsumoto. Detect me if you... oh wait. an internet-wide view of self-revealing honeypots. In *Symposium on Integrated Network and Service Management*, pages 134–143. IEEE, 2019.

[31] D. Nardella. Snap7. `https://snap7.sourceforge.net/`. Accessed: 2025-06-06.

[32] Netfilter. The netfilter.org "iptables" project. `https://www.netfilter.org/projects/iptables/index.html`. Accessed: 2025-06-06.

[33] Nmap.org. Nmap modbus-discover.nse. `https://nmap.org/nsedoc/scripts/modbus-discover.html`. Accessed: 2025-05-29.

[34] OffSec. Exploit database. `https://www.exploit-db.com`. Accessed: 2025-06-06.

[35] F. Ondrikov, D. Donadel, F. Lupia, M. Merro, D. dos Santos, E. Zambon, and N. Zannone. A Comparative Study of ICS Honeypot Deployments. In *International Workshop on Cyber-Physical Security for Critical Infrastructures Protection*, LNCS 16233, pages 125–145. Springer, 2025.

[36] ONYPHE. `https://www.onyphe.io/`. Accessed: 2025-06-06.

[37] Packet Storm Security. `https://packetstorm.news`. Accessed: 2025-06-06.

[38] F. Pouget and T. Holz. A pointillist approach for comparing honeypots. In *Detection of Intrusions and Malware, and Vulnerability Assessment*, pages 51–68. Springer, 2005.

[39] N. Provos. A virtual honeypot framework. In *USENIX Security Symp.*, pages 1–14, 2004.

[40] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic. Cyber-physical systems: The next computing revolution. In *Design Automation Conference*, pages 731–736, 2010.

[41] Y. Shan, Y. Yao, T. Zhao, and W. Yang. NeuPot: A Neural Network-Based Honeypot for Detecting Cyber Threats in Industrial Control Systems. *IEEE Transactions on Industrial Informatics*, 19(10):10512–10522, 2023.

[42] Shodan. `https://www.shodan.io/`. Accessed: 2025-06-06.

[43] T. Sochor and M. Zuzcak. Attractiveness study of honeypots and honeynets in internet threat detection. In *International Conference on Computer Networks*, pages 69–81. Springer, 2015.

[44] S. Srinivasa, J. Pedersen, and E. Vasilomanolakis. Interaction matters: a comprehensive analysis and a dataset of hybrid IoT/OT honeypots. In *Annual Computer Security Applications Conference*, pages 742–755. ACM, 2022.

[45] S. Srinivasa, J. Pedersen, and E. Vasilomanolakis. Gotta catch'em all: a multistage framework for honeypot fingerprinting. *Digital Threats: Research and Practice*, 4(3):1–28, 2023.

[46] A. Tambe, Y. L. Aung, R. Sridharan, M. Ochoa, N. O. Tippenhauer, A. Shabtai, and Y. Elovici. Detection of threats to IoT devices using scalable VPN-forwarded honeypots. In *Conference on Data and Application Security and Privacy*, pages 85–96. ACM, 2019.

[47] Tcpdump. `https://www.tcpdump.org`. Accessed: 2025-06-06.

[48] C. Vasilatos, D. J. Mahboobeh, H. Lamri, M. Alam, and M. Maniatakos. LLMPot: Dynamically Configured LLM-based Honeypot for Industrial Protocol and Physical Process Emulation. In *European Symposium on Security and Privacy (EuroS&P)*, pages 963–979. IEEE, 2025.

[49] A. Vetterl and R. Clayton. Bitter Harvest: Systematically Fingerprinting Low- and Medium-interaction Honeypots at Internet Scale. In *USENIX Workshop on Offensive Technologies*. USENIX Association, 2018.

[50] J. You, S. Lv, L. Zhao, M. Niu, Z. Shi, and L. Sun. A scalable high-interaction physical honeypot framework for programmable logic controller. In *Vehicular Technology Conference*, pages 1–5. IEEE, 2020.

[51] ZGrab 2.0. https://github.com/zmap/zgrab2. Accessed: 2025-06-06.

[52] W. Zhu. s7comm_scan.py. https://github.com/dark-lbp/isf/blob/master/icssploit/modules/scanners/s7comm_scan.py. Accessed: 2025-03-24.

[53] J. Zou, Z. Sun, C. Ku, X. Li, and A. Dahbura. WiP: Developing High-interaction Honeypots to Capture and Analyze Region-Specific Bot Behaviors. In *Hot Topics in the Science of Security Symposium*, 2024.

Appendix A. Modbus fingerprints

Table A.18 presents the Modbus fingerprints.

Table A.18: Modbus fingerprints

Tool		Transaction ID	Unit ID	Function Code
`Zgrab2` [51]		23111	0	43/1
`Nmap modbus-discover.nse` [33]	Conpot	0	0	17
		0	1	17
		0	1	43/1
		0	255	17
		0	255	43/1
	HoneyICS	0	0	17
	Physical PLC	0	0	43/1
		0	1	17
		0	1	43/1
		0	255	17
		0	255	43/1
`Tool A`		4919	0	43/1
`Tool B`		0	0	43/1
`Tool C`		[0,65535] \ {0,1000,4919,23111,34420,46479}	0	43/1
`Tool D`		[0,65535]	1	43/1
`Tool Bitsight` [7]	Copot	1000	0	43/1
	HoneyICS	1000	0	43/1
	Physical PLC	1001	1	43/1
		1255	255	43/1
`Tool Driftnet` [14]	Conpot	x∈ [0,65535]	1	43/1
		x+1	1	43/1
		x+2	1	43/1
		x+3	1	17
	HoneyICS	x ∈ [0,65535]	1	43/1
		x+1	1	17
	Physical PLC	x ∈ [0,65535]	1	43/1
`Tool Crowdstrike` [12]	Conpot	0	0	3
		0	1	3
		0	9	3
	HoneyICS	0	0	3
	Physical PLC	0	1	3
		0	2	3
		0	3	3
		0	4	3
		0	5	3
		0	6	3
		0	7	3

Continued on next page

Table A.18: Modbus fingerprints

Tool		Transaction ID	Unit ID	Function Code
		0	8	3
		0	9	3
`Tool Ningxia`		0	0	43/1
		0	3	43/1
`Port Scan BinaryEdge.io` [6]		12420	2	1
		256	0	1
`Port Scan Ucloud`	Conpot	34420	0	43/1
		12420	2	1
		450	0	0
	HoneyICS	46479	0	43/1
	Physical PLC	34420	0	43/1
		46479	0	43/1

Appendix B. Results

Appendix B.1. Interactions and Complexity

Table B.19 presents an overview of the interactions captured by the different deployments and their complexity.

Table B.19: Interactions and complexity

	Interactions	Conpot						HoneyICS1						HoneyICS2						Physical PLC					
		Min	Q1	Mdn	Q3	Max	Count	Min	Q1	Mdn	Q3	Max	Count	Min	Q1	Mdn	Q3	Max	Count	Min	Q1	Mdn	Q3	Max	Count
Corporate Network	HTTP	1	1	1	2	8003	8146	1	1	1	2	7926	7748	1	1	1	2	8021	7759	1	1	1	1	363	7047
	S7comm	1	3	3	3	4	490	1	3	3	3	4	594	1	3	3	3	7	580	1	3	3	3	4	594
	Modbus	1	1	1	1	241	510	1	1	1	2	10	382	1	1	1	2	10	339	1	1	1	1	10	354
EU	HTTP	1	1	1	2	7115	11090	1	1	1	2	7109	10620	1	1	1	2	7110	11734	1	1	1	2	7269	10921
	S7comm	1	3	3	3	8	256	1	3	3	3	4	301	1	3	3	3	4	317	1	3	3	3	4	309
	Modbus	1	1	1	1	5	314	1	1	1	1	10	334	1	1	1	1	10	345	1	1	1	1	10	320
US	HTTP	1	1	1	2	19994	10800	1	1	1	2	7112	10382	1	1	1	2	7112	10990	1	1	1	2	7365	9214
	S7comm	1	3	3	3	4	280	1	3	3	3	4	302	1	3	3	3	16	303	1	3	3	3	4	297
	Modbus	1	1	1	1	4	314	1	1	1	1	10	328	1	1	1	1	16	337	1	1	1	1	10	324
Asia	HTTP	1	1	1	2	7113	12285	1	1	1	2	12534	11658	1	1	1	2	7113	11387	1	1	1	2	7194	11199
	S7comm	1	3	3	3	4	292	1	3	3	3	16	338	1	3	3	3	4	329	1	3	3	3	4	284
	Modbus	1	1	1	1	5	349	1	1	1	1	10	349	1	1	1	1	10	327	1	1	1	1	10	316

Appendix B.2. Fingerprint Coverage

Table B.20 presents the fingerprint coverage for S7comm.

Table B.20: S7comm Fingerprint Coverage

	Tool	Conpot	HoneyICS1	HoneyICS2	Physical PLC
Corporate Network	Non Covered Interactions	4	21	4	18
	Total Interactions	490	594	580	594
	`Zgrab2`	438	478	484	485
	`PLC Device Scanner`	8	49	44	48
	`Nmap s7-info.nse`	9	13	16	14
	`S7comm_scan.py`	0	1	1	0
	`Tool ONYPHE`	31	32	31	29
	Coverage	99.2%	96.5%	99.3%	97.0%
EU	Non Covered Interactions	8	1	0	1
	Total Interactions	256	301	317	309
	`Zgrab2`	233	252	275	276
	`PLC Device Scanner`	3	4	3	3
	`Nmap s7-info.nse`	3	9	9	10
	`S7comm_scan.py`	1	0	1	0
	`Tool ONYPHE`	8	35	29	19
	Coverage	96.9%	99.7%	100.0%	99.7%
US	Non Covered Interactions	10	0	2	1
	Total Interactions	280	302	303	297
	`Zgrab2`	243	267	282	273
	`PLC Device Scanner`	8	4	2	6
	`Nmap s7-info.nse`	1	9	10	9
	`S7comm_scan.py`	1	0	1	0
	`Tool ONYPHE`	17	22	6	8
	Coverage	96.4%	100.0%	99.3%	99.7%
Asia	Non Covered Interactions	10	4	1	1
	Total Interactions	292	338	329	284
	`Zgrab2`	239	283	294	266
	`PLC Device Scanner`	5	8	4	3
	`Nmap s7-info.nse`	3	11	10	5
	`S7comm_scan.py`	1	0	1	0
	`Tool ONYPHE`	34	32	19	9
	Coverage	96.9%	98.8%	99.7%	99.6%

Table B.21 presents the fingerprint coverage for Modbus.

Table B.21: Modbus Fingerprint Coverage

	Tool	Conpot	HoneyICS1	HoneyICS2	Physical PLC
Corporate Network	Non Covered Interactions	47	1	15	10
	Total Interactions	510	382	339	354
	`Zgrab2`	297	196	169	198
	`Nmap modbus-discover.nse`	42	65	36	57
	`Tool A`	54	54	58	24
	`Tool B`	2	4	4	1
	`Tool C`	7	2	2	1
	`Tool D`	1	3	2	5
	`Tool Bitsight`	10	9	11	10
	`Tool Driftnet`	30	31	29	29
	`Tool Crowdstrike`	8	8	8	9
	`Tool Ningxia`	5	3	0	4
	`Port Scan BinaryEdge.io`	1	0	0	0
	`Port Scan Ucloud`	6	6	5	6
	Coverage	90.8%	99.7%	95.6%	97.2%
EU	Non Covered Interactions	17	0	0	0
	Total Interactions	314	334	345	320
	`Zgrab2`	166	192	195	192
	`Nmap modbus-discover.nse`	1	0	0	1
	`Tool A`	66	78	89	67
	`Tool B`	6	6	5	6
	`Tool C`	5	4	3	5
	`Tool D`	4	1	1	2
	`Tool Bitsight`	7	12	7	7
	`Tool Driftnet`	20	24	24	22
	`Tool Crowdstrike`	7	9	9	8
	`Tool Ningxia`	7	2	6	4
	`Port Scan BinaryEdge.io`	2	0	0	0
	`Port Scan Ucloud`	6	6	6	6
	Coverage	94.6%	100.0%	100.0%	100.0%
US	Non Covered Interactions	11	0	1	1
	Total Interactions	314	328	337	324
	`Zgrab2`	162	183	192	197
	`Nmap modbus-discover.nse`	0	0	2	0
	`Tool A`	75	82	78	64
	`Tool B`	6	7	6	6
	`Tool C`	5	3	5	2
	`Tool D`	3	3	3	3
	`Tool Bitsight`	10	12	7	9
	`Tool Driftnet`	24	24	24	22
	`Tool Crowdstrike`	8	7	9	11
	`Tool Ningxia`	3	2	4	3
	`Port Scan BinaryEdge.io`	3	0	0	0
	`Port Scan Ucloud`	4	5	6	6
	Coverage	96.5%	100.0%	99.7%	99.7%
Asia	Non Covered Interactions	16	0	0	2
	Total Interactions	349	349	327	316
	`Zgrab2`	183	192	194	189
	`Nmap modbus-discover.nse`	1	0	1	2
	`Tool A`	89	90	76	61
	`Tool B`	4	7	6	5
	`Tool C`	3	3	3	6
	`Tool D`	1	5	2	0
	`Tool Bitsight`	6	9	7	6
	`Tool Driftnet`	24	24	20	26
	`Tool Crowdstrike`	8	9	7	11
	`Tool Ningxia`	5	3	5	3
	`Port Scan BinaryEdge.io`	3	0	0	0
	`Port Scan Ucloud`	6	7	6	5
	Coverage	95.4%	100.0%	100.0%	99.4%