Interaction matters: a comprehensive analysis and a dataset of hybrid IoT/OT honeypots

Shreyas Srinivasa
Aalborg University
Denmark

Jens Myrup Pedersen
Aalborg University
Denmark

Emmanouil Vasilomanolakis
Technical University of Denmark
Denmark

ABSTRACT

The Internet of things (IoT) and critical infrastructure utilizing operational technology (OT) protocols are nowadays a common attack target and/or attack surface used to further propagate malicious actions. Deception techniques such as honeypots have been proposed for both IoT and OT but they either lack an extensive evaluation or are subject to fingerprinting attacks. In this paper, we extend and evaluate RIoTPot, a hybrid-interaction honeypot, by exposing it to attacks on the Internet and perform a longitudinal study with multiple evaluation parameters for three months. Furthermore, we publish the aforementioned study in the form of a dataset that is available to researchers upon request. We leverage RIoTPot's hybrid-interaction model to deploy it in three interaction variants with six protocols deployed on both cloud and self-hosted infrastructure to study and compare the attacks gathered. At a glance, we receive 10.87 million attack events originating from 22, 518 unique IP addresses that involve brute-force, poisoning, multistage and other attacks. Moreover, we fingerprint the attacker IP addresses to identify the type of devices who participate in the attacks. Lastly, our results indicate that the honeypot interaction levels have an important role in attracting specific attacks and scanning probes.

CCS CONCEPTS

• **Security and privacy → Network security**.

KEYWORDS

honeypots, deception, operation technology, IoT

ACM Reference Format:
Shreyas Srinivasa, Jens Myrup Pedersen, and Emmanouil Vasilomanolakis. 2022. Interaction matters: a comprehensive analysis and a dataset of hybrid IoT/OT honeypots. In *Annual Computer Security Applications Conference (ACSAC), December 5–9, 2022, Austin, TX, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3564625.3564645

1 INTRODUCTION

The number of cyber attacks targeting mission-critical infrastructure has increased steadily [20]. Recent attacks on industrial systems like the US Colonial Pipeline [22] and the Florida Water Treatment plant [35] have shown the impact caused on both the public and the government. Mission-critical systems in Operational Technology (OT) rely on sensors and connected devices to automate industrial control processes. However, a study of recent attacks reveals a lack of security on these devices [50]. Furthermore, the vast adoption of IoT devices in both consumer and industrial applications has increased the attack space. Research shows a large number of vulnerable, misconfigured IoT devices exposed to the Internet and sophisticated malware that can exploit them [16, 40]. Moreover, the ENISA Threat Landscape Report 2020 states that malware continues to be the most challenging attack vector and up to 230, 000 variants are distributed daily [10].

Honeypots are deception systems that act as a trapping mechanism for attackers. They have low false positives as there is no justification to communicate with a honeypot, and hence, all communication can be considered suspicious. Honeypots are classified into low, medium, and high interaction based on the interaction capabilities they offer to the attackers. Over the years, many honeypots have been proposed for various protocols or device profiles. Well known examples include Cowrie [26], Conpot [34], HosTaGe [46], Glastopf [33], Dionaea [42], and T-Pot [29].

Honeypots come with some limitations as a result of either the lack of interaction, their static responses, or poor maintenance. Honeypot fingerprinting is the process of determining if the system in communication is a honeypot through probing mechanisms. Successful fingerprinting attacks can undermine the value of honeypots as their identity is exposed. A number of techniques for fingerprinting honeypots have been proposed in recent research [23, 39, 47]. Moreover, many open-source honeypot projects are abandoned or depend on libraries that are not in active maintenance; this leads to a lack of extensibility and reduced scope [39].

In this paper, we extend RIoTPot, a modular and hybrid-interaction honeypot that addresses the gap between extensibility and interaction to simulate application-layer protocols used in IT, IoT, and OT environments [41]. The contributions of this paper are as follows.

- We extend RIoTPot and provide the source code for IoT and OT device profiles that emulate the Telnet, SSH, MQTT, Modbus, CoAP, and HTTP protocols.
- We perform a longitudinal study of RIoTPot imposing many evaluation parameters based on interaction-level, simulation environment, deployment infrastructure, and geolocation. We report our findings and discuss the impact of the evaluation parameters on the operation of honeypots. In addition, we examine how RIoTPot performs compared to another popular state of the art honeypot.
- We offer the attack dataset to the research community.

The rest of the paper is structured as follows. Section 2 provides an overview of the related work in the use of honeypots for analyzing attack trends. We discuss RIoTPot, the extensions we implemented and the offered dataset in Section 3. In Section 4, we outline the methodology of our study and the experimental setup and evaluate our approach in Section 5. We discuss our findings in Section 6 and conclude in Section 7.

2 RELATED WORK

Honeypots are commonly classified into low, medium, and high-interaction based on the interaction level they offer to attackers. Low-interaction honeypots offer limited simulations of a protocol or service and are easy to manage. Medium-interaction offer an extended interaction level to low-interaction and involve emulating a device that constitutes multiple protocols. High-interaction honeypots are real (or virtual) systems/devices with customized logging, limited egress traffic rules, and ephemeral configurations to prevent misuse. At a glance, various well-known low/medium interaction honeypots include: Cowrie [26], Conpot [34], HosTaGe [46], Glastopf [33], Dionaea [42] and TPot [29]. In addition, there are some high-interaction honeypots proposed such as Honware [48], Siphon [13], and Sarracenia [36]. All of the aforementioned honeypots operate at low or high interaction modes thus offering a binary interaction capability. Moreover, they vary a lot on how much extensibility they offer for integrating additional protocols or targeting specific environments.

Another factor that is important to be taken into account is fingerprinting; adversaries may be able to detect if a target system is a honeypot using techniques that leverage minimal data obtained from the target system. On the one hand, recent research reveals that while low/medium-interaction honeypots are easy to manage and are minimal in risk, they are more vulnerable to fingerprinting attacks [39, 47]. On the other hand, high-interaction honeypots risk a compromised environment that can be leveraged for malicious activities entailing higher maintenance.

Litchfield et al. propose HoneyPhy to address the issues of limited simulation of honeypots that simulate cyber-physical environments [18]. The authors argue that the honeypots fail to emulate the actual behavior of cyber-physical systems which can lead to fingerprinting. The authors extend their work by proposing HoneyBot; a hybrid-interaction honeypot designed explicitly for networked robotic systems [15] that is capable of switching interaction based on the attack requests. However, HoneyBot is limited to a specific environment and cannot be extended for diverse operations.

The evaluation of honeypots in research studies that aim at presenting an overview of the attack landscape is not uncommon [1, 5, 9]. Dang et al. performed a longitudinal study of 12 months in a combined approach with both hardware and software honeypots [7]. The experiment involved the deployment of four hardware and 108 software IoT honeypots in gathering attacks on the IoT landscape. The software/virtual honeypots are deployed on eight public cloud providers and various geolocations. The authors present an overview of the attacks received on the honeypots and the implications. Minn et al. propose *IoTPOT*, a honeypot that simulates Telnet services from various IoT devices, to study the attack trends on the Telnet protocol [27]. The *IoTPOT* honeypot consists of a low-interaction front-end connected with a high-interaction back-end called *IoTBOX* that simulates the Telnet service from various IoT device profiles. The authors deployed the honeypot for 39 days and collected 43 distinct malware samples.

Srinivasa et al. find a large number of misconfigured IoT devices on the Internet and deploy six open-source IoT honeypots to study attack trends [40]. Tabari et al. present a multi-faceted IoT-honeypot ecosystem with extendable sophistication by observing real-world attacks [52]. The authors develop a honeypot for IoT cameras to observe the attack landscape around them. Furthermore, the authors propose *ProxyPot*, a honeypot proxy that sits between IoT devices and external networks to observe inbound and outbound traffic. The IoT camera honeypots were deployed for two years and an increase in the number of attacks was observed.

Vetterl et al. propose *Honware*, a high-interaction, virtual honeypot framework that can emulate a wide range of IoT device firmware without the hardware [48]. The authors evaluated the honeypot by deploying four device profiles of ADSL modems and found various attacks targeting vulnerabilities specific to emulated devices. Furthermore, Guarnizo et al. present *Siphon*, a scalable high-interaction honeypot architecture which utilizes IoT devices that are physically deployed at a geographical location and connected to the Internet for simulation [13]. The authors deploy 85 honeypot instances, in various locations, that utilize five physical cameras, an NVR, and an IP printer. An analysis of the attacks revealed that honeypots in certain cities received more attack traffic compared to others. Valeros et al. provide *Hornet 40*, a network dataset of geographically placed honeypots to study the impact of geolocation [44]. The data consists of 118 features, including 480 bytes of content for each flow. The dataset does not contain interactive attack traffic as only passive honeypots were used in the study. The attack data is collected from honeypots deployed at eight locations and contain 4.7 million network flows collected over a period of 40 days.

Barron et al. performed a four-month study that involved 102 medium-interaction honeypots [3]. In addition to deploying the honeypots at multiple locations, the authors experiment with parameters like the difficulty in the break-in and file generation. They observe how the differences in these parameters caused deviance in the attackers' behavior. In addition, the authors leak the access information of the honeypots on hacking forums and paste sites to monitor the attackers' actions. The authors find that the differences in the parameters introduced affected the human-based attackers and list key takeaways from the experiment. We consider this work the closest to our approach, where the authors introduce specific parameters in the study and how they influence the attacker.

Appendix Table 6 provides an overview of the qualitative comparison of honeypots proposed in related work. The honeypots are compared based on their source-code availability, supported protocols, interaction level, operational environments and known fingerprinting techniques. Most of the proposed honeypots are available as open source and support multiple protocols. However, we observe that there are no high-interaction honeypots are available as open source. Furthermore, we observe that for open source honeypots, certain fingerprinting methods have been proposed. Most of the honeypots are designed to be deployed as virtual environments except IoTPot, which runs on hardware. While most of the honeypots operate at binary interaction levels, i.e. low or medium or high, RIoTPot is capable of operating in either low, high or hybrid interaction levels.

The related work listed in Table 1 summarizes research that involves the deployment of honeypots to study different attack surfaces. Nevertheless, none of the studies compare attacks by deploying honeypots with varied interaction levels and operating environments for multiple protocols. Furthermore, there are no public datasets that offer diverse data based on interaction levels and protocols. We aim to address this gap by deploying multiple honeypot instances with varied interaction levels and operating environments that are geographically distributed. In addition, we analyze and compare the attacks received on multiple interaction levels simulated on different IoT and OT application protocols.

Study	**Interaction level**	**Study period**	**Geographically distributed**	**Deployment**
Honeycloud [7] (2019)	Medium	12 months	Yes	hardware, cloud
IoTPOT [27](2015)	Low	39 days	No	physical
Open for hire [40] (2021)	Low, Medium	1 month	No	physical
Muti-faceted Honeypot [52](2020)	Low	2 years	No	physical
Honware [48] (2019)	High	14 days	No	physical
Siphon [13](2017)	High	2 months	Yes	physical, cloud
Hornet 40 [44](2021)	Passive	40 days	Yes	cloud
Picky Attackers [3] (2017)	Medium	4 months	Yes	physical, cloud
RIoTPot (2022)	**Low, High, Hybrid**	3 **months**	**Yes**	**physical, cloud**

Table 1: Overview of related work & evaluation parameters

3 EXTENDING RIOTPOT

RIoTPot aims to address the limitations of extensibility, interaction, and operational environments [41]. It follows a modular architecture and offers hybrid interaction levels. We extend RIoTPot[1] to adapt to this longitudinal study along with various enhancements (see also Figure 1). RIoTPot offers multiple features such as extensibility, mode of operations, and compatibility with diverse deployment environments.

The motivation for RIoTPot is to offer administrators with the ease of deploying honeypots in their infrastructure, especially high-interaction honeypots in the form of containers. RIoTPot is open-source and can run on virtual infrastructure, unlike other high-interaction honeypots. Furthermore, RIoTPot offers administrators with the flexibility of changing the interaction level based on the resources. RIoTPot is able to start being deployed as low-interaction and switch to high-interaction on the fly. RIoTPot is designed to focus on modularity to facilitate the integration of additional protocols and maintenance. Many open-source projects are often abandoned due to the lack of extensibility. Modular architecture addresses this gap by providing extensibility in the form of modules. For low-interaction mode, honeypot administrators can use the default template for adding new simulations and easily integrate them with the startup configuration. Similarly, for the high-interaction mode, the path for the container image is provided for simulation. Through a modular implementation, administrators can add any relevant scenarios to the RIoTPot simulation portfolio.

Many honeypot projects face the threat of fingerprinting (see Section 2). Fingerprinting enables an adversary to detect honeypots based on elementary information obtained through crafted request probes. This is particularly harmful for low and medium interaction honeypots that are often utilizing specific libraries or hard-coded responses that can be fingerprinted. Nevertheless, such honeypots are of low maintenance and risk and thus an attractive deception mechanism. The hybrid interaction feature of RIoTPot provides honeypot administrators with a choice of operating the emulation in either low, high, or hybrid interaction. Hybrid allows some protocols to run in low and some on high interaction. This allows defenders to lure attackers into specific protocols in a breadcrumb-like approach. For instance, the administrators can run a protocol that interests them on high interaction and some other (that would be expected by the attacker) in low. The attacker would then eventually spend most of their time and effort on the high interaction protocol.

While a number of honeypot projects are developed focusing on containerized deployment, RIoTPot also offers a hybrid deployment scenario where administrators can choose to run the high-interaction containers on remote hosts or local infrastructure. This feature is beneficial in resource-constrained environments. For example, RIoTPot can be deployed on Raspberry Pi, and the high-interaction containers can be deployed in a cloud environment. Furthermore, if the simulation profile requires specific hardware (e.g., sensors), RIoTPot simulation containers can be deployed on the supporting infrastructure.

3.1 RIoTPot extended architecture

The architecture of RIoTPot follows modularity that enables quick integration of protocols and emulation modules [41]. Modular software architecture is a structural approach to building software components as modules by separating the functionality of a program into independent, interchangeable modules, such that each contains everything necessary to execute only one aspect of the desired functionality [21]. Figure 1 shows the extended architecture adapted from RIoTPot [41]. In addition to the quick integration of additional modules, the architecture facilitates the integration of extensions to support extended analysis or configuration. The prominent modules in the architecture are the *RIoTPot core* module, the *low-interaction* modules, the *high-interaction* modules, and the *attack database*. The *packet capture and noise filter* modules serve as extensions to support further analysis of the attack data.

3.2 Extended components

3.2.1 RIoTPot Core. The *RIoTPot core* consists of the essential components that enable the configuration, administration and orchestration services. The core module facilitates the administrators to configure the parameters like specifying the protocols for emulation, the desired interaction-level and the path for the loading the images in case of high interaction mode. The startup configuration allows users to choose the desired protocols and the interaction-level. In addition to the configuration, the core facilitates the network management between the containers in the high interaction mode. The traffic is forwarded to the containers based on the simulated protocols on which the attacks are targeted. Furthermore, the core is responsible for communication with remote containers, in case of a cloud-based deployment. We extend the configuration module of the core by enhancing the static file-based configuration to a shell-based interactive configuration. The shell-based configuration provides interactive prompts where the desired startup configuration can be chosen by the administrator. The prompts include selection of protocols for emulation, choosing the interaction levels as operation modes, providing the parameters for remote database, *pcap* repository and the container images in case of operation in high interaction mode.

[1]https://github.com/aau-network-security/riotpot

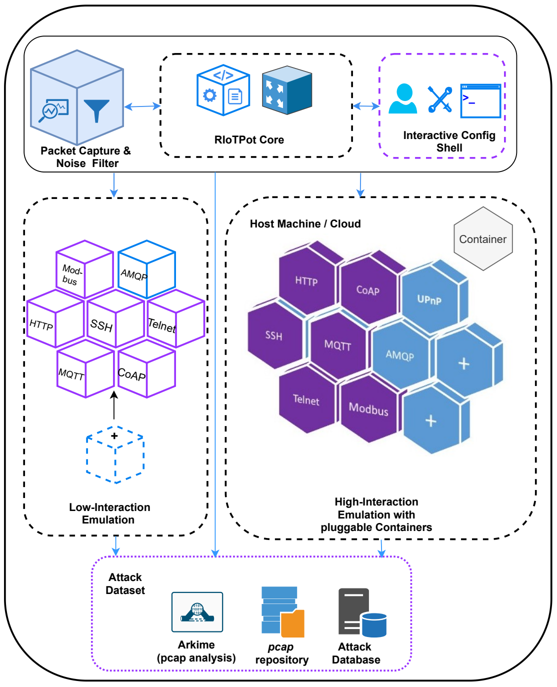

Figure 1: RIoTPot architecture adapted from [41], purple components suggest novel or enhanced parts

3.2.2 Low-interaction modules. The *low-interaction mode* is enabled by the implementation of packages that simulate individual protocols. RIoTPot is implemented in Go language [11] that enables development of independent packages. The modular architecture of RIoTPot is achieved by developing the protocols as independent packages that can further be integrated as plug-ins. For example, the Telnet and the SSH protocols use the *fakeshell* package that emulates a system shell and responds to a list of commands. The *fakeshell* package can be extended to include more commands. A default template is provided with RIoTPot that can be leveraged for the integration of additional protocols. By default, RIoTPot supports emulation of seven protocols that include Telnet, SSH, HTTP, MQTT, AMQP, Modbus and CoAP. We extend RIoTPot by enhancing the Telnet, SSH, HTTP, MQTT, Modbus and CoAP modules for adapting to our study. The changes include extending the shell emulation capabilities for the *fakeshell* module and enhancing the emulation capabilities for the Modbus, MQTT and CoAP protocols.

3.2.3 High-interaction modules. RIoTPot operates in *high-interaction mode* by emulating protocols as services running on container images. For each configured protocol, the administrator provides a relevant image that will be deployed on a container for emulation. As the protocols operate as full services on the containers, they act as high interaction modules that provide a full implementation of a protocol and thereby provides the attackers with high interaction capabilities. Additional protocols can be integrated by specifying the protocol and the image path in the startup configuration. The high-interaction modules have been extended in this work by leveraging container images provided through the Docker Hub repository [8]. An advantage of using Docker Hub is that it employs a verification procedure for most of its images (e.g., verifying that the Apache Foundation is the publisher of the httpd image and BusyBox for the busybox image). We use the Busybox [4], OpenSSH [17], HTTPD [31], Modbus-Server [25], Eclipse-Mosquitto [30] and CoAP-Gateway [28] images for high-interaction of protocols.

3.2.4 Hybrid interaction. The hybrid interaction mode allows to choose the desired interaction level for selective protocols. Through the hybrid interaction mode, a specific protocol like SSH can run in low interaction mode while another (e.g., HTTP) can run in high interaction mode. This facilitates administrators to set up device profiles that constitute a collection of protocols with less resource requirements and can choose the hybrid operation mode through the interactive shell configuration prompts during startup.

3.2.5 Noise filter and packet capture. RIoTPot has two default extensions - the attack capture component and the noise filter. The attack capture component stores all the traffic received on RIoTPot as *pcap* files using *tcpdump* to facilitate comprehensive analysis. Through the attack capture extension, the users can further specify the required rotation levels for the packet capture. The attack capture component is responsible for storing the attack packets as *pcap* files, using *tcpdump*, which can be used for detailed analysis (e.g., deep packet inspection). The noise filter component filters out the traffic received from Internet-wide scanners like Shodan [38] and Censys [6]. The component can filter traffic from 19 Internet scanning services. The attack sources are labeled accordingly in the attack database. This helps the administrator concentrate on attacks that matter by removing the noise generated by such services.

3.2.6 Attack dataset. The traffic received on RIoTPot is stored in the *attack database*. It is provisioned as an independent container to ensure no disruption in logging in the event of a crash or failure. The attacks received on both low and high interaction modes are stored in the database. The database instance is accessible from the low and high interaction containers. To facilitate this longitudinal study we extend RIoTPot to log all the attacks to a remote database in the cloud that ensures scalability and simplified backup process. We further enhance RIoTPot by integrating *Arkime*, an open-source indexed packet capture and searching tool [51]. We leverage Arkime to search through the *pcap* files generated from the honeypot deployments. Arkime imports the *pcap* files from the *pcap* repository and stores them in its back-end (Elasticsearch) for indexed searching capability. Furthermore, Arkime supports querying based on attributes and integration with Virustotal [49] that helps in identification of malicious events and sources.

4 METHODOLOGY

Our work aims to capture attacks on the IoT and OT landscape and evaluate RIoTPot by leveraging its hybrid-interaction operational feature. Furthermore, we impose six evaluation parameters to observe their influence on our experiment and compare the results. We describe the evaluation parameters and the experimental setup of our longitudinal study in the following sections.

4.1 Evaluation parameters

4.1.1 Interaction levels. As RIoTPot can operate in low, high, and hybrid interaction levels, we study and compare the attacks received on each interaction level. The analysis based on interaction levels will provide an insight into the effectiveness of deception used on each interaction level.

4.1.2 Multiple honeypot instances. By deploying multiple instances of honeypots, we get a better understanding of the observed attacks. For instance, an attack from a specific IP identified on all honeypot instances can construe that the attack source is either an Internet-wide scanning service or part of a reconnaissance process from a bot. On the contrary, unique attack sources identified in specific instances provide insight into distinct attack types and approaches.

4.1.3 Deployment infrastructure. The honeypots' deployment infrastructure plays a significant role in the deceptive layer. While it is common for some application protocols to be open in cloud environments (e.g., Telnet, SSH, MQTT, HTTP), it is peculiar to have protocols like Modbus on the cloud. The cloud infrastructure that includes the containers for simulation, attack database, and the *pcap* file repository are provisioned on the Digital Ocean cloud with configurations to limit the egress traffic. We evaluate RIoTPot in self-hosted lab as well as cloud infrastructure to study the influence of deploying honeypots in different settings.

4.1.4 Geographical distribution. There is research on the impact of deploying honeypots in different geographical locations [45] and we therefore take this into consideration by deploying honeypots on different continents and countries. In particular, our experiments are conducted at four geographical locations, namely: New York City (cloud), Frankfurt (cloud), Singapore (cloud), and Denmark (lab infrastructure) to review region-specific attacks. This way the attack data can be analyzed to discover attacks that are region-specific or potential region-specific malware variants.

4.1.5 Protocol emulation. We study six application protocols, namely: Telnet, SSH, HTTP, MQTT, Modbus, and CoAP. The reason for choosing these protocols is to have a mixed emulation portfolio that includes the most commonly used application protocols in both self-hosted and cloud infrastructure and IoT and ICS environments. The protocols are simulated as both low-interaction in the form of modules and high-interaction in dedicated ephemeral containers. The analysis of attacks provides protocol-specific threats and attack trends resulting from misconfiguration.

4.1.6 Period of study. The evaluation of RIoTPot is performed for three months, both for self-hosted and cloud environments that result in a dataset of a large volume of attack traffic captured on each RIoTPot instance. The attacks gathered over time provide an overview of the attack trends on each protocol simulation. Furthermore, we run the Conpot honeypot to compare the attacks received from RIoTPot. The study was carried out from December 10, 2021 - to March 10, 2022.

4.2 Experimental setup

We intend to deploy RIoTPot in diverse environments, interaction modes, geographical locations, and simulation environments to get a comprehensive view of the attacks. The experiment was distributed across our lab and the cloud infrastructure to facilitate the evaluation and the parameters. We deploy RIoTPot in tertiary interaction modes - low, high, and hybrid interaction for further evaluation. We describe our lab's experimental setup and cloud infrastructure in the following sections.

4.2.1 Lab setup. The experimental setup in our lab is depicted on the Appendix Figure 8. RIoTPot was deployed on three hosts R1 (high-interaction), R2 (low-interaction) and R3 (hybrid-interaction). In addition to RIoTPot, the Conpot [34] honeypot is deployed on host C1 (medium-interaction). All four hosts are connected to the Internet and configured with a public IP address on an unfiltered network. However, the hosts are configured with limited egress traffic to avoid misuse of honeypots. The containers spawned by RIoTPot run as ephemeral instances that are re-spawned periodically to avoid infection spread and recover from availability crashes. The attack data from all the hosts are stored as partitioned, individual, and rotated files on a remote file repository to facilitate further analysis. All the attack traffic is stored in the attack database to facilitate querying and analysis. The attack database and the file repository are provisioned on remote systems to avoid disruption in the logging in situations of system failure. Host R3 operates in a hybrid interaction mode where the SSH, MQTT, Modbus, and CoAP operate in high-interaction mode, and Telnet and HTTP operate in low-interaction modes.

4.2.2 Cloud setup. The experimental setup on the cloud infrastructure is shown in the Appendix Figure 9. Similar to the lab setup, the cloud instances are provisioned on the Digital Ocean as Droplets and has 12 honeypot instances (R4-C4). The 12 honeypot instances are distributed across three geographical locations - New York City, Frankfurt, and Singapore and configured with a public IP address accordingly. Similar to the lab setup, the attack traffic from all the instances is stored as both *Pcap* files and in a database running on dedicated remote systems. The containers are re-spawned periodically and re-provisioned with a static configuration file. The egress traffic from all the containers is limited for potential misuse of the vulnerable environments. The database is provisioned with an elastic model to support the large volumes of attack traffic collected from the honeypot instances. Digital Ocean droplet monitoring helps in tracking the state of the honeypot instances that helps in identifying any failure situations [24].

4.2.3 Summary. Table 2 summarizes the experimental setup of the evaluation. To summarize the evaluation parameters of the longitudinal study described in Section 4.1, we deploy RIoTPot in three interaction levels (low, high, hybrid), two deployment environments (lab, cloud), twelve independent hosts per interaction-level, four geographical locations(Denmark(lab) , New York City, Frankfurt, and Singapore), six application protocol emulations (Telnet, SSH, HTTP, MQTT, Modbus, CoAP), comparison with one honeypot in medium interaction (Conpot) and an evaluation period of three months (10Dec,2021-10Mar,2022).

4.3 Dataset

The traffic received on all the honeypot instances from the study are stored in the attack database and as *pcap* files in the *pcap cloud*. In

Host	**Environment**	**Geo-Location**	**Interaction-level**	**Protocols Emulated**
R1	Lab	Denmark	High	Telnet, SSH, HTTP, MQTT, Modbus, CoAP
R2	Lab	Denmark	Low	Telnet, SSH, HTTP, MQTT, Modbus, CoAP
R3	Lab	Denmark	Hybrid	High - SSH, MQTT, Modbus, CoAP Low - Telnet, HTTP
C1	Lab	Denmark	Medium	Telnet, SSH, HTTP, Modbus, S7
R4	Cloud	New York City	High	Telnet, SSH, HTTP, MQTT, Modbus, CoAP
R5	Cloud	New York City	Low	Telnet, SSH, HTTP, MQTT, Modbus, CoAP
R6	Cloud	New York City	Hybrid	High - SSH, MQTT, Modbus, CoAP Low - Telnet, HTTP
C2	Cloud	New York City	Medium	Telnet, SSH, HTTP, Modbus, S7
R7	Cloud	Frankfurt	High	Telnet, SSH, HTTP, MQTT, Modbus, CoAP
R8	Cloud	Frankfurt	Low	Telnet, SSH, HTTP, MQTT, Modbus, CoAP
R9	Cloud	Frankfurt	Hybrid	High - SSH, MQTT, Modbus, CoAP Low - Telnet, HTTP
C3	Cloud	Frankfurt	Medium	Telnet, SSH, HTTP, Modbus, S7
R10	Cloud	Singapore	High	Telnet, SSH, HTTP, MQTT, Modbus, CoAP
R11	Cloud	Singapore	Low	Telnet, SSH, HTTP, MQTT, Modbus, CoAP
R12	Cloud	Singapore	Hybrid	High - SSH, MQTT, Modbus, CoAP Low - Telnet, HTTP
C4	Cloud	Singapore	Medium	Telnet, SSH, HTTP, Modbus, S7

Table 2: Experimental setup overview

this longitudinal study, we collect data from 12 honeypot instances of RIoTPot and four instances of Conpot over a period of three months. The dataset is a collection of the database dumps and the *pcap* files. The *pcap* files capture the ingress and egress traffic from the honeypot instances. The dataset is segregated based on honeypot instance, protocol, geolocation and interaction-levels. The filtering of the scanning-service from the ingress traffic is performed by labeling the traffic in the database. The labeled dataset of the labeled events of scanning-services can be exported from the attack database. Currently, the dataset is checked for 19 scanning-services. The traffic captured on the *pcap* files are "packet-buffered", so that the output is written to *pcap* file at the end of each packet rather than at the end of each line. The administration traffic is excluded from the *pcap* files and the attack database. The *pcap* files are periodically rotated (daily). The dataset will be provided to academic researchers upon request and following a non disclosure agreement[2].

5 EVALUATION

To provide a comprehensive overview of the findings during the study, we break down the results based on the evaluation parameters. The findings are discussed in the following sections.

5.1 Interaction levels

We discuss our findings based on interaction-levels in the following sections.

5.1.1 Total Events. Figure 2 shows the total number of events on all RIoTPot instances based on interaction levels low, high and hybrid. Compared to the low and hybrid interaction, the high-interaction level received higher events. A total of 10.87 million events were received on all instances, of which 32% (3, 487, 877) were from low, 35% (3, 788, 435) from high, and the remaining 33% (3, 600, 823) from hybrid interaction. The total events are inclusive of the probing traffic received from Internet-wide scanning probes (e.g., [6], [38]).

The Appendix Figure 10 shows the percentage of events received per day by interaction level. From the outset, we see a rise in the events received on the high-interaction level compared to low and hybrid interaction levels. We see sharp differences in the number of events between December 13-15,2021, and February 13-20,2022. A

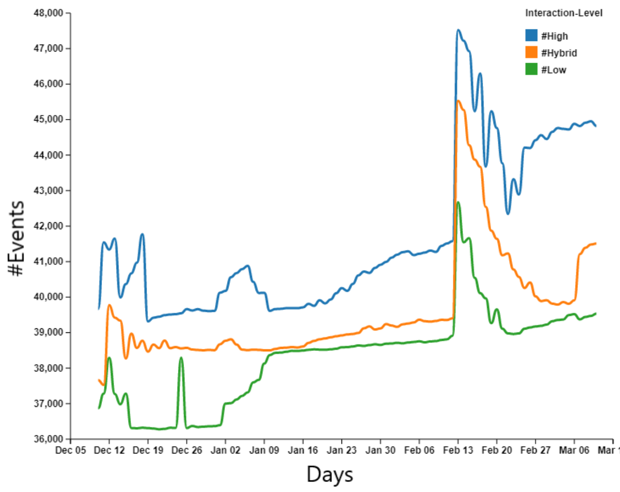

Figure 2: Total events by interaction

possible explanation for such uncertainty could be that the hybrid-interaction level involves both low and high interaction levels on the simulated protocols. We discuss the possible reasons for the deviations in Section 6.3.

5.1.2 Event classification. A total of 10.87 million events were received on all the RIoTPot deployments. Table 3 summarizes the event classification by type and interaction level. Of the total events, 56% of the traffic was identified from Internet scanning-services (e.g., Shodan [38], Censys [6]). We consider the traffic from scanning-services as benign because of the known intent behind their scans. Filtering out benign traffic simplifies the analysis process of focusing on the traffic with malicious intent. A total of 19 unique scanning-services[3] were identified and labeled by RIoTPot's noise filter, which has a database of benign scanning-services. While it is common to observe recurring traffic from scanning-services like Shodan and Censys, some scans occur multiple times per day while other services follow a different pattern that ranges from days to weeks. Note that we simulate six protocols in RIoTPot and some scans target specific port ranges and some target specific ports with custom requests [43]. Lastly, we did not detect any deviations in the received scanning-services traffic on the basis of the interaction.

The traffic that is not labeled as scanning-services is classified as malicious. The malicious classification includes both suspicious traffic and traffic with clear malicious intent in the requests or payloads. The suspicious traffic includes probing traffic from unknown sources and probable backscatter noise. As honeypots do not have any production value, we consider any communication, excluding the aforementioned scanning services, towards them suspicious. 4.8 million events are marked as malicious based on our criteria. Note that the number of malicious events stated here is not unique by attack source. We observe multiple attacks from the same attack source in the traffic volume. Further classification of malicious traffic volume received based-on interaction level is shown in Table

[2]https://doi.org/10.11583/DTU.21088651

[3]https://github.com/aau-network-security/riotpot#12-Noise-Filter

Interaction-level	**Even-type**	**Count**
Low-interaction	Scanning-service	2.02 M
High-interaction	Scanning-service	2.02 M
Hybrid-interaction	Scanning-service	2.02 M
Low-interaction	Malicious	1.46 M
High-interaction	Malicious	1.76 M
Hybrid-interaction	Malicious	1.57 M
Total scanning-services events		**6.07 M**
Total malicious events		**4.8 M**
Total events		**10.87 M**

Table 3: Total events by type and interaction level

3. We observe that the high-interaction instances received higher volume than low and hybrid interaction levels.

5.1.3 Attack sources by interaction-level. Figure 3 shows the number of unique IP addresses identified from the malicious traffic over days and interaction level. We observe a steady increase in the total unique IP addresses over days, with a peak from 13 February 2022 and a subsequent decline for the next four days. Furthermore, we observe that the high-interaction level instances received attacks from more unique sources than the other interaction levels.

Interaction Level	**#Malicious Events**	**#Unique IPs**
High-Interaction	1, 763, 395	18, 431
Hybrid-interaction	1, 575, 807	12, 618
Low-interaction	1, 463, 883	8, 635
Distinct IPs from all interaction levels	22, 518	

Table 4: Summary of malicious events and unique IPs

Table 4 summarizes the distinct cumulative total number of unique source IP addresses by interaction-level. The maximum number of unique IPs were detected on the high-interaction instances. We identify 22, 518 unique IPs across all the malicious events. We want to emphasize that in our study, RIoTPot emulates six protocol services, and the unique attack sources summarized in Table 4 are based on traffic received across these protocol emulations.

5.2 Deployment infrastructure

RIoTPot is deployed on both lab (self-hosted) and cloud infrastructure for evaluation. The lab infrastructure hosts three instances, whereas the cloud infrastructure hosts nine instances of RIoTPot. Figure 4 shows the distribution of malicious events received on RIoTPot instances based on the hosted infrastructure. The number of events on the cloud infrastructure is high due to higher instances of RIoTPot deployed (9 instances) in comparison to the lab (3 instances). Furthermore the figure shows the number of malicious events per instance in lab and cloud infrastructure. We observe that the cloud instances have a higher number of malicious events than the lab instances. This could be because of any suspicious scans or malicious requests that are region-specific.

In Figure 4 the number of malicious events per interaction level on the lab and cloud infrastructure is summarized. Although there are deviations in the traffic volume, the number of malicious events across all interaction levels is increasing over time. The high interaction instance received more attacks than low and hybrid interaction levels. Note that the number of malicious events on the cloud is higher than in the lab, as there are more instances of RIoTPot deployed on the cloud infrastructure compared to the lab environment. We observed minor variations in the trend of malicious events in both operating environments.

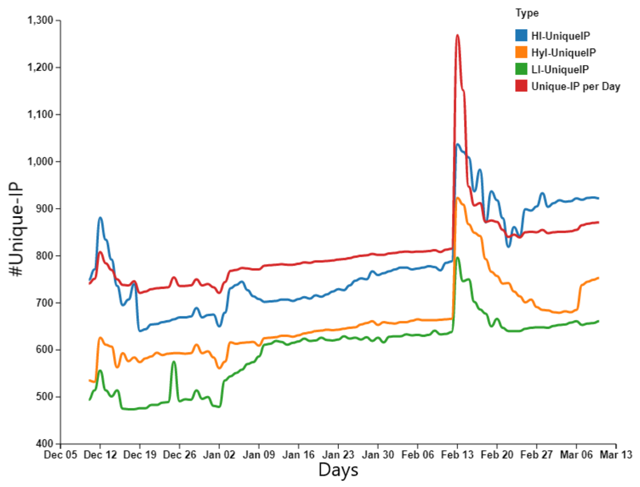

Figure 3: Unique-IPs over day and interaction

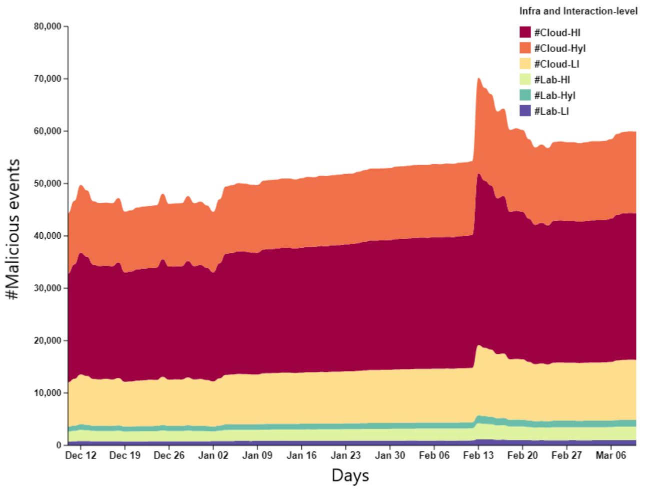

Figure 4: Comparison of total malicious events by infrastructure and interaction

5.3 Geographical location

To study the influence of geo-location and region-specific attack distribution, we deploy RIoTPot in four locations - New York City, Frankfurt, Singapore, and Denmark (Lab). Figure 5 shows the distribution of malicious events from each location and interaction levels. The interaction levels are color-coded, and the solid sphere represents the number of daily events by interaction. The sphere's radius is proportional to the number of events denoted in the legend.

The lowest number of attacks received per interaction per day is 743, while the highest is 13, 287. Compared with the cloud instances, the lab instances received significantly lower malicious events. Initially, the New York instances received higher traffic; however, the Frankfurt instances received the highest traffic overall. The instances deployed in Singapore reported the lowest traffic compared to the other cloud deployments for the whole duration of the evaluation. We observed suspicious events specific to the region that were not seen in other cloud instances. The suspicious events consisted of port scans, brute-force attempts, and attacks specific to protocols emulated by RIoTPot. We find region-specific benign scans from known entities like educational institutions and government-aided organizations other than the malicious events. In the Appendix Figure 14 we further discuss how the location and the cloud vs. lab deployment is connected to the number of attacks.

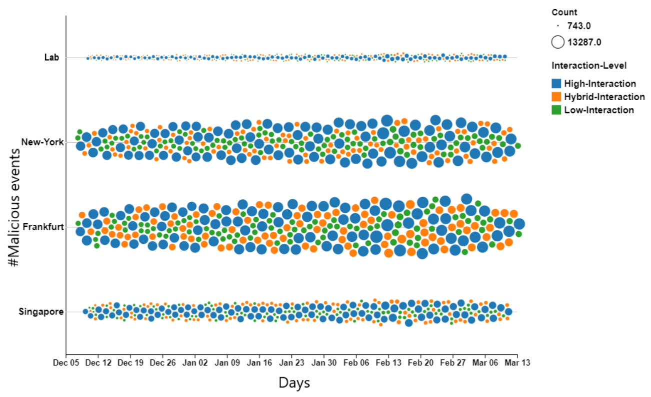

Figure 5: Distribution of malicious events across lab and cloud deployments

5.4 Emulated protocols

RIoTPot emulates six protocols - Telnet, SSH, HTTP, Modbus, MQTT, and CoAP. The protocols are emulated in diverse interaction levels across the deployments. Table 2 summarizes the interaction level of the individual protocols emulated on the instances. Figure 6 shows the number of malicious events recorded on each protocol. We observed the highest number of events on the SSH protocol, followed by HTTP, Telnet, MQTT, Modbus, and CoAP. Note that the number of events is not unique per source IP and is the total count of the events observed across all interaction levels.

Appendix Figure 13 summarizes the malicious events received per protocol by interaction level. We observe that the highest number of malicious events in all protocol emulations are received on the high-interaction instances. In protocols like Telnet, SSH, MQTT, and Modbus, we observed a gradual decrease in the number of events on the low-interaction instances. Many attack types like brute-force, poisoning, pivoting and reflection attacks were observed. The attack types are discussed further in Section 6.1.

5.5 Comparison with Conpot

To compare the attacks received on RIoTPot instances, we deploy Conpot [34], a medium-interaction honeypot that can emulate the SSH, Telnet, HTTP, Modbus, and S7 protocols. We deploy Conpot

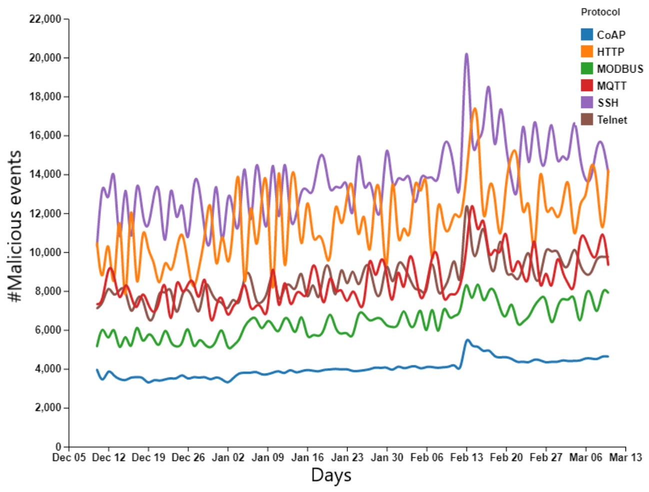

Figure 6: Total malicious events by protocol

on both the lab and cloud infrastructure, including the geo-locations at which RIoTPot instances are deployed. Appendix Figure 11 shows the comparison of the malicious events received on RIoTPot and Conpot instances by interaction-level, protocol, and hosted infrastructure. The figure lists the deployed instances (see Table 2) and the total malicious events received per instance. The Conpot instances simulated four protocols (Telnet, SSH, HTTP, and Modbus) that can be compared with the protocols emulated by RIoTPot instances. In comparison to Conpot, we observe that RIoTPot received a higher number of events on the high and hybrid interaction instances and a similar number of events with low interaction-level instances. The figure also compares the number of malicious events observed on each RIoTPot instance deployed by hosted infrastructure, location interaction-level, and emulated protocols. We observe that the instances deployed on the Frankfurt cloud infrastructure (R7) received the highest number of malicious events. The instances R1,R2,R3,C1 were deployed in the lab; R4,R5,R6,C2 in New York City; R7,R8,R9,C3 in Frankfurt and R10,R11,R12,C4 in Singapore. We suspect that the difference in the number of malicious events between Conpot and RIoTPot could be as a result of limited interaction capabilities of Conpot in comparison to RIoTPot.

6 DISCUSSION

This section discusses the attack types observed during the evaluation process and our findings on the varied malicious events received based on the evaluation parameters. We further state the limitations in our approach, and the ethical considerations followed in our methodology.

6.1 Malicious events

We received a total of 4.8 million malicious events on all the instances. Note that all the events that are not labeled as scanning-services are classified as malicious. The malicious events further include traffic with malicious intent in the requests or payloads. We observe diverse attack types in the malicious traffic received on all instances. The attack types and the exclusive attacks observed during the evaluation are discussed in the following sections.

6.1.1 *Attack type by interaction-level.*

Figure 7 shows an overview of the attack types observed during our evaluation by percentage and interaction level. We observe attack types like brute-force, poisoning, reflection, and portscans from unknown scanners. The brute-force attacks were the most common observed attacks across all instances and targeted all the emulated protocols. The emulated protocols were configured with weak access controls and credentials to capture advanced attack types. A persistent volume of brute-force attacks was observed at all interaction levels. Furthermore, we see brute-force attacks from the same actors (IPs) across all the interaction levels. However, some regional attacks were observed in specific instances where the attack source appeared to be from the same continent. The poisoning attacks focused on modifying data following unauthorized access. For example, we discover messages on the CoAP protocol to modify data. A larger volume of poisoning attacks was observed on the high-interaction level. In addition, we observe that the attacks from the same attack source interrupted the connection on low-interaction instances while pursuing the connection on high-interaction instances. With this, we entail that the threat actors use specific information from the sessions to determine the pursuit of the attack.

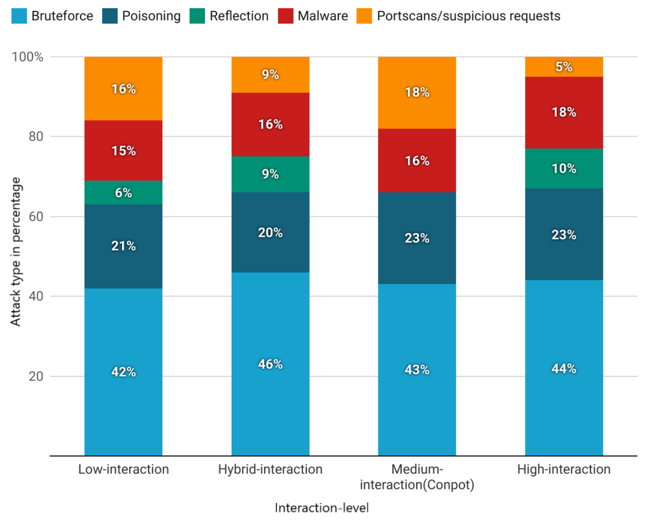

Figure 7: Attack types by interaction

Reflection attacks were detected on the CoAP protocol. We identified reflection attacks when the destination address port is port 80 and the source port is 5832. A larger volume of reflection attacks was again observed in the high-interaction instances. However, we acknowledge that the reflection attacks may be a part of backscatter traffic. We observe malware injection attacks where an attacker tries to download malware from malicious links. The malicious links are identified by analyzing the attacks logged in the *pcap* files. Upon finding a suspicious link in the payload, we check the link with Virustotal [49] to determine the maliciousness. We observe multiple variants of the Mirai [2] malware along with LuaBot, Mozi [19] and BrickerBot, among others. Lastly, we observed a specific volume of non-recurring portscanning traffic that was not identical to known scanning-services or attack types. We group such kind of traffic under portscans and suspicious requests.

6.1.2 *Attack type by protocol.*

Appendix Figure 12 shows the attack types in percentage received by emulated protocols. The attack traffic was stored both as pcap files and the session logs in the attack database. We summarize the attack types found on each emulated protocol.

Telnet and SSH. The Telnet and the SSH protocols received high volumes of brute-force attacks. The Telnet protocol further received certain malware injections on successful brute force attempts. Upon checking with Virustotal, the malware links observed on the Telnet attacks were detected to be variants of the Mirai family. Similarly, several variants of Mirai were detected on the SSH protocol. In addition to the trivial brute-force attacks and malware injections, many port scans were observed. This entails that there are still many actors looking for vulnerable Telnet and SSH instances.

HTTP. The HTTP protocol emulated a static login page for a Siemens LOGO 230 RCEo Modbus controller. The protocol received a large number of brute-force attempts. In addition to the brute-force attacks, the HTTP protocol received many Log4j attacks, although the Apache webserver [31] was used for the emulation. The attacks tried injection attacks through RMI (Remote Method Invocation) calls from remote servers. Lastly, a large volume of web scrapers was identified along with unknown scanning services.

MQTT. The MQTT protocol was emulated using the Eclipse-Mosquitto [30] image for the high-interaction and the library for the low-interaction. Although the protocol was configured to allow anonymous logins, there was a large volume of brute-force attacks. Moreover, several data poisoning attempts were detected where the attackers tried to modify data in the queues. In addition, we detected that some attacks created new topics and tried accessing the *SYS$* topic specifically. The protocol was scanned mainly by benign scanning services, however, some regional suspicious scans were detected in the instances deployed in Frankfurt and Singapore.

Modbus. The Modbus protocol received a large volume of poisoning attacks to read and modify the data from the registers and coils. The attacks were observed to target three function codes of the nineteen available to fetch information on the device, the reporting server, and the holding register. Furthermore, we observe that most of the attacks used invalid function codes to access the data in the registers. This entails that the scans search for a device with specific function codes for a known exploit.

CoAP. The CoAP protocol was configured to serve on UDP port 5683. Many brute-force attacks were detected that tried to access the CoAP service. Furthermore, we identify data poisoning attacks aimed at modifying values through publishing messages. In addition to the data poisoning attacks, the CoAP protocol saw reflection flooding attacks where the attackers tried to spoof the source packets to divert all the response traffic to a victim. Such attacks were identified by observing the destination port. We observed 27 victim IPs, of which 12 of them were located in Brazil, 4 in South Korea, 4 in Russia, 3 in China, 1 in France and 1 Germany. However, we found that the victim IPs did not have a valid domain and served blank HTML pages.

6.1.3 Region-specific attacks. RIoTPot instances were deployed in four geo-locations. We detect attacks and attack sources of the targeted instances in specific regions. Appendix Table 7 lists the attack type and volume in percentage of the source of malicious traffic, observed exclusively in specific regions. We observe attacks from specific attack sources on several protocols and attack types. The unique source IPs listed in the table denote the source of attacks that targeted that region exclusively. While Internet scanning-services are known to use regional hosts for scanning specific locations, the unique IPs listed in Table 7 are from malicious events. To filter our results, we check if the IPs are Tor relays [32] or from a VPN [14] and find that they are neither. We check the IPs sources on Internet scanning-services like Shodan [38] to find that the hosts had SSH ports open. Furthermore, upon looking up the IPs history, we find that they were recently moved from an ASN.

6.2 Attack sources

A total of 22, 518 unique IPs were identified from the malicious events. To get an understanding of the attack sources, we try to identify the attack sources using banner-based fingerprinting techniques. We send connection attempts on Telnet (port 23) and HTTP requests on port 80, 8080 and 443 on the IPs by using Lift[4], an open-source low-impact fingerprinting tool. We then check the banners and the response for potential identifiers for the attack sources. We take care to send a minimal number of packets in our probes to limit the traffic with the attack sources. Table 5 shows the device types identified by the banner grabbing checks. A total of 5264 (23%) devices were identified through the banner checks. We suspect that these are compromised devices that are causing attacks on the Internet. In addition to the infected devices, we notice that a vast majority of the HTTP response contained default test websites from Apache, NGinX and Tengine web servers. A total of 4218 (19%) of such responses were identified. We perform a reverse-DNS lookup to identify if there were any domains associated with the IP address ranges and found 21 domains. The domains were associated with some generic top-level domains that include *.art* (5), *.games* (6), *.love* (3), *.website* (2) and *.webcam* (5). Lastly, the rest (58%) of the devices could not be determined.

Device type	Protocol	Count
Router	HTTP	1819
DVR	HTTP	1621
Router	Telnet	721
IP Phone	HTTP	311
Switch	HTTP	287
Switch	Telnet	211
IP Printer	HTTP	176
NAS	HTTP	118
Total		**5264**

Table 5: Attack-source types

6.3 Impact of interaction-levels in honeypots

Our evaluation and findings reveal that the attacks on low-interaction honeypots decreased gradually for some protocols (see Figure 10), while the high-interaction instances received a higher number of attack events. Hence, our results suggest that the interaction levels play an essential role in attracting specific attacks. Our observation of a gradual decrease in non-recurring scanning probes indicates that modern scanning probes may have checks that help in characterizing if the scanned system is a honeypot[5]. The malicious events received on the hybrid-interaction model reveal that a combination of low and high-interaction emulation indeed attracts more attacks and successfully deceives the checks from suspicious scanning probes. To summarize the impact of interaction levels, low-interaction honeypots are still effective in capturing scanning and bot traffic. However, we suggest that deploying high-interaction honeypots with limited network configuration on some protocols is more effective to achieve a higher deception layer.

6.4 Limitations

We acknowledge the following limitations in our approach. First, the lab infrastructure is limited to one location, while the cloud deployments range to three locations. This limitation causes an uneven comparison directly between the lab and the cloud deployments. The comparison between the instances deployed in the lab and cloud would be descriptive if the number of deployed instances are the same. Second, we deploy RIoTPot in four cities limiting the scope to three continents. Deploying RIoTPot in all continents would provide a broader perspective of the attack landscape. Third, we limit the number of emulated protocols to six. We acknowledge that more protocols would provide us with an extensive dataset for analysis. However, this work aims to visualize the impact of many evaluation and design parameters that can affect the purpose of honeypots. Fourth, we consider each event as a connection and this entails some limitations in terms of over-counting. As the connection terms vary across protocols, we generalize counting by events and not as connections. Lastly, the dataset does not group the attack data as Netflow formats. Storing the data as Netflow formats facilitates wider integration possibilities with analysis platforms.

7 CONCLUSION

In this work, we extend RIoTPot, a modular and hybrid-interaction honeypot and facilitate a longitudinal study. To ascertain the impact of parameters like the interaction levels of honeypots, we perform an extensive longitudinal evaluation of RIoTPot by measuring the malicious events gathered based on parameters like interaction level, deployed infrastructure, geographical location, and emulated protocols. Our results indicate that these parameters are essential in honeypot studies and can provide a broader overview of the attack landscape. The results suggest that high-interaction honeypots receive more sophisticated attacks in comparison with the low-interaction honeypots. Moreover, we observe attacks specific to the hosting environment and geo-location. Compared with Conpot, RIoTPot's high interaction instances received a higher volume of malicious events on all evaluation parameters. We observe diverse attacks like reflection, data poisoning and malware on the honeypots. Lastly, we observe large volumes of traffic from scanning-services that may cause alert fatigue and are false positives.

[4]https://github.com/trylinux/lift

[5]In fact, services like Shodan already have such capabilities [37]

ACKNOWLEDGMENTS

This research was supported as part of COM^3, an Interreg project supported by the North Sea Programme of the European Regional Development Fund of the European Union.

REFERENCES

[1] P Dilsheer Ali and T. Gireesh Kumar. 2017. Malware capturing and detection in dionaea honeypot. In *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*. IEEE, Vellore, India, 1–5. https://doi.org/10.1109/IPACT.2017.8245158

[2] Manos Antonakakis, Tim April, Michael Bailey, Matt Bernhard, Elie Bursztein, Jaime Cochran, Zakir Durumeric, J. Alex Halderman, Luca Invernizzi, Michalis Kallitsis, Deepak Kumar, Chaz Lever, Zane Ma, Joshua Mason, Damian Menscher, Chad Seaman, Nick Sullivan, Kurt Thomas, and Yi Zhou. 2017. Understanding the Mirai Botnet. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC, 1093–1110. https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[3] Timothy Barron and Nick Nikiforakis. 2017. Picky Attackers: Quantifying the Role of System Properties on Intruder Behavior. In *Proceedings of the 33rd Annual Computer Security Applications Conference* (Orlando, FL, USA) *(ACSAC '17)*. Association for Computing Machinery, New York, NY, USA, 387–398. https://doi.org/10.1145/3134600.3134614

[4] Busybox. 2022. *Busybox DockerHub*. BusyBox. https://hub.docker.com/_/busybox

[5] Warren Z. Cabral, Craig Valli, Leslie F. Sikos, and Samuel G. Wakeling. 2021. Analysis of Conpot and Its BACnet Features for Cyber-Deception. In *Advances in Security, Networks, and Internet of Things*, Kevin Daimi, Hamid R. Arabnia, Leonidas Deligiannidis, Min-Shiang Hwang, and Fernando G. Tinetti (Eds.). Springer International Publishing, Cham, 329–339.

[6] Censys. 2021. Censys Search. Retrieved June 28, 2021 from https://censys.io/

[7] Fan Dang, Zhenhua Li, Yunhao Liu, Ennan Zhai, Qi Alfred Chen, Tianyin Xu, Yan Chen, and Jingyu Yang. 2019. Understanding Fileless Attacks on Linux-Based IoT Devices with HoneyCloud. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services* (Seoul, Republic of Korea) *(MobiSys '19)*. Association for Computing Machinery, New York, NY, USA, 482–493. https://doi.org/10.1145/3307334.3326083

[8] Docker. 2022. *DockerHub*. Docker. https://hub.docker.com/

[9] Michael Dodson, Alastair R. Beresford, and Mikael Vingaard. 2020. Using Global Honeypot Networks to Detect Targeted ICS Attacks. In *2020 12th International Conference on Cyber Conflict (CyCon)*, Vol. 1300. IEEE, Estonia, 275–291. https://doi.org/10.23919/CyCon49761.2020.9131734

[10] ENISA. 2020. *ENISA Threat Landscape 2020 - Malware*. ENISA. https://www.enisa.europa.eu/publications/malware

[11] Golang. 2021. Go Language. Retrieved March 16, 2022 from https://golang.org/

[12] GreyNoise. 2022. GreyNoise. https://viz.greynoise.io/

[13] Juan David Guarnizo, Amit Tambe, Suman Sankar Bhunia, Martin Ochoa, Nils Ole Tippenhauer, Asaf Shabtai, and Yuval Elovici. 2017. SIPHON: Towards Scalable High-Interaction Physical Honeypots. In *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security* (Abu Dhabi, United Arab Emirates) *(CPSS '17)*. Association for Computing Machinery, New York, NY, USA, 57–68. https://doi.org/10.1145/3055186.3055192

[14] iphub. 2022. *iphub*. iphub. https://iphub.info/

[15] Celine Irvene, David Formby, Samuel Litchfield, and Raheem Beyah. 2018. HoneyBot: A Honeypot for Robotic Systems. *Proc. IEEE* 106, 1 (2018), 61–70. https://doi.org/10.1109/JPROC.2017.2748421

[16] Xingbin Jiang, Michele Lora, and Sudipta Chattopadhyay. 2020. An Experimental Analysis of Security Vulnerabilities in Industrial IoT Devices. *ACM Trans. Internet Technol.* 20, 2, Article 16 (may 2020), 24 pages. https://doi.org/10.1145/3379542

[17] Linuxserver.io. 2022. *Openssh DockerHub*. OpenSSH. https://hub.docker.com/r/linuxserver/openssh-serve

[18] Samuel Litchfield, David Formby, Jonathan Rogers, Sakis Meliopoulos, and Raheem Beyah. 2016. Poster: Re-thinking the honeypot for cyber-physical systems. In *Poster at IEEE Symposium on Security and Privacy*. IEEE, San Jose, California.

[19] Microsoft. 2021. *The Mozi Botnet*. Microsoft. https://www.microsoft.com/security/blog/2021/08/19/how-to-proactively-defend-against-mozi-iot-botnet/

[20] Thomas Miller, Alexander Staves, Sam Maesschalck, Miriam Sturdee, and Benjamin Green. 2021. Looking Back to Look Forward: Lessons Learnt from Cyber-Attacks on Industrial Control Systems. *Int. J. Crit. Infrastruct. Prot.* 35, C (dec 2021), 14 pages. https://doi.org/10.1016/j.ijcip.2021.100464

[21] Mawal Mohammed, Mahmoud Elish, and Abdallah Qusef. 2016. Empirical insight into the context of design patterns: Modularity analysis. In *2016 7th International Conference on Computer Science and Information Technology (CSIT)*. IEEE, Amman, Jordan, 1–6. https://doi.org/10.1109/CSIT.2016.7549474

[22] Lisa O. Monaco. 2021. *DAG Monaco Delivers Remarks at Press Conference on Darkside Attack on Colonial Pipeline*. The United States Department of Justice. https://www.justice.gov/opa/speech/dag-monaco-delivers-remarks-press-conference-darkside-attack-colonial-pipeline

[23] Shun Morishita, Takuya Hoizumi, Wataru Ueno, Rui Tanabe, Carlos Gañán, Michel JG van Eeten, Katsunari Yoshioka, and Tsutomu Matsumoto. 2019. Detect me if you… oh wait. An internet-wide view of self-revealing honeypots. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, IEEE, Washington DC, USA, 134–143.

[24] Digital Ocean. 2022. Digital Ocean Droplet monitoring. Retrieved March 16, 2022 from https://docs.digitalocean.com/products/monitoring/

[25] OITC. 2022. *Modbus-server DockerHub*. OITC. https://hub.docker.com/r/oitc/modbus-server

[26] Michel Oosterhof. 2016. Cowrie SSH/telnet honeypot. https://github.com/micheloosterhof/cowrie

[27] Yin Minn Pa Pa, Shogo Suzuki, Katsunari Yoshioka, Tsutomu Matsumoto, Takahiro Kasama, and Christian Rossow. 2015. IoTPOT: Analysing the Rise of IoT Compromises. In *9th USENIX Workshop on Offensive Technologies (WOOT 15)*. USENIX Association, Washington, D.C. https://www.usenix.org/conference/woot15/workshop-program/presentation/pa

[28] plgd. 2022. *CoAP-Gateway*. plgd. https://hub.docker.com/r/plgd/coap-gateway

[29] Deutsche Telekom AG Honeypot Project. 2022. T-Pot: A Multi-Honeypot Platform.

[30] Eclipse Project. 2022. *Eclipse Mosquitto DockerHub*. Eclipse Project. https://hub.docker.com/_/eclipse-mosquitto

[31] The Apache HTTP Server Project. 2022. *HTTPD DockerHub*. The Apache Project. https://hub.docker.com/_/httpd

[32] The Tor Project. 2022. *ExoneraTor*. The Tor Project. https://metrics.torproject.org/exonerator.html

[33] L Rist. 2009. Glastopf project.

[34] Lukas Rist, Johnny Vestergaard, Daniel Haslinger, A Pasquale, and J Smith. 2013. Conpot ics/scada honeypot.

[35] Frances Robles and Nicole Perlroth. 2021. *'Dangerous Stuff': Hackers Tried to Poison Water Supply of Florida Town*. The New York Times. https://www.nytimes.com/2021/02/08/us/oldsmar-florida-water-supply-hack.html

[36] Stewart Sentanoe, Benjamin Taubmann, and Hans P. Reiser. 2018. Sarracenia: Enhancing the Performance and Stealthiness of SSH Honeypots Using Virtual Machine Introspection. In *Secure IT Systems*, Nils Gruschka (Ed.). Springer International Publishing, Cham, 255–271.

[37] SHODAN. 2022. Honeypot or Not? https://honeyscore.shodan.io

[38] SHODAN. 2022. Shodan. https://www.shodan.io/

[39] Shreyas Srinivasa, Jens Myrup Pedersen, and Emmanouil Vasilomanolakis. 2021. Gotta catch 'em all: a Multistage Framework for honeypot fingerprinting. arXiv:2109.10652 [cs.CR]

[40] Shreyas Srinivasa, Jens Myrup Pedersen, and Emmanouil Vasilomanolakis. 2021. *Open for Hire: Attack Trends and Misconfiguration Pitfalls of IoT Devices*. Association for Computing Machinery, New York, NY, USA, 195–215. https://doi.org/10.1145/3487552.3487833

[41] Shreyas Srinivasa, Jens Myrup Pedersen, and Emmanouil Vasilomanolakis. 2021. RIoTPot: a modular hybrid-interaction IoT/OT honeypot. In *26th European Symposium on Research in Computer Security (ESORICS) 2021*. Springer, Springer, Darmstadt, Germany.

[42] Dino Tools. 2010. Web Honeypot. https://github.com/DinoTools/dionaea/

[43] R Trapkickin. 2015. Who is scanning the Internet?

[44] Veronica Valeros and Sebastian Garcia. 2022. Hornet 40: Network dataset of geographically placed honeypots. *Data in Brief* 40 (2022), 107795. https://doi.org/10.1016/j.dib.2022.107795

[45] Veronica Valeros and Sebastian Garcia. 2022. Hornet 40: Network dataset of geographically placed honeypots. *Data in Brief* 40 (2022), 107795. https://doi.org/10.1016/j.dib.2022.107795

[46] Emmanouil Vasilomanolakis, Shankar Karuppayah, Max Mühlhäuser, and Mathias Fischer. 2014. HosTaGe: A Mobile Honeypot for Collaborative Defense. In *Proceedings of the 7th International Conference on Security of Information and Networks* (Glasgow, Scotland, UK) *(SIN '14)*. Association for Computing Machinery, New York, NY, USA, 330–333. https://doi.org/10.1145/2659651.2659663

[47] Alexander Vetterl and Richard Clayton. 2018. Bitter Harvest: Systematically Fingerprinting Low- and Medium-interaction Honeypots at Internet Scale. In *12th USENIX Workshop on Offensive Technologies (WOOT 18)*. USENIX Association, Baltimore, MD, 9. https://www.usenix.org/conference/woot18/presentation/vetterl

[48] Alexander Vetterl and Richard Clayton. 2019. Honware: A Virtual Honeypot Framework for Capturing CPE and IoT Zero Days. In *2019 APWG Symposium on Electronic Crime Research (eCrime)*. IEEE, Pittsburgh, PA, USA, 1–13. https://doi.org/10.1109/eCrime47957.2019.9037501

[49] Virustotal. 2022. Virustotal. https://www.virustotal.com

[50] Jianxin Wang, Ming K. Lim, Chao Wang, and Ming-Lang Tseng. 2021. The evolution of the Internet of Things (IoT) over the past 20 years. *Computers & Industrial Engineering* 155 (2021), 107174. https://doi.org/10.1016/j.cie.2021.107174

[51] Andy Wick and community. 2022. *Arkime*. Arkime. https://arkime.com/index#home/

[52] Armin Ziaie Tabari and Xinming Ou. 2020. *A Multi-Phased Multi-Faceted IoT Honeypot Ecosystem.* Association for Computing Machinery, New York, NY, USA, 2121–2123. https://doi.org/10.1145/3372297.3420023

A QUALITATIVE COMPARISON

Table 6 provides an overview of the qualitative comparison of honeypots proposed in related work. The honeypots are compared based on their source-code availability, supported protocols, interaction level, operational environments and known fingerprinting techniques. Most of the proposed honeypots are available as open source and support multiple protocols. However, we observe that there are no high-interaction honeypots are available as open source.

Honeypot	Opensource	Supported protocols	Interaction levels	Virtual vs. Hardware	Known fingerprinting methods
Conpot[34]	Yes	SSH, Telnet, Modbus, BACNet, HTTP	medium	virtual	Yes
Cowrie[26]	Yes	SSH, Telnet	medium	virtual	Yes
Glastopf[33]	Yes	HTTP, HTTPS	medium	virtual	Yes
IoTPot[27]	Yes	No	low	hardware	No
Dionaea[42]	Yes	Yes	medium	virtual	Yes
Honware[48]	No	image based	high	virtual	No
RIoTPot[41]	Yes	image based	low, hybrid. high	virtual	No

Table 6: qualitative comparison of honeypots

B APPENDIX: EXPERIMENT OVERVIEW

B.1 Lab setup

The lab setup of our experimental setup is shown in Figure 8. Three instances of RIoTPot R1,R2,R3 and one instance of Conpot C1 were deployed and assigned a public IP each. All the traffic received and sent from the honeypots are stored in a remote file system as a repository in addition to storing the session parameters in the attack database.

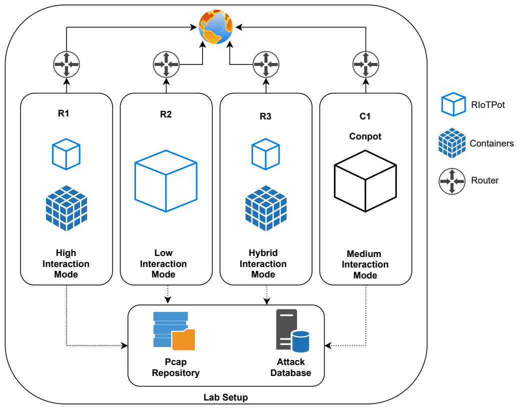

Figure 8: RIoTPot evaluation - lab setup

B.2 Cloud setup

The cloud setup of the methodology is shown in Figure 9. The cloud instances are provisioned at three geographical nodes, Frankfurt, New York city and Singapore.

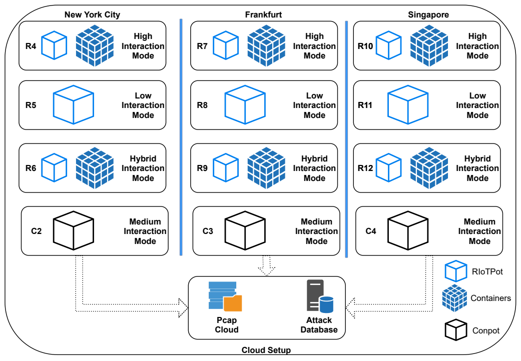

Figure 9: RIoTPot evaluation - cloud setup

C APPENDIX: SUPPLEMENTARY RESULTS

C.1 Percentage of events by interaction-level

Figure 10 shows the percentage of daily events received on RIoTPot instances based on interaction level. We observe a sharp decrease in the percentage of events over time on the low interaction when compared with events on high and hybrid interaction instances. We suspect that this could be because of limited interaction levels.

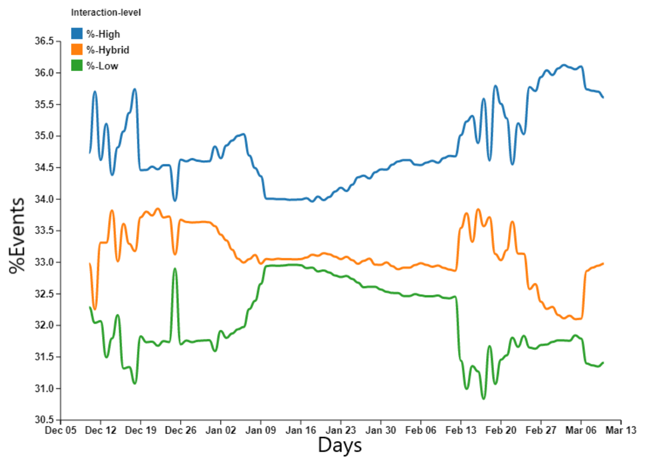

Figure 10: Percentage of events by interaction-level and percentage

C.2 Comparison by interaction-level, location, honeypot and emulated protocols

Figure 11 shows the comparison of the number of attacks received by honeypot instance of RIoTPot(R) and emulated protocols with Conpot(C). We observe a high number of malicious events on the high interaction instances of RIoTPot in comparison to the other deployments.

C.3 Attack Types by Protocol

Figure 12 shows the percentage of attacks types on the emulated protocols. We observe multiple attack types that include bruteforce,

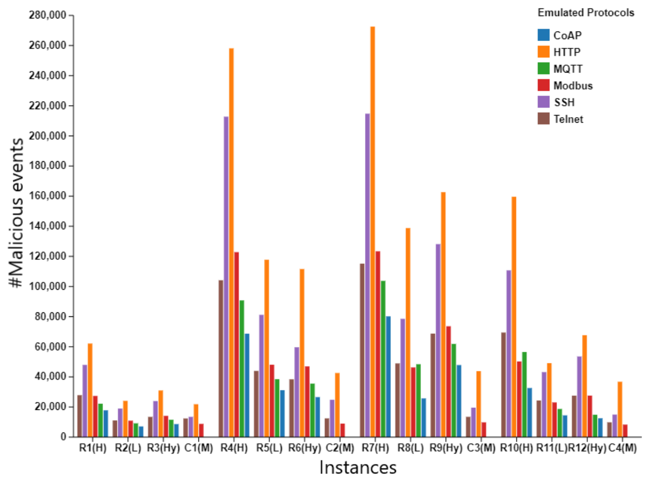

Figure 11: Total malicious events by instances

poisoning, reflection, malware and portscans. Attacks like brute-force and portscans are observed across all protocol emulations and attacks like malware are observed with Telnet and SSH emulations.

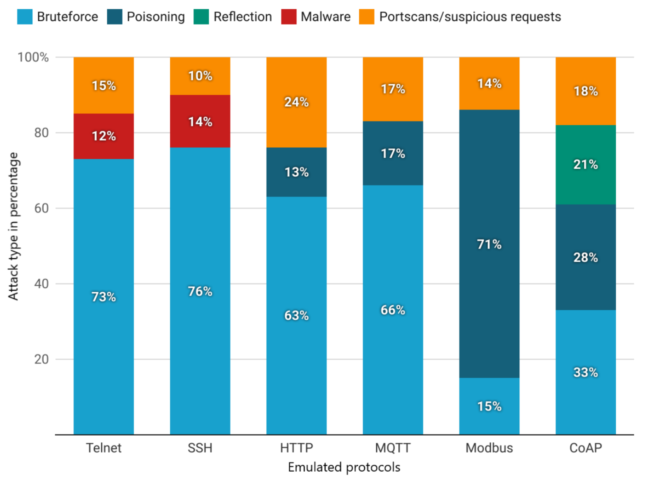

Figure 12: Attack types in percentage by emulated protocols

C.4 Malicious events received per protocol by interaction level

Figure 13 summarizes the malicious events received per protocol by interaction level. We observe that the highest number of malicious events in all protocol emulations are received on the high-interaction instances. In protocols like Telnet, SSH, MQTT, and Modbus, we observed a gradual decrease in the number of events on the low-interaction instances. Many attack types like brute-force, poisoning, pivoting and reflection attacks (CoAP) were observed.

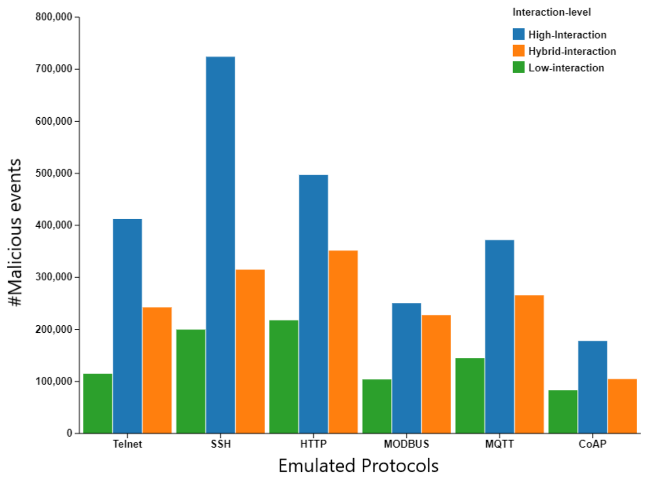

Figure 13: Total malicious events by protocol and interaction

C.5 Attacks and geographical distribution

Figure 14 shows the aggregation of the maximum and the minimum number of malicious events obtained per interaction and city. The instances in Frankfurt city recorded the maximum number of events on each interaction level, while the lowest number of events were recorded at the lab infrastructure daily. The high-interaction instances received more malicious events, regardless of infrastructure or location, followed by the events on hybrid-interaction instances.

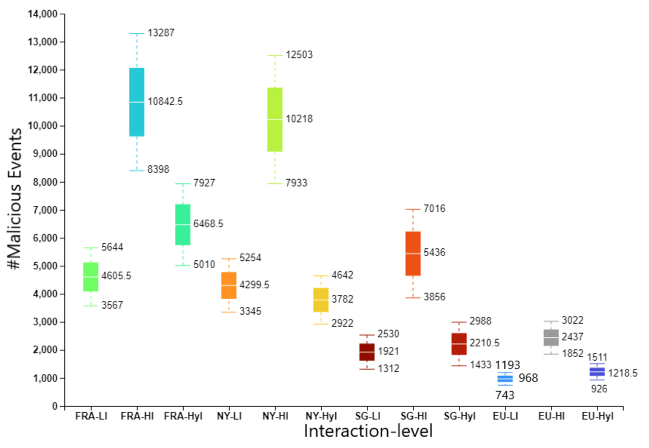

Figure 14: Total malicious events by interaction-level and city: lowest, average and highest per day

C.6 Region specific attack types

Table 7 lists the attack type and volume in percentage of the source of malicious traffic, observed exclusively in specific regions.

C.7 Multistage attacks

Among the attack types discussed above, we observe multistage attacks on the instances. Multistage attacks are attacks that are

Instance	Region	Attack-type	Protocol	Unique attacker IP	Volume
R1	Denmark(lab)	Brute-force	Telnet	19	7%
R4	New-York	Brute-force	SSH	36	11%
R7	Frankfurt	Brute-force	Telnet	27	14%
R10	Singapore	Brute-force	Modbus	7	14%
R5	New-York	Brute-force	HTTP	33	17%
R7	Frankfurt	Poisoning	MQTT	21	18%
R10	Singapore	Poisoning	MQTT	13	12%
R10	Singapore	Reflection	CoAP	6	16%

Table 7: Summary of region-specific attack types

from the same adversary and sequentially target multiple protocols emulated on the target system. A total of 4786 multistage attacks were detected across all the RIoTPot deployments. Figure 15 shows the protocols targeted sequentially by adversaries. The start node denotes the protocol first attacked, and the nodes step-2 and step-3 denote sequential attacks on the other protocols carried out by the same adversary. The numbers below the protocols denote the volume of requests received on the protocols used in the attack. Although such behavior is typical in scanning-services, in this case, the attacks are classified as multistage attacks exclusively when malicious content is observed in the requests. A majority of the requests start from the Telnet and SSH protocols. The MQTT protocol is observed to have received the highest volume of subsequent attacks.

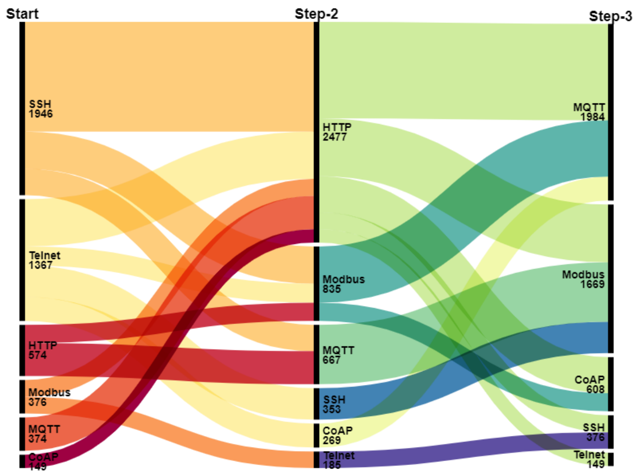

Figure 15: Multistage attacks

D LABELING BENIGN TRAFFIC

RIoTPot has a database of known Internet-wide scanning services. However it is currently limited to 19 services and thus it may be missing benign services. To further classify the unique source IPs identified in our dataset, we check them with Greynoise API [12]. Greynoise provides a classification of suspicious IPs whether they are benign, malicious or unknown. Figure 16 shows the classification as retrieved from Greynoise. Upon correlating the classification from Greynoise to the IPs from which malicious traffic was observed on our honeypot instances, we find that all the IPs were either classified as malicious or unknown from Greynoise.

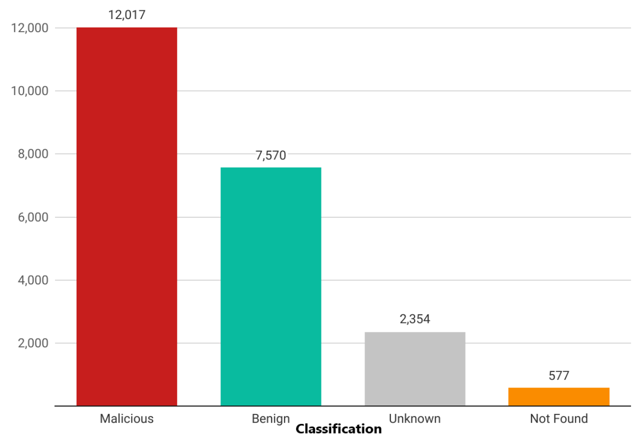

Figure 16: Greynoise classification

E APPENDIX: SUPPLEMENTARY DISCUSSION

E.1 Ethical Considerations

We deploy 12 instances of RIoTPot in varied interaction levels. As honeypots are configured to appear as vulnerable systems, they are prone to be used for causing attacks on the Internet. We configure egress rules on all our deployments to limit the traffic leaving our instances to prevent such misuse. In addition, to avoid the infection spread by any malware attacks, we use ephemeral container instances for our honeypot deployments. New instances are spawned regularly to avoid the spread of any infections.