Next-Generation Industrial Control System (ICS) Security

Towards ICS Honeypots for Defence-in-Depth Security

Sam Maesschalck

School of Computing and Communications
Lancaster University

This thesis is submitted for the degree of
Doctor of Philosophy

August 2023

“As cyber threats evolve, we need to evolve as well.”

- Christopher A. Wray

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text, acknowledgements and publications list.

Some ideas within this thesis were the product of discussions with my supervisors Vasileios Giotsas, Nicholas Race and Benjamin Green, my colleagues at Lancaster University, and colleagues within the NG-CDI project.

Sam Maesschalck
August 2023

Acknowledgements

Completing this PhD thesis was made possible not just by my personal effort but also by the significant contributions and steady support from many.

Firstly, my deepest gratitude goes to my principal supervisor, Dr Vasileios Giotsas, for his guidance throughout this journey. His expertise and tireless dedication have been instrumental in steering this research, and his patience and insightful advice have continually inspired me to strive for excellence. Equally, I would like to extend my sincere appreciation to Professor Nicholas Race for his invaluable mentorship and support. His astute observations and incisive feedback have significantly enriched this work, fostering both its development and my growth as a researcher.

I would also like to extend my heartfelt appreciation to Dr. Benjamin Green, Prof. David Hutchison, and both the Systems Security and Networking Research Groups. The camaraderie, insightful perspectives, and intellectual engagement shared with my colleagues and friends were vital throughout this process. Our numerous debates have been a source of inspiration, proving instrumental at many pivotal moments. The NG-CDI Project has played a critical role in accomplishing this thesis. Their generous sponsorship was key to the progress and completion of this research, and I'm thankful for the opportunity to be part of an important project and to work with colleagues across the country.

I want to express my thanks to my examiners, Dr. Charalampos Rotsos and Prof. Shujun Li. Their expert scrutiny and constructive feedback greatly contributed to the quality of this work. I am profoundly grateful for the time and energy you expended in reviewing my work.

Lastly, my family's unwavering encouragement and faith in me provided a strong support system throughout this journey. Despite not being the easiest, with the distance between us and the pandemic, I could always rely on family, and for that, I am forever grateful.

This thesis represents the collective effort of all these individuals. Any success stemming from this research is a shared accomplishment, a symbol of the collaborative spirit of academia. I hope this work will contribute positively towards our shared goal of advancing cyber security for society.

Abstract

The advent of Industry 4.0 and smart manufacturing has led to an increased convergence of traditional manufacturing and production technologies with IP communications. Legacy Industrial Control System (ICS) devices are now exposed to a wide range of previously unconsidered threats, which must be considered to ensure the safe operation of industrial processes. Especially as cyberspace is presenting itself as a popular domain for nation-state operations, including against critical infrastructure. Honeypots are a well-known concept within traditional IT security, and they can enable a more proactive approach to security, unlike traditional systems. More work needs to be done to understand their usefulness within OT and critical infrastructure.

This thesis advances beyond current honeypot implementations and furthers the current state-of-the-art by delivering novel ways of deploying ICS honeypots and delivering concrete answers to key research questions within the area. This is done by answering the question previously raised from a multitude of perspectives. We discuss relevant legislation, such as the UK Cyber Assessment Framework, the US NIST Framework for Improving Critical Infrastructure Cybersecurity, and associated industry-based standards and guidelines supporting operator compliance. Standards and guidance are used to frame a discussion on our survey of existing ICS honeypot implementations in the literature and their role in supporting regulatory objectives. However, these deployments are not always correctly configured and might differ from a real ICS. Based on these insights, we propose a novel framework towards the classification and implementation of ICS honeypots. This is underpinned by a study into the passive identification of ICS honeypots using Internet scanner data to identify honeypot characteristics. We also present how honeypots can be leveraged to identify when bespoke ICS vulnerabilities are exploited within the organisational network—further strengthening the case for honeypot usage within critical infrastructure environments.

Additionally, we demonstrate a fundamentally different approach to the deployment of honeypots. By deploying it as a deterrent, to reduce the likelihood that an adversary interacts with a real system. This is important as skilled attackers are now adept at fingerprinting and avoiding honeypots. The results presented in this thesis demonstrate that honeypots can provide several benefits to the cyber security of and alignment to regulations within the critical infrastructure environment.

Publications

Parts of this thesis have, in parts verbatim, been published in the following peer reviewed conferences and journals:

- **Maesschalck, S.**, Staves, A., Derbyshire, R., Green, B., Hutchison, D., 2023. Walking Under the Ladder Logic: PLC-VBS: a PLC Control Logic Vulnerability Scanning Tool. Computers & Security, Volume 127. doi:doi.org/10.1016/j.cose.2023.103116
- **Maesschalck, S.**, Vasileios, G., Green, B., Race, N., 2022. Don't get stung, cover your ICS in honey: How do honeypots fit within industrial control system security. Computers & Security, Volume 114. doi:10.1016/j.cose.2021.102598
- Miller, T., Staves, A., **Maesschalck, S.**, Sturdee, M., Green, B., 2021. Looking back to look forward: Lessons learnt from cyber-attacks on industrial control systems. International Journal of Critical Infrastructure Protection, Volume 35. doi: 10.1016/j.ijcip.2021.100464
- **Maesschalck, S.**, Vasileios, G., Race, N., 2021. World wide ICS honeypots: A study into the deployment of conpot honeypots. Seventh Annual Industrial Control System Security (ICSS) Workshop, Annual Computer Security Applications Conference. https://www.acsac.org/2021/workshops/icss/2021-icss-maesschalck.pdf

The following publications have been published during my doctoral studies but are not included in this thesis:

- Derbyshire, R., **Maesschalck, S.**, Staves, A., Green, B., Hutchison, D., 2023. To me, to you: Towards Secure PLC Programming through a Community-Driven Open-Source Initiative. IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), pp. 358-362. doi: 10.1109/EuroSPW59978.2023.00045
- Staves, A., **Maesschalck, S.**, Derbyshire, R., Green, B., Hutchison, D., 2023. Learning to Walk: Towards Assessing the Maturity of OT Security Control Standards and Guidelines. IFIP Networking Conference, pp. 1-6. doi: 10.23919/IFIPNetworking57963.2023.10186424
- **Maesschalck, S.**, Bradbury, M., Vasileios, G., 2023. Into the Heat of the Debate: Simulating a Program Committee Within Computer Science Education. IEEE Global Engineering Education Conference (EDUCON), pp. 1-5. doi: 10.1109/EDUCON54358.2023.10125275

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Cyberspace has been gaining importance globally over the past decades, as the Internet becomes more and more a genuinely global network. The impressive rate of adoption and evolution within this area is phenomenal. Back in 1969, the first computers were connected with each other via ARPANET (Abbate, 2000; Lukasik, 2010) which we have evolved upon and has eventually led to the Internet as we know it. The evolution from ARPANET to the first stages of the modern Internet, where individual networks were being linked together to create a network of networks, involved many experts and intermediate stages and was managed in about 20 years (Leiner et al., 2009). In current times, we see and use Internet-connected devices on a near-constant basis. Even the coffee machine we use or our fridge might be connected to the Internet, and even our cars come with a data connection. In the modern age, having multiple Internet-connected devices has become a normality. This is also applicable to governments and militaries all over the world, where the Internet has become a major tool to conduct warfare (Harris, 2014), and there is an emergence of an Internet of Military Things (Banerjee et al., 2018). The evolution and adoption of cyberspace provide society with many benefits, but there is also an opposite side of the coin that affects both nations and their citizens alike. This calls for increased efforts to keep the Internet or cyberspace safe. To achieve this, it has to be done in both technical and non-technical ways.

The increasing adoption of the Internet of Everything (IoE) (Evans, 2012) also sees a shift within industry, as more and more industrial devices are becoming connected to the Internet leading to a wider threat landscape being able to access these systems. However, the devices not connected to the Internet are often still internally connected to the intranet which also brings security issues with them. This trend also encapsulates the industrial technologies categorised under the umbrella term of Industrial Control Systems (ICSs) (Bodenheim, 2014), which are used within critical infrastructure and are designed for high reliability. The convergence of traditional ICSs with machine-to-machine (M2M) and IP communications

has been characterised as part of the fourth industrial revolution, or Industry 4.0 (Lasi et al., 2014), which promises to improve operational functionality, manageability, and ease of access.

However, ICSs are not the only systems that are part of this fourth industrial revolution. The overarching term for many of these systems is cyber-physical systems (CPSs) which have computational and physical capabilities that interact with humans (Baheti and Gill, 2011). Industrial control systems as a subset of these are heavily used within industrial environments and in many critical infrastructure sectors and are used as the focus of this thesis. Many aspects of this thesis are also applicable to CPSs as they share similar challenges to ICSs. The focus of the thesis the focus lies within critical national infrastructure as an application environment that sees many targeted attacks (Miller et al., 2021) and has seen a significant shift in recent years as part of the convergence of IT and Operational Technology (OT).

1.1 The Fourth Industrial Revolution

The fourth industrial revolution has and will continue to alter the way we live and work. Industrial revolutions are not sudden events; they encompass multiple facets and can last a long period of time (More, 2002). During this time, multiple minor shifts within the industrial environment changed the way industry works. This is a good thing, as it allows other sectors to feed in and adapt to these changes. That is also true, especially within the fourth revolution, from a cyber security perspective. From the very start of the revolution, cyber security had to adapt and change with it. Within this thesis, we investigate one way how cyber security can adapt to this revolution and its domino effects within the critical infrastructure sector, by using honeypots.

From the name itself, it is clear we already went through three previous revolutions that impacted the way industry and society worked. The first revolution, estimated to have started in the 16th century, was characterised by the rise of industrialism and heavy machinery and had a significant impact on industries such as the cotton, agricultural and transport sector (Deane and Deane, 1979). This was the start of the major shift in how the industry would switch from manual labour to automation. The second revolution started in the late 19th century and was instigated by the adoption of electricity, oil, and steel within industry (Von Tunzelmann, 2003). This revolution also saw extraordinary growth in the scale of the economies and manufacturing due to several important inventions (e.g. food preservation and fertiliser) that allowed changes like the introduction of mass production (Mokyr and Strotz, 1998). The third revolution which began in the later part of the 20th century and saw the adoption of new technologies such as computers and the start of the so-called *Information*

Age (Greenwood, 1997). This also led to a change in the skill set needed in society, leading to an increased demand for skilled labour and IT expertise. Within critical infrastructure, we also saw an adoption of technology and the need for a device to bridge the physical and digital environment; a control system. Throughout this adoption, the OT environment still remained air-gapped from external connections and was locally managed (Ponomarev and Atkison, 2015). The fourth revolution is where we currently find ourselves and encompassed the rapid change to technology, as can be seen with the introduction of artificial intelligence and smart industry (Schwab, 2017). This revolution also found its way into our critical infrastructure. Now we see small sensors leveraging machine learning and artificial intelligence and a push towards remote management of ICS devices. However useful for managing these systems, this change is also very worrying from a security perspective. Within this thesis, we investigate this issue further, focusing on how the adaption of Internet connectivity has already led to significant impacts and how we could lower the risks involving this.

CPSs are a key aspect of the fourth industrial revolution and are at the heart of the emerging Internet of Everything (IoE) paradigm. The convergence of the physical, biological and digital worlds that characterises Industry 4.0 depends largely on the advanced interconnectivity enabled by CPS. These systems integrate computing power into physical processes, enabling real-time data collection and analysis and providing actionable insights for various applications. It is important to recognise that the breadth of the IoE framework, which envisions connections between people, processes, data, and things, can only be fully realised through the seamless integration and interoperability of CPSs. By merging the physical world with the virtual landscape, a CPS creates smarter, adaptive and autonomous systems that can improve efficiency, safety and productivity across multiple sectors of the economy. In this context, the role of CPS goes beyond mere technological innovation to play a vital role in shaping the socio-economic impact of the Fourth Industrial Revolution. Furthermore, the impact of CPSs in the context of IoE highlights key challenges and opportunities in privacy, cybersecurity, governance, and infrastructure development and deserves in-depth study.

As a subclass of CPSs, ICSs are particularly important in embodying the fourth industrial revolution. The seamless orchestration of physical operations and digital functions enabled by ICS provides the foundation for the smart industrial infrastructure of the future. ICS integrates sensor networks, control equipment and information systems to monitor and manage industrial processes in real-time. By enabling this high level of automation and control, an ICS brings unprecedented efficiency and optimisation to manufacturing, power distribution and supply chain processes, among others. The central role of ICSs in the larger CPS ecosystem bridges the gap between the physical and digital worlds and essentially lays the foundation for a broad IoE. Integrating ICSs into the broader vision of Industry 4.0 and

IoE not only increases the potential for technological innovation but also raises pressing questions about data security, system stability and regulatory compliance. Effective use of ICSs within the broader context of CPSs and IoE, therefore, requires a holistic approach to risk management, governance and cybersecurity, driving the transformative potential of the Fourth Industrial Revolution.

1.2 Industrial Control Systems and the Internet

ICSs were not designed with Internet connectivity in mind (Ahmed et al., 2017), and often lack basic security features (Mirian et al., 2016), making their newfound Internet-connectivity a more significant issue. Moreover, when security features are available for ICS, they are usually *bolt-on*. These can include vulnerabilities of their own, such as Secure Authentication version 5 (SAv5) for DNP3 (Distributed Network Protocol 3) (Crain and Bratus, 2015), which is vulnerable to single-frame attacks and does not encrypt data between server and outstation (Cremers et al., 2019), and allows for the use of SHA-1 (Rosborough et al., 2019) amongst others. Generally, bolt-on security is considered a weaker option to secure a system (Shiva et al., 2010) and harms both usability and security (Yee, 2004). The risk of ICS attacks is amplified by the fact that these systems are often implemented as part of the critical infrastructure within a country, including water and electricity distribution (Green et al., 2017). Therefore, ICS security is paramount to the safety and economic prosperity of a nation while also presenting an attractive target for cyber-warfare operations. Not to mention the shift of cyberattacks targeting critical infrastructure from initially internal personnel to nation-states in the present time (Miller et al., 2021). Furthermore, this needs to be taken into account within risk assessment, such as described in the adversary cost framework (Derbyshire et al., 2021b), as a nation-state adversary generally has more resources at its disposal.

Even when manufacturers provide patches for known vulnerabilities, patch deployment times can be significantly higher compared to traditional IT systems, leading to more prolonged exposure times (Dey et al., 2015; Marnerides et al., 2019). Delayed patching can be explained through the requirement for continued operation and minimal downtime. Within an ICS environment system, reliability takes precedence over security (Maglaras et al., 2018), meaning ICS operators may prefer to leave systems unpatched. However, where vulnerabilities exist, their exploitation has the potential to harm operational productivity, reliability, and even human life. This is particularly worrying given the potential impact an attack on these systems can have on a nation and surrounding nations when looking at, for example, the nuclear sector. Honeypots are an interesting approach to security and could provide a

different way to solve these issues we identified. However, these systems have not been widely researched from critical infrastructure and operational technology aspects.

1.3 Motivation

To effectively protect ICSs, developing new methods for attack detection and mitigation is necessary. Adopting traditional firewalls and anti-virus solutions is no longer sufficient, as they are reactive and require updates to detect/prevent new forms of malicious traffic (Bilge and Dumitras, 2012). Consequently, zero-day exploits, an exploit that is not yet publicly disclosed, can potentially penetrate networks and infect systems while remaining undetected. The introduction of "bring your own device" within organisations and the prevalence of social engineering (Derbyshire et al., 2018), has rendered conventional perimeter defences inadequate (Wang et al., 2014). Due to the merging of OT and IT, these threats now also apply to ICSs. In addition to this, these environments are targeted by advanced persistent threat (APT) groups (Bhamare et al., 2020; Chen et al., 2014; Kim et al., 2014), usually sponsored by nation-states, which require more bespoke security measures as they do not go for the lowest hanging fruits but prioritise remaining undetected (Alshamrani et al., 2019). Additionally, these threat actors are more likely to exploit ICS-specific vulnerabilities, which might be highly sophisticated.

One of the ways in which we can aim to mitigate attacks on the network and discover novel attacks is through deceptive cyber defence techniques such as honeypots. Honeypots are systems with no inherent purpose other than capturing attacks, either on the Internet or within a network, and generally do not receive any legitimate traffic. Both academia and industry have been using and researching honeypots in a range of different use cases. These different use cases tend to have many different setups. Whereas academia can deploy honeypots in many different configurations, industry tends to focus on honeypots that more closely align with their operations. The distinction between research and production honeypots encompasses these differences in purpose and setup. There are many different types of honeypots, ranging from emulating specific services such as SSH to a fully-fledged system running several services at the same time. Not only the honeypot itself is important to consider, but also where it is deployed. For example, it would be a red flag for attackers to see an ICS honeypot deployed by an organisation that generally does not use these systems. Contrary to traditional security systems that are often reactive, honeypots enable a more proactive approach to security. Adversaries are encouraged to attack these systems to reveal valuable threat intelligence. Capturing attacks performed by real-world adversaries can be used to discover new vulnerabilities and associated exploits alongside a broader view

of offensive tactics and techniques. The level of encouragement differs depending on the purpose and environment in which the honeypots are deployed, as honeypots could fall into the legal aspect of entrapment.

Generally, in a real-world environment, you would want an adversary that has entered the organisational network to be more likely to investigate a honeypot than an operational system. Therefore, honeypots need to be an attractive target for intruders. In 2020 four zero-day exploits were discovered by ICS honeypots setup for research purposes (Ranger, 2020), which is one of the success stories that prove the viability of these systems in detecting novel attacks. The introduction of honeypots within these environments has to be carefully thought through as these sectors are tightly regulated, and the adversaries who target the systems are typically highly advanced. These APT groups and the sophisticated exploits they use sometimes require the deployment of bespoke security measures, and honeypots can be used to present as an alluring target when deployed to emulate a particular vulnerability. Therefore, adequate consideration and investigation have to be done to make sure any security measure developed for these environments takes these variables into account. For example, how do new security measures, such as honeypots, fit within these regulations, and can they be used to support adherence to them?

Further to this, in relation to the state-sponsored APT groups targeting the environments, it is necessary to establish why this is happening. This includes the exploration of why cyberspace has become such a popular domain for nations to operate in, as attacks on ICS infrastructure shifted from insider attacks to nation-states (Miller et al., 2021), and what the impact of attacks on critical infrastructure within this space can be. Answering this final question should help with investigating how these infrastructures can be protected effectively, both technically and by having necessary laws and regulations.

The field of honeypot research has identified and aimed to address a number of challenges already, including scalability, platform development and monitoring. Papers looking into the development of realistic honeypots are plentiful, aiming to make honeypots indistinguishable from real systems and increase their value even more. These challenges are also applicable to the field of OT honeypots. However, in this field, there are a number of limitations that also have to be taken into account. For example, the adoption of machine learning and artificial intelligence is not as welcomed, due to the nature of the environments, as these are often critical to operations and accidents can easily lead to loss of life. Especially in environments related to critical infrastructure the difference between acting on a false positive or false negative can be consequential.

Based on this, the following challenges have been identified in relation to the usage of honeypots within OT environments:

1. **Regulation**
   The critical infrastructure is heavily regulated because of its critical nature to society. Hence, before implementing any system, it needs to be investigated how it fits within these. In relation to honeypots, this not only ensures the security of the environment but also strengthens the case that honeypots should be incorporated and investing in these systems is beneficial.

2. **Deployment challenges**
   When investigating the deployment of honeypots within critical infrastructure it has to be certain that they will not interfere with existing systems, not open up the environment to risks the organisation is unwilling to take, and that they are deployed to effectively gather intelligence. This means that thought has to be put into what the purpose of the honeypot that will be deployed and based on that the type of honeypot has to be decided. Further to this, it is important to deploy the honeypots in such a manner that their nature is not obvious to adversaries.

3. **Honeypot detection**
   Honeypots are not uncommon to adversaries, especially knowledgeable adversaries that are targeting critical infrastructures. This means that anti-honeypots techniques are used by adversaries which can potentially limit what a honeypot can detect from an attacker using these. Therefore, it has to be investigated how an attacker can identify honeypots. Although research has been done into honeypot fingerprinting, these techniques have been reliant on active scanning, and there remains a possibility an adversary can passively identify honeypots through Internet scanners (e.g. Shodan), which have become popular. In addition to this, this raises the question if anti-honeypot techniques can be turned into a benefit for the security of systems.

4. **Honeypot avoidance and deception**
   Adversaries aim to avoid and circumvent systems that are in place to identify them, especially when looking at state-sponsored and nation-state attackers with the aim to remain undetected as long as possible. This also includes honeypots, which, as stated in the previous challenge, can be detected. Therefore, further research has to be done to mitigate this issue and enable honeypots to remain valuable in their effort to identify attackers and reduce risk within the environment. Work in this area can include techniques such as deception and the development of more advanced honeypots.

1.4 Thesis Aims and Contributions

This thesis aims to investigate the role of honeypots, and novel honeypot-based security measures within an ICS environment. This is achieved through an examination of the following key aims and contributions summarised below:

1. **Analyse the role of honeypots within ICS security and critical infrastructure frameworks, standards and guidance**

   Honeypots have been very common within traditional IT environments due to their unique capabilities in luring adversaries to them and their value within the threat intelligence process. Data obtained through honeypots can be invaluable in improving the security of systems and networks as they are capable of detecting zero-day vulnerabilities by providing real-time information on network activity and enabling a proactive approach to security. However, honeypots have not been as popular within critical infrastructure and ICS environments even though they can provide many benefits. To strengthen the adoption of honeypots within these environments it is important to show how they can assist in meeting strict regulatory requirements, such as those set out by NIST and the ONR, which are often supported by standards and guidance for specific implementation. In Chapter 3 of this thesis, we discuss how honeypots fit within these regulations and ICS security. This has been done by mapping the requirements of the UK Cyber Assessment Framework, a framework to assist with NIS regulation, to the capabilities of honeypots. Other regulations, standards and guidance where honeypots can provide benefits are also discussed.

2. **Develop a novel honeypot deployment framework to improve the detection of bespoke ICS vulnerabilities**

   The deployment of honeypots is one of their most important considerations, as wrongly deployed honeypots can fail to capture data or introduce more security risks to an environment. An incorrectly deployed honeypot can provide adversaries with a system they can leverage to exploit the network. They can, for example, be used as a vantage point to obtain further visibility of the network or for privilege escalation. This means different types of honeypots are more suitable to different environments. In addition to the risks, a wrongly configured honeypot can also be obvious to the attacker. Therefore reducing the chances they would interact with the system and in turn, reducing the quality of data gathered. Correct deployment and configuration to mitigate the risks and improve the data gathered by honeypots is paramount. Within this thesis, in Chapter 5, the link between the configuration of the honeypot and the data obtained

is investigated, which is used to provide key takeaways on how to improve honeypot configurations. In Chapter 6, the deployment of honeypots is discussed based on a novel framework that emphasises where each type of honeypot should be deployed and how to achieve the maximal benefits from their deployment. Chapter 7 discusses the deployment of bespoke honeypots for a specific vulnerability, and provides an example of a novel memory vulnerability discovered.

3. **Measure the fingerprintability of ICS honeypots based on passive data analysis of Internet-wide scans**

   An adversary has to be mindful of security measures deployed within the network and on the systems. Although honeypots can sometimes be easily identified when interacting with them, the interaction itself makes the honeypot successful. Therefore, knowledgeable adversaries aim to identify honeypot characteristics before interacting with the system. Especially within an ICS environment, this is important, as highly-skilled state-sponsored adversaries are active within it. Internet-wide scanners such as Shodan are useful for this, as they scan the system and make the data available that can be used for passive reconnaissance. These scanners can sometimes also be used to identify systems as honeypots directly by checking for honeypot tags or obtaining the Shodan Honeyscore. However, these are not always accurate. In this thesis, Chapter 8 investigates the feasibility of passively scanning for ICS honeypots by using data from Shodan or other Internet-wide scanners. Furthermore, it investigates the accuracy of the Shodan Honeyscore, which was specifically developed to identify ICS honeypots.

4. **Design and implement a programmable ICS obfuscation network overlay to deflect adversaries** When deploying honeypots one of the most challenging tasks is to configure the system in such a way that it does not have any honeypot signatures as these deter adversaries from interacting with it. This requires in-depth knowledge of the system the honeypot is aiming to mirror and can require a significant amount of resources to develop an interactive environment. For example, the honeypot might need fake data and fake traffic. However, the usage of honeypots as obfuscation systems has not yet been properly investigated in practice. Turning the main downside of honeypots into a strength takes a fundamentally different approach and can provide a compelling reason to deploy more honeypots as it lowers the resources needed to make honeypots present themselves as realistic systems and provide system-centric security. Chapter 9 investigates this complete change in the usage of honeypots and the concept of obfuscation within an ICS environment by developing and implementing a Ryu-based obfuscator honeypot on top of a PLC leveraging SDN.

1.5 Thesis Structure

The remainder of this thesis is structured as follows:

Chapter 2 introduces essential concepts that must be considered in the field of critical infrastructure. Given the Internet connectivity of critical infrastructure environments, the notion of cyberspace is essential to consider when working on the security of these systems. This includes understanding what industrial control systems are and how Internet connectivity has impacted them. Given the appeal of these systems in terms of cyber operations, such as Stuxnet, these concepts are also investigated within this Chapter. Additionally, the calls for further regulation of this space, partially due to the criticality of these systems, are explored, and several ICS attacks are discussed.

Chapter 3 goes further in-depth, building upon the previous Chapter, on the concepts of honeypots and ICS security. It covers the background of honeypots, ICS security and ICS Regulation. Concepts such as defence-in-depth are essential to consider now that networks are getting Internet-connected, and honeypots fit well within this concept. In terms of regulations, the main ones covered within this Chapter are the U.S. NIST Framework for Improving Critical Infrastructure Cybersecurity and the U.K. Cyber Assessment Framework. However, other regulations are also touched upon. This feeds into an investigation into how honeypots fit within ICS security and regulation.

Chapter 4 covers the current state-of-the-art regarding ICS honeypots by conducting a survey of the literature through academic databases where we investigate low- and medium-interaction honeypots and high-interaction honeypots deployed. We evaluate and analyse each of the implementations and provide a summary. Based on this analysis we propose a novel honeypot classification to emphasise the importance of the environment the honeypot is deployed within.

Chapter 5 investigates how honeypots can be deployed such that organisations obtain the most value from their deployment. This leverages the work covered in Chapters 3 and 4. It covers both low- and high-interaction honeypots and presents a case study where different Conpot honeypots are evaluated. Further, improvements that should be considered are presented to strengthen honeypot deployments. An overview of easily identified Conpot Deployments on the Internet is also given to reinforce the need for careful configuration and deployment of honeypots. This is further bolstered by an investigation into how Shodan discovers and identifies ICSs.

Chapter 6 introduces HoneyPlant, a framework for honeypot deployment within critical infrastructure environments. This evolved from the work presented in Chapters 2, 3, 4 and 5. The framework is based on making the most of honeypots and their alignment with the regulations within the sector. It presents different areas where honeypots can be deployed

within and outside the organisational network. This is then mapped to the ICS attacks discussed in Chapter 2 as we investigate how HoneyPlant could have made a difference within these attacks and which benefits the framework provides.

Chapter 7 studies the deployment of honeypots to identify the exploitation of bespoke vulnerabilities. The work in this Chapter leverages the work done in Chapters 5 and 6. Several ICS vulnerabilities are discussed, and a novel PLC memory vulnerability is presented. This novel vulnerability is then evaluated by creating a tool that scans PLC control logic for this bespoke vulnerability. Several attack scenarios are also presented. Finally, a high-interaction honeypot is explicitly deployed to allow for and identify the exploitation of this new vulnerability.

Chapter 8 investigates the concept of ICS honeypot fingerprinting in a passive manner. Several honeypot characteristics can be used to identify ICS honeypots, which can be obtained through data provided by Internet scanners. Shodan is leveraged to obtain a large dataset of ICSs available on the Internet. This builds upon the work covered in Chapters 4 and 5. Afterwards, a passive classification algorithm is developed to identify the likelihood of systems within this dataset being honeypots. This algorithm is evaluated with active scanning and compared with Shodan Honeyscore. It is identified that the Shodan Honeyscore, although widely used to evaluate novel honeypots, is inadequate to scan for ICS honeypots.

Chapter 9 examines a novel way to leverage and deploy honeypots. Instead of deploying honeypots to present themselves as close to a real system as possible, honeypots are being used to obfuscate real systems and lower the adversary's appetite to interact with the system. This negates the challenges presented and the resources needed to make a honeypot not appear to be a honeypot, which is presented in Chapters 4 and 5. An architecture for the Obfuscator system is presented and evaluated with a Hidden Markov model and a practical evaluation within a realistic environment. Within the evaluation, SDN is leveraged to enable a dynamic network environment.

Chapter 10 concludes the thesis by presenting an overview of the thesis contributions and their outcomes and outlining future work within the area of ICS honeypots.

Chapter 2

Cyber Operations and ICS

As we have established earlier, society has become increasingly reliant on the Internet to the extent it would not function without it; the presence of Internet-connected devices has become ubiquitous with the emergence of the Internet-of-Things (IoT). This trend is seen in all aspects of our society, also in the military. With militaries around the world adopting more cyber capabilities, we have to evaluate both sides of the coin. This work focuses on the security of OT environment within critical national infrastructure where ICSs are increasingly being connected to the Internet, opening them up to a wide range of attacks and vulnerabilities. This Chapter introduces cyberspace and how we can define it. Then, it covers what industrial control systems are and touches upon cyber warfare and cyber terrorism. Covering these concepts and operations within cyberspace is important to help us investigate attacks on ICS infrastructure later in this Chapter.

2.1 Introduction

The Internet as we know ts has evolved from ARPANET, which was an experiment initiated by the U.S. Department of Defense (Leiner et al., 2009), so its importance in military operations has always been a critical factor in its development. However, the Internet has relatively quickly evolved beyond its military purposes. In the current day and age modern home devices, such as “smart” TVs, rely on the Internet to connect with services and channels, while applications such as WhatsApp and iMessage have essentially replaced traditional telephone communications. Cash payments have been replaced by contactless over the Internet transactions, business meetings are conducted over video calls, and our critical infrastructure has become increasingly digital. Importantly, the Internet has grown beyond commercial and entertainment applications; today’s governments and militaries have, just like most of society, become increasingly dependent on the Internet (Garamone, 2019).

Within the military, now, specialist forces are being trained to fight in cyberspace (Apps, 2012), and modern weapons systems are controlled online (Hall, 2016). By becoming more reliant on digital infrastructure, the military is opening itself up to a new domain from both offensive and defensive perspectives.

This combined reliance of society and the military on the Internet has also turned it into an interesting attack vector. Especially as part of the adoption of the Internet has led to advances such as CPSs including ICSs and the industrial IoT which sectors such as manufacturing, water and nuclear sectors have adopted as well. From smart sensors to smart robots and the connection of previously disconnected systems to the Internet, industry has seen a significant revolution as well.

Warfare has also increasingly evolved and now encompasses this new domain of cyberspace. In 2018 NATO stated that "[NATO] must be able to operate as effectively in cyberspace as we do in the air, on land, and at sea to strengthen and support the Alliance's overall deterrence and defence posture" (NATO, 2018). This strengthens the view that cyberspace is a key domain in the current forms of aggression between States, and it is vital to protect against operations in this domain. Even during peacetime, cyber-attacks are being used to interfere with nation-states (Fitton, 2016), allowing adversary governments to operate under the "cloak" of anonymity. States that rely on such covert cyber operations may even deliberately use attacks of low sophistication that are typically below the level of military expertise in an attempt to deny their involvement (Courtney, 2017). Therefore, cyber operations have blurred the lines between conflict and peacetime and created a complicated landscape of international relationships (Fitton, 2016).

Other nations are focusing on this new domain as well; in the U.S., Cybercom has been formed as a new division within the Pentagon to focus on cyber operations (The Economist, 2010), and President Barack Obama declared digital infrastructure to be a national asset (The White House, 2009). Within the United Kingdom, the Government Communications Headquarters (GCHQ) has expanded to include a new National Cyber Security Centre (GCHQ, 2016). These actions are proof that the digital infrastructure is of imperative national importance and consequently, a potential target for state and non-state actors to launch attacks intending to damage or influence the critical operations of a country.

Because of this, it is important to investigate what the state of cyber warfare is and how this relates to ICS, as ICS is a prime target to disrupt countries and put pressure on governments. Especially as these systems are critical to the operation of many vital operations that underpin our society. Therefore, these concepts need to be explored and kept in mind when investigating security measures for critical infrastructure. As these environments are highly specialised and critical to the operations of a country, it can be assumed that there

will be highly skilled adversaries actively looking for vulnerabilities and ways to impact operations.

2.2 Cyberspace

Before looking deeper into cyber operations, we first explore cyberspace itself. Due to the predominantly virtual nature of cyberspace, it can be challenging to comprehend its concept fully. Kuehl (2009) divides cyberspace into four aspects: the natural domain, the operational space, interconnected networks, and the information-based aspect. The natural domain refers to the electromagnetic activity used to transmit digital information in bits. The operational space is defined as the usage of cyberspace to act and create effects within it or within other domains. For instance, cyber operations can affect other systems within cyberspace, like databases. On the other hand, disabling communications will affect the other four domains of warfare that lie outside cyberspace. As cyberspace is used to create, modify, store, change and utilise information, it is information-based. Everything in cyberspace boils down to zeros and ones; the binary information used to represent data transmitted across the domain. The interconnected aspect of cyberspace is seen as the connections that allow electromagnetic activity to flow within it. These connections are essentially the physical media (cables, radio waves and fibre optics) used to transport data and connect several home and enterprise networks with each other. These networks are distributed all over the world and together form the Internet.

A vital aspect of this definition is the global nature of cyberspace; it is accessible everywhere and by anyone. As we stated before, cyberspace has a predominately virtual nature, but its physical representations must not be forgotten. The infrastructure itself, i.e. the machines on which it runs, is vital for its operations. Any disruptions within those systems could have an impact on cyber operations.

In another representation of cyberspace, we can divide cyberspace into four layers (Clark, 2010): information, logical, physical and the people. All these layers rely upon each other to exist and are vital to the operation of cyberspace. We can categorise the layers into two groups, ‘transformative’ and ‘logistical’. Under transformative, we place the people, as they have the power, based on their actions, to transform cyberspace itself. Without this layer, cyberspace would be static and useless. It is the people who provide the data within cyberspace; without it, cyberspace would be an empty book. The other layers are needed to operate the space itself. The information layer stores and transmits data throughout cyberspace and is arguably one of its most virtual and valuable aspects. It is the data that tends to be fought over and can provide crucial information when stolen. The logical layer is

central to the nature of the Internet, and it represents the services that support the platform. These services can be low-level, such as the transport protocols and data formats, but we can compose them to create new and more complex services by combining these services. These complex services are the ones most people interact with whether it is browsing the web (databases, transport protocols, presentation protocols etc.) or writing a document in a text editor. All these layers are supported by the physical layer, which is seen as the foundation of cyberspace. Without the physical devices and their interconnection, there would be no communication possible. A computer by itself has far fewer capabilities than when it is connected to a global network of information. Services such as email, streaming and video calling would be impossible without these connections.

Due to its accessible nature, cyberspace can be seen as a domain in which states and non-state actors can compete on reasonably equal terms (Venables et al., 2015). This is very different from the traditional domains of warfare (land, air, water and space), where having a large budget enables a State to gain an advantage on warfare capabilities by investing in the best equipment. Non-state actors now have the opportunity to participate in conflicts against states anonymously and inexpensively. Disruptive attacks that are hard to defend – such as Distributed Denial of Service – can be executed with minimal expertise using automated services (Karami and McCoy, 2013). States are increasingly aided by non-state actors in their conflicts (Sigholm, 2013). For example, China has been leveraging patriotic hackers for this purpose, and even if they do not support them directly, they do turn a blind eye when these groups are targeting foreign organisations (Hvistendahl, Mara, 2010). Russian patriotic hackers have also been at the forefront of conflicts between Russia and other nations as can be seen with the attacks on Estonia and Georgia (Karatzogianni, 2013). Additionally, the cyber aspect of the Russian-Ukrainian was very much anticipated to have a significant role in the conflict and we saw a lot of participation from non-state groups such as hacktivists and APTs (Serpanos and Komninos, 2022). Due to these distinctive features, cyber operations are considered a part of the new fifth military domain of cyberspace (The Netherlands Ministry of Defence, 2018), along with land, air, sea and space. This is vital as by recognising the space as a new military domain; countries recognise the military opportunities and impact that can be made within this space. It further acknowledges the increased reliance on cyberspace and the adoption of technology within the military environment. Whereas cyber started out as a technology used within the other domains, by recognising it as a new domain, the world admits it has grown from this and become big enough to warrant being a new military domain. This can also be seen in the increasing attention to cyberattacks in the public discourse, which has made them the second most serious threat as perceived by the public in the United States (Pew Research Center, 2013).

2.3 Industrial Control Systems

ICSs, together with Supervisory control and data acquisition (SCADA) systems (Gaushell and Darlington, 1987) are a subset of cyber-physical systems (CPS) (Baheti and Gill, 2011) and underpin critical parts of national infrastructure as part of the OT environment (Hahn, 2016). Whereas SCADA systems focus on the acquisition of data and the processing thereof to be used by the operator, ICSs control and automate industrial process operations within a variety of industries, including nuclear, water, oil and gas, and electricity (Green et al., 2017; McLaughlin et al., 2016). These control systems. in particular PLCs (Programmable Logic Controllers), are the main focus of this thesis. Due to the organisations that deploy ICSs, it is clear that the impact of an attack on these systems can be considerable. Therefore, appropriate defence mechanisms should be in place to prevent potential damage. The current trend of Internet-connected ICSs opens these systems up to a variety of threats. ICSs were not originally designed to communicate over the Internet (Ahmed et al., 2017). The operating systems (OS) and other software used within these systems can have vulnerabilities that are not regularly patched, and specific protocols used present many difficulties due to their design, which adversaries can potentially exploit. The vital function an ICS has within critical infrastructure, combined with the insecure design of ICS protocols, can lead to potentially catastrophic events.

Because of this, novel defence approaches are needed to mitigate emerging threats. ICS devices are built using commercial OSs that are highly specialised, and therefore, ICS security differs considerably from standard approaches to security (Knapp and Langill, 2014). Existing ICS security solutions aim to minimise disruptions in ICS availability by focusing on protecting the IT infrastructure around the ICS devices (Larkin et al., 2012). Due to the importance of these devices, any interference or additional latency can have significant effects (Jie and Li, 2011). ICSs can be operational without interruptions for up to two decades (Jie and Li, 2011), unlike IT systems which are regularly updated (Hunter, 2006) or replaced (Frye, 2013). Such a gap between the discovery of a vulnerability and the implementation of the patch allows attackers time to discover and exploit them for years after those vulnerabilities have been published (Marnerides et al., 2019).

Typically ICSs are deployed within a complex environment that consists of several layers of logically-related operational abstractions. One of the most popular representations of these layers is the Purdue model (Figure 2.1) which consists of an enterprise zone, demilitarised zone (DMZ), manufacturing zone and safety zone (Didier et al., 2011; Green et al., 2016). The function of the *safety zone* is to house systems that provide predictable fail-safe shutdowns to protect processes, personnel, and the environment. They also monitor the processes running for any anomalies (Obregon, 2015). More recently, CIP Safety (ODVA, n.d.) allows

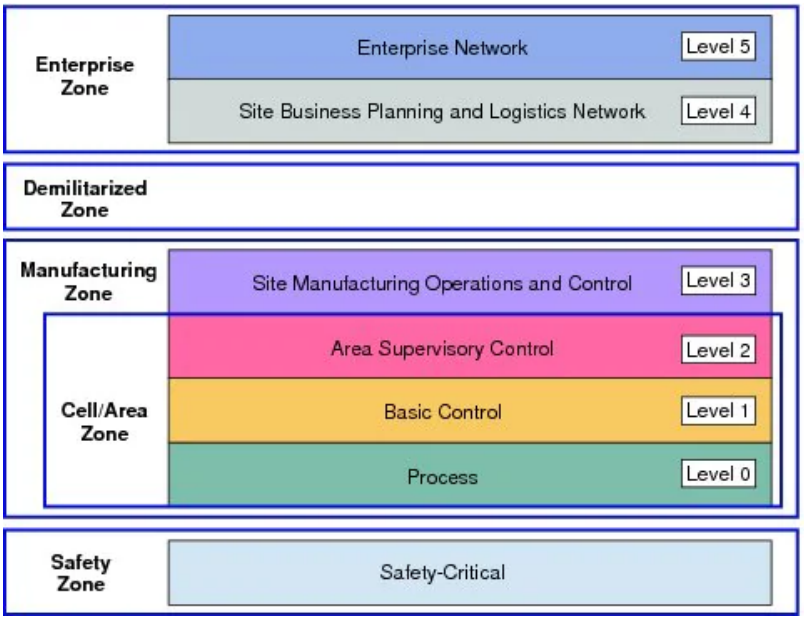


Fig. 2.1 The Extended Purdue Model which Describes the Layered Architecture of ICS (Didier et al., 2011)

devices such as safety sensors to operate alongside level 0 systems and safety controllers with controllers on level 1. Within the *manufacturing zone*, there are four levels that contain a set of devices including PLCs, HMIs (Human Machine Interfaces), and RTUs (Remote Transmission Units), which are used to monitor, control and automate processes. Devices within this zone include sensors, human-machine interfaces, remote terminal units and control servers. The *DMZ* is implemented as a boundary between the manufacturing and enterprise zone, and it generally contains IT infrastructure that has the capability to communicate with the OT devices. It presents an interface for further processing on data and facilitates services like remote desktop and remote alarm management. Within the *enterprise zone*, conventional IT devices such as clients and servers are deployed and use data collected via the DMZ to supervise and dictate future strategic planning for the entire infrastructure.

2.4 Cyber Warfare

Cyber war needs definition before exploration, with terms like cyber war, cyber warfare, cyber operations, and cyber attacks often used interchangeably. Beginning with cyber war, it's necessary to first define war. War is typically viewed as a state where organised groups are engaged in hostility, and no party has effective control over the conflict area (Wolfendale, 2017). Although generally, war starts with a declaration and ends with a peace treaty, and

there can be exceptions (Ronan, 1937). For a deeper discussion on war's definition, refer to Dinstein (2017). Next, cyber war can be defined as warfare happening solely in cyberspace between nations, which could also involve non-state actors (Cornish et al., 2010), underlining cyberspace's accessibility. Cyber warfare, also formerly known as information warfare, refers to planned attacks on enemy data and computing systems by nations or their agents but does not necessarily imply enemy losses (Janczewski and Colarik, 2007). Cyber operations and cyber attacks are specific actions in cyberspace associated with cyber warfare (Eom et al., 2012; Lin, 2010), not necessarily resulting in data loss. Examples include data gathering, spreading malware to modify system operation, as seen in the Stuxnet attack, creating botnets, or extracting system information. For an extensive exploration of cyber warfare, refer to Wolfendale (2017).

The debate about cyberspace's role in warfare is ongoing. Thomas Rid asserts cyber should be seen as a tool for sabotage, espionage, and subversion, rather than a new form of war (Rid, 2012). However, this viewpoint is contested by McGraw (2013) and Stone (2013). Arquilla, too, revised his 1990s claim that cyber wars were not imminent, acknowledging by 2013 that sustained cyberattacks were already happening (Arquilla, 2013). The Russian-Ukrainian war showcased cyber's role in conventional warfare, highlighting its usage by non-state actors. The interpretation of cyber attacks and operations within conflict remains disputed, with some experts viewing it as a tool within grey zone warfare (Carment and Belo, 2018; Fitton, 2016; Willett, 2019), and others considering it more of a diplomatic instrument (Maness and Valeriano, 2016). This ongoing debate could be due to cyber's dual nature as a tool and a form of warfare, its continuous evolution, or the need for a better understanding of cyberspace. While complete wars within cyberspace seem unlikely at present, cyber has become an integral part of warfare. This is evidenced by cyberspace's recognition as the fifth military domain and actions such as the UK's establishment of the National Cyber Force (Devanny et al., 2021), consolidating offensive cyber capabilities. It implies the potential for full cyber wars, as each military domain historically started similarly, as the US Navy was founded in 1775, although naval battles predated it (Rodger, 2004).

Attacks on critical national infrastructure (CNI) can have a disastrous effect on a large area and can have a serious impact on a country and its military capabilities. Whilst grey zone warfare denotes operations that lie between the concept of peace and war, launching a cyber-attack on critical infrastructure or integral systems could have an impact on civilians and militaries as high as traditional warfare operations. One such attack on CNI that could have had serious consequences for civilians was the attack on the Florida water treatment plant in 2021 (Vera et al., 2021). If this attack was not detected and did succeed the water supply to a city would have been poisoned. The potential for cyber operations to creep above

the threshold for military retaliation has already been demonstrated in recent times as well. Israel's airstrike on a building linked to a Hamas hacker group in 2019 is a key example. In what has been described by some as a 'potential first for cyber war' (Newman, 2019) this event marked the first time that a kinetic military response has been carried out in retaliation for an alleged cyber-attack or threat thereof. Although this example is not one of a cyber attack on CNI, it denotes the seriousness of the risks of escalation and war that surround the use of cyber in conflicts. Additionally, militaries are also exploring the integration of AI within tactical and operational levels (Davis, 2022), which would open up a new attack vector within cyberspace. This could lead to systems performing unpredictably and could allow adversaries to impact automated responses by interfering with the AI system.

Cyber operations can greatly benefit traditional warfare, often yielding disruptive rather than destructive outcomes Brangetto and Veenendaal (2016). Unlike traditional weapons, cyber-attacks primarily aim to disrupt infrastructure rather than destroy it. This ties into the military concept of manoeuvre, essentially using cyber operations to gain positional advantages in other warfare domains (Joint Chiefs of Staff, 2017). Anonymity is a significant advantage of cyberspace as attributing attacks is challenging (Rid and Buchanan, 2015). Furthermore, it is easier to conceal cyber capabilities, with the attacker's identity often remaining unknown for years after an attack. Cyberspace is also an ideal domain to influence goals and policies. Besides system and infrastructure attacks, cyber strategies can also include 'hacking minds' or using social engineering to manipulate public opinion and government actions (Rugge, 2018). Thus, today's cyber attack vectors target both strategic national infrastructure and people Brangetto and Veenendaal (2016).

The first attack vector is the hacking of strategic national infrastructure. One of the most critical pieces of the national infrastructure is the power grid. When it gets disrupted, the whole nation can be impacted, including civil, government and military systems. Within the military, this might disrupt C4I (Command, Control, Communications, Computer, and Intelligence) systems, which are relied upon to command all aspects of the war (Rugge, 2018). Whilst the State is trying to recover from the loss of systems, an attack would require the defending nation to divide its available personnel and resources between recovery and defence. For example, it is possible that airspace monitoring and control systems become unavailable; thus, enemy aircraft cannot be detected when they enter the airspace. On a non-military level, the disruption of critical national infrastructure has an enormous impact on civilian life within the targeted country. Because of this, more countries are increasingly calling for agreed rules that govern the use of cyber weapons in relation to critical national infrastructure (Courtney, 2017).

Hacking minds sounds interesting, but in essence, it is influencing opinions. Humans are generally viewed as the weakest point within cybersecurity, as over 95% of security incidents can be attributed to the human element (Widdowson and Goodliff, 2015). A system can be hardened against exploits; vulnerabilities in systems can be patched, but humans themselves have the power to make decisions on their own. Within the military, social engineering proves to be a significant risk. When an employee is phished, and the computer is infected with a RAT (Remote Access Trojan), a malicious actor can access the compromised device. Depending on the level of access the employee has, additional systems might be accessible as well. This would allow an adversary from another State to monitor systems, gather data, and even disrupt their operation.

The potential of other States monitoring and gathering data from the vulnerable systems introduces a new form of espionage, cyber espionage. Through cyber espionage, critical information can be accessed and monitored. For example, troop movements, locations of infrastructure, and communications between officers can all be obtained. This information can be pivotal for another State to gain an advantage within both peacetime and war. Before cyberspace, espionage was conducted in person through spies, which increased the risk for a foreign adversary and the spy conducting the espionage. Instead, within cyberspace, boots on the ground are not necessary. In 2015, a group of Russian nationalists claimed responsibility for the infection of 20,000 computers in the German parliament which started sending sensitive data back to the hackers (Courtney, 2017). The attack was allegedly supported by the Russian government, but no concrete evidence could be found. Furthermore, private companies related to the military and government can also be targeted as there have been cases where terabytes of sensitive military data have been stolen from corporate networks (Eom et al., 2012).

2.5 Cyber Terrorism

Cyberspace is accessible to all, including those with malicious intentions like hackers and fraudsters. Cyber terrorism activities, such as those by the Islamic State's Cyber Caliphate (Liang, 2017), benefit from the anonymity and ease of participation in cyberspace. This connects to actions by individual hackers and groups like Anonymous (Olson, 2013), responsible for numerous cyber attacks against governments and corporations. Hacktivism, particularly among tech-savvy younger generations, is increasingly prevalent, likely to grow further. Cyber attacks offer an alternative to traditional protests, with the potential to disrupt entire organisations. While anonymity is prevalent in hacktivism and cyber terrorism, laws like the U.K. Computer Misuse Act, the U.S. Computer Fraud and Abuse Act, and the

EU Directive on attacks against information systems (2013/40/EU) are in place to sanction non-state actors conducting illegal cyber activities. However, enforcement is challenging. If an actor is identified, issues such as the perpetrator or crucial investigation data being in another country can complicate actions against them, with jurisdiction issues often hindering cyber investigations (Ghappour, 2017).

Looking into examples of reasonably recent attacks that fall under terrorism, hacktivism or are conducted by non-state actors, there are plenty to choose from. In February 2021 (Vera et al., 2021), a hacker gained access to a water treatment plant in Florida and attempted to change the level of a chemical compound added to the treated water by more than a hundredfold. It is unknown who the perpetrator was and if they were located locally or even within the United States. This is a real issue with critical infrastructure connected to the Internet, as they are now opened up to threats from all over the Internet. Another attack on a water treatment plant was conducted a month earlier, in January 2021, in the San Francisco Bay Area (Collier, 2021). In this attack, the hacker used the password and username from a former employee's TeamViewer account. This allowed the hacker to access the plant, as TeamViewer was set up to allow remote working and delete several programs used within the facility. There were no reported incidents. This was a hack that required little knowledge, which shows there is not always a need for extensive knowledge of hacking to gain access to critical infrastructure. Cyber security lies with all employees and one weak point can have significant effects.

The United States acknowledges the importance of cyber security, as President Biden signed an executive order (The White House, 2021) to improve upon it throughout the nation. It mentions the importance of collaboration between the Federal Government and the private sector, which is key in improving security. As we all use cyberspace and are more interconnected than ever, some private organisations are linked with government systems; there is a clear need to work together. A prime example of the intertwining of private and governmental systems can be found within the NotPetya attack on Ukraine, where private companies such as Maersk also shared in their impact (Ritchie, 2019).

The year 2014 witnessed notable instances of cyber terrorism. In a remarkable case, Sony Pictures experienced a significant data breach in connection to the film "The Interview". The breach was accompanied by threats of terrorism, which led to numerous theatres withdrawing the film from their screening schedules. This breach marked one of the first instances of a product being pulled from the market due to cyber terrorism threats Brill (2015). Subsequently, Sony's PlayStation and Microsoft's Xbox networks suffered a large-scale DDoS attack. These events underscored the emerging power and influence of non-state actors within cyberspace. The main motives for such cyber terrorism and cyber criminal

activities are typically financial gain, commercial espionage, and hacktivism. Cyberspace, in its interconnectedness, facilitates criminals and terrorists' operations, extending their reach beyond traditional boundaries. Notably, the potential implications of cyber terrorism extend beyond typical criminal actions.

In 2021, a particularly alarming incident occurred. An unidentified party manipulated the tracking data of two NATO warships in Ukraine, causing the Automated Identification System (AIS) to display the ships as being in a Russian-controlled naval base in Sevastopol, Crimea - nearly 300 kilometres from their actual location (Sutton, 2021). The seriousness of this provocation cannot be overstated; such misrepresentation could be interpreted as a threat to Russian sovereignty, potentially triggering severe political and military responses. This incident reveals the critical risks associated with our increasing reliance on technology within defence systems, demonstrating that any disruptions or falsifications in such data could bear potentially grave consequences. As such, the urgent need to secure cyberspace becomes apparent.

2.6 ICS Attacks

Over recent years, several high-impact attacks on ICSs have been carried out these are summarised in figure 2.2 (Miller et al., 2021). Some examples of ICS attacks referenced by Miller et al. (2021) that happened over the past 30 years are Salt River Project in 1994, Gazprom in 1999, Daimler Chrysler in 2005, Night Dragon in 2009, Rye Brook Dam in 2015 and Triton in 2017. Within this section, we present three important case studies of such attacks to highlight the potential impact that an attack on an ICS device can have in the operation of industrial systems. We have selected these three attacks for their distinctiveness, and the level of coverage Stuxnet and the Ukrainian attack have received. Stuxnet being an attack that originated from within the organisation, the BlackEnergy on the Ukrainian energy systems which exploited MS Word vulnerabilities and used a known piece of malware and Wolf Creek which is a great example of how appropriate security measures are important to mitigate the possible effects of an attack.

It is important to consider the nature of the adversaries faced within the environment systems are deployed in to achieve effective security. As this thesis uses CNI as a focus the work is influenced by the attacks that have been targeting ICSs in these environments. Therefore, the shift of attacks from insider to organised groups and nation-states which are more knowledgeable is important to consider when looking at honeypots in this sector. The three attacks discussed give an overview of the impact attacks on these systems can have and in the case of Wolf Creek how these environments previously were protected.

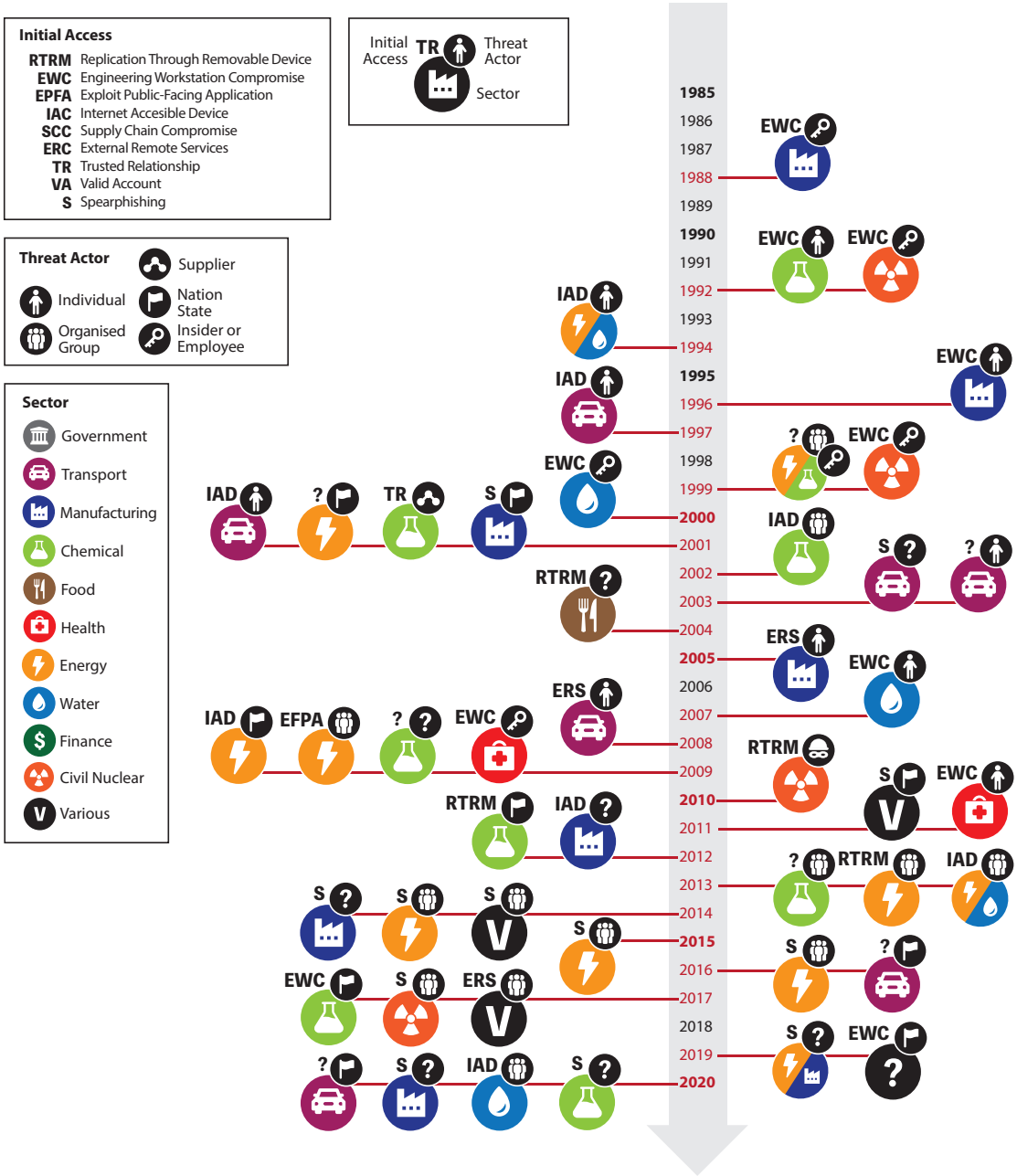


Fig. 2.2 Timeline of Attacks Against ICSs

2.6.1 Stuxnet

Stuxnet is widely recognised within the cybersecurity community and viewed as one of the most well-known ICS-focused cyber attacks. After the discovery of Stuxnet, it has been stated the world has entered in a new area of warfare and a pivotal moment in cyber security (Langner, 2011). Stuxnet has been used as an argument to improve cybersecurity, to question the current international laws regulating this space (Richmond, 2011), and to explore the future of warfare (Farwell and Rohozinski, 2011).

In 2010 Sergey Ulasen (Kaspersky, 2011) discovered malware that targeted Iranian nuclear facilities, which is widely suspected to be carried out as a joint military attack by the United States and Israel (Nakashima and Warrick, 2012). Nevertheless, like most cyberattacks, attributing it to a party is difficult (Farwell and Rohozinski, 2011). Unlike other pieces of malware seen before, Stuxnet was much more complicated and did not have any intention to steal data but instead had the objective to destroy a physical target (centrifuges) and delay the Iranian nuclear program (Collins and McCombie, 2012; Langner, 2011). This made the security world aware that cyberattacks can impact the physical and virtual worlds alike. Further, according to Farwell and Rohozinski (2011), Stuxnet has been the first malware of the 'fire and forget' generation, as it was designed to work in a quasi-autonomous manner. This increased the spread rate and also lowered the control the adversary has over it. The initial infection happened via a USB drive that was plugged into a machine in the facility. Then the worm spread automatically over the local network or USB drives connected to the systems with the ambition to further infect Windows computers on the network. Stuxnet exploited four zero-day vulnerabilities (Kushner, 2013). Worldwide, it is believed there were around 100,000 systems infected by the dropper (Langner, 2011).

2.6.2 BlackEnergy

In December 2015 it was discovered that the BlackEnergy malware was used to attack electricity distribution companies across Ukraine, which resulted in power outages that left more than 225,000 people without electricity (Department of Homeland Security, 2016; Khan et al., 2016). However, this was not the first or only time BlackEnergy was used in an attack. Within the United States, an attack on critical infrastructure using BlackEnergy could have had disastrous effects on the country, if gone undiscovered (ThreatStop, 2016).

In total there are four known versions of BlackEngergy. BlackEnergy version 3 (BE3), which was used in the Ukrainian ICS attacks, exploited vulnerabilities in Microsoft Office and propagated through Microsoft Word documents via spear phishing (US-CERT, 2016), and eventually managed to target the breakers of seven substations (Khan et al., 2016). Due

Feature	v1	v2	Lite	v3
Plugins		✓	✓	✓
Denial of Service	✓	✓	✓	✓
C2C Controller	✓	✓	✓	✓
Anti Virus Obfuscation	✓	✓	✓	✓
Kernel Rootkit			✓	✓
Bypass Driver Signing				✓
Reside in Memory				✓
Detect Virtual Environment				✓
Detect Countermeasures				✓

Table 2.1 Key Improvements of BlackEnergy

to the ongoing political dispute between Ukraine and Russia, it is suspected that the Russian State sponsored the attack, although such involvement has not been proven (Cherepanov and Lipovsky, 2016).

BlackEnergy is a notable example of how malware evolves over time, rendering traditional defences inefficient as the malware evolves to evade new security measures. Since the first version, it has evolved into a complex multi-purpose piece of malware. Version 2 expanded the espionage, spam and fraud capabilities significantly, and used a modular design which allowed adversaries to use plugins to customise the attack to specific targets (Khan et al., 2016). The latest version (BE3) simplified the method that is used to deliver the malware payload (ThreatStop, 2016). Further, it expanded the functionalities it had to evade detection and used different communication protocols. An overview of the BlackEnergy versions and their evolving capabilities can be found in Table 2.1 (Khan et al., 2016; Miller et al., 2021).

2.6.3 Wolf Creek

Unlike the Stuxnet and BlackEnergy attacks, the attack on the Wolf Creek Nuclear Operating Corporation (2017) caused no disruptions to the facility itself. As with other nuclear power plants, the operational systems are not part of the business network, and the ICSs are not connected to the Internet (Caravelli, 2019). This shows that an ICS environment that is separated from the IT network is better protected, however, with the current trend this separation is seen less and less. To gain a foothold in the network emails containing malicious documents were sent to senior industrial control engineers, through which the adversaries, supposedly, wanted to map the network for further attacks on the facility (Perlroth, 2017).

Despite the increasing awareness from governments and international agencies, the attack against the Wolf Creek plant highlights the challenge of tackling such threats. Around the same time as the Wolf Creek attack, a dozen other U.S. power plants were breached by adversaries (Riley et al., 2017). While in this case, none of the systems that are part of the manufacturing zone were compromised, there was still a severe threat. If one of the infected devices were to be connected to the network controlling the manufacturing zone, the malware could have spread to the ICSs and caused catastrophic failure. Therefore it is of uttermost importance that all systems connected to the facility are sufficiently protected and monitored. Once malware is spotted in the network, proper acts of mitigation have to be taken as soon as possible to prevent further breaches.

2.6.4 Analysis of Stuxnet, BlackEnergy and Wolf Creek Attacks

As visible within the discussed ICS attacks, there are many ways for adversaries to gain access to a system. Perimeter defence is, although useful and necessary, ineffective against a range of attacks. There is a need for improved security that goes beyond the usage of traditional tools and devices such as anti-viruses and firewalls. These systems will have to adapt to the tools adversaries use, similar to how adversaries adjust to new security mechanisms. Currently, security is continuously catching up.

Attacks like Stuxnet leveraged an accomplice to physically enter the plant and plug a USB drive into a system, which circumvented perimeter security measures and allowed the malware to spread through the network without being detected. Due to its approach, the detection became even harder, as there were no abnormal patterns or traffic from outside the facility on the network. Only deep within the network traffic, there would have been evidence of suspicious code being transmitted. BlackEnergy introduced the word to yet another type of malware, a modular form that can be modified for a specific attack.

The impact of BlackEnergy is undeniable, and its continually evolving nature poses substantial security threats. As mentioned before, traditional security software relies on signatures and constant updates when new forms of malware are detected. Continuous monitoring of application behaviour and network traffic provides a certain level of security, but once malware intrudes the network, it can be challenging to remove it altogether. An effective way to limit the potential impact of an infected system is by blocking connections to the Command and Control (C&C) server, either by quarantining the system or by limiting outside communication. Nonetheless, identifying the new malware variants as quickly as possible is imperative in defending against such attacks. The MS Word vulnerability exploited in the Ukrainian power plan attacks was leveraged by using spear-phishing, for which the security lies with the end-user (Hong, 2012). Traditional security measures are

Characteristic	Stuxnet	BlackEnergy	Wolf Creek
Windows Vulnerability	✓	✓	✓
Phishing		✓	✓
Zero-Days	✓(4)	✓	✓
Propagated Internally	✓	✓	✓
Originated on the Internet		✓	✓

Table 2.2 Comparison between Stuxnet, BlackEnergy and Wolf Creek

generally ineffective against this form of attack, and the infection of the system on which the file is opened is nearly unavoidable. The spread of the infection through the network was, from that moment, imminent. The attack on the Wolf Creek power plant also used spear-phishing to get into the network, but luckily the threat was limited due to other security measures in place. However if one of those infected computers were to be connected to the manufacturing zone, it would have spread nonetheless. A general overview of differences can be seen in Table 2.2 which highlights some of the aspects used in each of the three ICS attacks and provides and indication where we have to look to better defend ourselves against them.

There are several legal requirements companies have to adhere to regarding the security of their systems. One of these is the European Union's NIS Directive which is discussed in the next Chapter. Alongside these legal requirements, there are several international guidelines that can be followed and certifications that can be obtained to prove an organisation has taken the necessary steps to protect its systems. However, these do not stem from legal requirements they can be used for compliance with the legal obligations.

Based on these attacks we can also identify a number of security requirements that could have assisted with the mitigation of these attacks. Especially more active security techniques could have been included. For example:

- Non-perimeter security mechanism
- Continuous monitoring system
- Device-level detection system
- Identification of bespoke vulnerabilities

These requirements are not exhaustive, but the ones identified above find their way into this thesis within the later chapters.

2.7 Summary

Within this Chapter many aspects of critical infrastructure have been discussed, particularly how these relate to State operations in cyberspace. This more social science perspective is important to consider when working on security measures for these environments. Due to their particular nature and the environment-specific systems deployed, adversaries generally have more knowledge than in traditional IT environments where *script kiddies* are abundant. We have also discussed the critical nature of the systems and organisations this thesis will focus on, providing malicious state actors with an alluring target. These operations have been observed in the Stuxnet and BlackEnergy attacks and many others. As can also be seen in the call for regulation, and how not all countries are willing to introduce tighter international laws and regulations regarding the usage of cyberspace. Currently, cyberspace provides them with a less regulated and very anonymous environment to conduct operations against other nations, which is appealing. Throughout this thesis, it is important to keep these aspects in mind, as they do feed into the approach we must take to implement security measures in critical infrastructure and industrial control system environments.

Chapter 3

Honeypots and ICS

The previous Chapter introduced some of the more social aspects of the critical infrastructure environment, which this Chapter builds upon. As stated previously, we cannot solve the security without asking ourselves who we are securing against, and what the requirements and boundaries are within this space. This Chapter introduces the more technical aspects of honeypots and ICS security and analyse how both can fit together, especially related to the regulatory requirements. We discuss how honeypots are currently deployed and how we can build and learn from those deployments. Concepts such as the defence-in-depth model are discussed with relation to ICS security, and framework that provide assistance to implement important regulation such as the UK Cyber Assessment Framework are utilised to connect the benefits of honeypots to the regulation. We aim to show that honeypots can not only provide benefits to the security of a system or network but also further strengthen the adherence to necessary regulations. This is an important aspect to consider as it bolsters the argument for their incorporation within critical infrastructure.

3.1 Honeypots

Deceptive cyber defence is a well-understood form of cyber security systems and are designed to mislead and manipulate attackers into thinking they're gaining access to sensitive systems, when in reality, they're interacting with a controlled and monitored environment. Honeypots are a well-known form of cyber deception and are often described as a computer system that is set up to act as a decoy to lure cyber attackers and to detect, deflect, or study attempts to gain unauthorised access to information systems. They are designed to mimic real systems with seemingly valuable data, but are often isolated from the main network. Honeypots are usually designed to be more easily exploitable and can come in different variants; they can be either virtual or physical (The Honeynet Project, 2001). Their intent is to distract the

attackers from the real systems and lead them towards the honeypot instead. This is done in an attempt to either deceive the attacker and luring them away from the real systems. Hoping attackers waste time and resources engaging with the honeypot, which can deter and slow down their operations. Or encourage them to reveal information about their tools, techniques, and strategies. Helping the security team anticipate and prepare for future attacks.

3.1.1 Honeypot Purposes

When deploying honeypots their purpose is imporant to consider, and mainly we refer to two different purposes; research or production. The goal of each honeypot is still to lure attackers into targeting them, however, the main goal of a honeypot changes depending on its purpose. For research honeypots, that are Internet-facing, and are deployed with the main goal of gathering information for research purposes. This is different than production honeypots which are usually not directly accessible and are deployed inside an organisational network to improve their security. To refrain from entering the legal area of entrapment, honeypots need to be deployed and configured with care. Within an organisation we would expect an aim to direct attackers to a honeypot once they are inside of the network. For research honeypots it is sufficient to make the honeypot accessible from the Internet to capture interest. When honeypots are compromised, they can be used to generate alerts or to deceive the attacker by diverting exploitation efforts away from the systems that need to be protected.

Therefore, the value of the honeypot is determined by the number of attacks it receives (Zhang et al., 2003). Honeypots that are actively attacked provide the most valuable information, but even when they are not exploited, honeypots can indicate if a network is being actively targeted. To achieve valuable activity, it is essential to both lure attackers to the honeypots by introducing vulnerabilities whilst also maintaining a reasonable level of security to resemble an operational system (Rowe et al., 2006). When a system is significantly less secure than others within the same network, it can be seen as an indication of a potential honeypot.

Data gathered through honeypots can be used in many ways. For example, they can provide useful data which can be used to create a timeline of an attack. This is important for accurate threat intelligence, and generally hard to construct (Caravelli, 2019). By implementing honeypots and luring attackers away from real infrastructure, an organisation can both improve its security through the data collected (Caravelli, 2019), and reduce the usage of resources on business systems. Therefore, monitoring of traffic to and from honeypots, and the attackers' actions within them, are crucial aspects of honeypot operations. Due to the nature of a honeypot, by not performing business operations, all traffic to them can be considered malicious. However, this lack of real operational purpose within the network

makes it harder to deceive attackers, as there is no actual active traffic between them (Rowe et al., 2006). Additionally, the level of interaction that an intruder is permitted to have with the honeypot can affect the behaviour of the intruder and therefore, the volume of collected attack data. Generally, the goal of honeypots and their detection capabilities is to gather data to feed into the protection of the network or systems.

3.1.2 Honeypot Categories

In general, honeypots are categorised based on their level of permitted interaction to high-interaction and low-interaction honeypots. A third category, medium-interaction honeypots (Mokube and Adams, 2007), does exist but its characteristics lay close to a low-interaction variant. Low-interaction honeypots pretend to be a specific device such as a Programmable Logic Controller (PLC) and mimic its functions, they run on a standard operating system (e.g. Ubuntu) and provide limited interaction for attackers (Chamotra et al., 2011).

For instance, the *Honeyd* honeypot (Provos, 2003) provides a TCP/IP stack emulator which allows an attacker to send network requests which it responds to. However, the attacker cannot have further interaction with the other parts of the system, such as the operating system. This might result in increased identification of the system as a honeypot, and shorter interactions with the system. Using an approach like PCaaD (Green et al., 2021), would also not be possible on a low-interaction honeypot. In contrast, high-interaction honeypots are, in essence, the same device as the one in the operational network, allowing attackers to interact with every aspect of the machine (Spitzner, 2002). These honeypots are generally less easy to identify and allow attackers to perform more actions and increase their time and interaction with the system.

The benefit of high-interaction honeypots is that attackers are less likely to identify them as honeypots, and they can provide considerably more data from the attack (Chamotra et al., 2011). On the other hand, high-interaction honeypots demand significantly more resources, and they entail the risk of the attacker taking over the system due to the high level of interaction they permit (Vetterl and Clayton, 2018). Within an ICS environment, deploying a high-interaction honeypot entails the usage of a real PLC or other ICS device. These devices are expensive, and a single device does not accurately represent a real ICS deployment. Therefore, to achieve full high-interaction multiple devices have to be deployed to transfer data between them.

Improving upon some low-interaction honeypots by integrating additional characteristics of a real system and creating a so-called medium-interaction honeypot, can provide more data whilst still entailing a lower level of risk. However, all features on them are simulated as

well. Therefore, we consider such medium-interaction honeypots as a sub-category of low-interaction honeypots. For example, they can simulate responses of a specific service, such as a web server. They can lure attackers that scan for particular vulnerabilities or exposed services, without possessing the risk of being exploited. However, medium-interaction honeypots do not run a full operating system (unlike those that are high-interaction) and therefore the data available about potential attacks is more limited (Spitzner, 2002). Looking specifically at an ICS environment, it can be challenging to have a simulated system perform close to a real system. Particularly when taking into account an device, like a PLC, does require input from another system to display data. This could potentially be circumvented by providing the system with static data, which is easy to identify by an attacker. More dynamic data is possible, but requires more effort as these data have to be somewhat realistic, e.g. a water tank cannot go from 100% to 0% in one second, and if there would be an identifiable pattern it could also be detected. Therefore, due to the nature of ICS and if the goal is to capture useful data from knowledgeable attackers, the deployment of a honeypot has to be extremely realistic. Overall, medium-interaction honeypot implementations tend to be less frequent than high- and low-interaction variants (Paralax, n.d.). Nonetheless, it is essential to note that the distinction between low- and medium-interaction is not always clear and largely depends on the context of the simulated environment.

3.1.3 Honeypot Characteristics

An overview of general honeypot characteristics we have determined can be found in Table 3.1 based on a 1 to 3 number scale. These characteristics range from resource usage, and risk involved in deploying them to the knowledge required to set-up and operate the honeypot, and ease of detection.

When talking about risk, we talk about the attacker being able to leverage the honeypot and use it against us. A low level of risk implies that the attacker has limited ways to leverage the honeypot and when correctly deployed, should not be able to use it to pivot into the network. A high level of risk means there is a possibility of the attacker taking control of the honeypot and using it as a way into the network.

Data capturing scales from basic to intermediate and comprehensive. Basic refers to the limited amount of information captured by the honeypot; this tends to be limited to the information included in the IP packet. Intermediate improves upon basic by having the capabilities to capture more IP packets, as the attacker can perform more interactions. Building further upon this is comprehensive, where the attacker has the opportunity to interact with the whole system and can perform any actions that are generally performed in an attack including uploading of data and interacting with other systems on the device.

Level of Interaction	Low	Medium	High
Risk	1	1	3
Data Capturing	1	2	3
Resource Usage	1	1	3
Simulation	1	2	N/A
Required Knowledge	1	2	3
Detection	1	1 - 2	2 - 3
Cost	1	1	3

Table 3.1 Comparison of Different Levels of Interactions of Honeypots and their Characteristics

Resource usage refers to the resources needed to operate a honeypot. Low resource usage means the honeypot can be deployed in a virtual environment; this allows for multiple separated honeypots on the same physical system. High resource usage refers to the need of a physical device set up for one specific honeypot, and typically there is also a need for more software on the system (e.g. keyloggers) and other monitoring systems on the network (e.g. IDS).

Simulation relates to the level of simulation, which can range from limited simulation of services (basic) to a more comprehensive level of simulation, which allows for more interaction (improved). To deploy honeypots, there is a knowledge aspect involved. For a low-interaction honeypot, this is relatively low, as these can generally be installed as an easy to deploy package. An advanced level of knowledge refers to the possible need to adapt and improve the simulation of the honeypot. For a high-interaction honeypot, we would advise a high level of knowledge due to the risks involved, and the more do-it-yourself actions involved in creating a honeypot from a real device.

Detection of the honeypots concerns the ease of detection, which can range from easy to normal for simulated services (depending on the comprehensiveness of the simulation), and normal to hard for high-interaction honeypots (depending on the deployment). To further clarify, for a high interaction honeypot, it is important to strike a balance between security and included vulnerabilities to lower the ease of detection.

Finally, the cost of the honeypot is a combination of the resources required to operate the honeypot and the resources necessary to deploy the honeypot. For low- and medium-interaction this is low as they only require a virtual environment and the honeypot software (which tends to be open-source). A high-interaction honeypot incurs more costs to purchase the device, required software to monitor the device and other systems, such as an IDS.

3.1.4 Conpot

There are a plethora of different honeypots available on the Internet that are easy to download and deploy (Paralax, n.d.). Particularly low-interaction honeypots are readily available to deploy as these do not require as much setup as a high-interaction one. For ICS, Conpot (MushMush Foundation, 2018) is one of the most popular. Conpot is a low-interaction honeypot that is specifically designed to mimic ICSs and has been widely used within research and industry. This honeypot plays an important part in this thesis as a standard of low-interaction ICS honeypots, for deployment, identification of honeypots and more.

3.1.5 Honeytokens

In the context of deception-based security honeytokens are similar to honeypots but focused on data itself. Honeytokens are also used to to deter intruders and uncover unauthorised system access or data management by provding data as bait or a trap (Petrunić, 2015). In essence, honeytokens are data points that are processed as if they had genuine value for the organisation even though they don't, serving as a ruse.

These tokens are put strategically throughout an organisation's IT infrastructure and can take many different forms, including as database entries, cookies, fictitious user credentials, unused IP addresses, or false files. A honeytoken's main function is monitoring; any interaction with one is almost certainly malicious because they are not designed to be accessed during normal operations. An alert is set off if a honeytoken is manipulated or accessed by an attacker, allowing the organisation to quickly react to the potential threat. To be clear, honeytokens are detection and early warning systems rather than preventative measures.

Although honeytokens do not form a focus point of this thesis, these do show some of the different possibilities that are possible within deceptive cyber defence.

3.2 ICS Security

We have already mentioned that existing ICS security techniques mainly focus on availability and the IT environment around the infrastructure. However, there have been calls to expand the focus from just availability to the security of the systems against malicious cyberattacks (Cárdenas et al., 2008). Previous studies have found that honeypots can be used to improve the security of Supervisory Control and Data Acquisition (SCADA) systems (Disso et al., 2013), which are a subset of ICSs. SCADA systems are mainly used to collect data and monitor ICS environments and reside in level 2 of the Purdue Model, just above PLCs.

But generally, honeypots are not often considered by ICS security researchers. One of the main drivers for the security of ICSs and critical infrastructure is standards and guidelines in conjunction with regulation. Standards like ISO 27019 (ISO/IEC, 2017), IEC/ISA 62443 (IEC, 2019) and NIST SP 800-82 (Stougger et al., 2015) are generally used within an ICS environment. Therefore, these documents are used by industry when deploying and securing their ICS infrastructure.

It is clear from looking at the ICS security space that there has been a lack of built-in security. This is understandable when looking at it from the perspective that these systems were not designed to be accessible aside from engineers working on them, but as stated before, this has changed in recent times. Bhamare et al. (2020) in their 2020 survey have mainly focused on Machine Learning (ML) to improve upon ICS security. These approaches range from risk assessment based on ML and ML to detect malicious communications within the SCADA environment to Cloud-based computing for attack mitigation. However, to achieve potent ML-based approaches, a significant amount of data is needed to train the system or feed into the system. We feel that honeypots can greatly improve this as, if deployed within the organisation, they can provide a lot of useful real-time data of threats inside the network. When deployed outside the organisation or deployed with a research focus within the organisation they can provide useful general data.

Generally techniques applied within the ICS space can be categorised as focusing on the architecture, strategy, attack modelling, attack detection and attack categorisation (Ani et al., 2018). Especially with the trend to expose these networks and systems to the Internet, thinking about these six categories becomes even more important. All these areas have to work together and feed into each other. Like within traditional IT environments, security should be encompassed throughout the design. Relying on bolt-on security, like SAv5 for DNP3 (Crain and Bratus, 2015; Cremers et al., 2019), should only be viewed as temporary fixes and not be relied on for an extensive period. The inherent vulnerabilities within ICS protocols such as DNP3 (DNP, n.d.), Modbus (Modbus, 2012) and IEC 61850 (International Electrotechnical Commission, n.d.) should be tackled with security in mind from the design phase onward.

There is an immediate need for research and development within the ICS security space. Research done with ICS testbeds such as described by Green et al. (2020) will be important for the future of ICS security and the security of our society. ICS security risks come from a socio-technical angle and requires an understanding of how stakeholder decisions from all levels impact the security of these devices. There is a real complexity that stems from the combination of devices and systems within critical infrastructure environments (Rashid

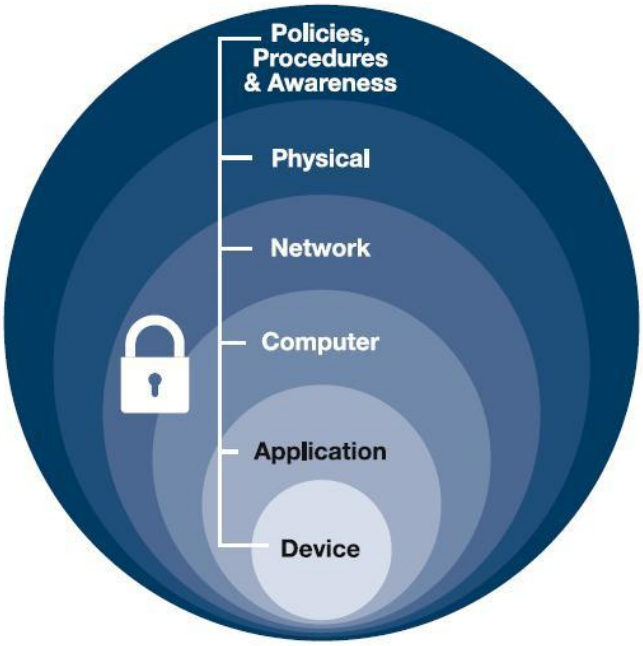


Fig. 3.1 Example of Defence-in-Depth Representation (Ryan, 2022)

et al., 2019). The introduction of honeypots would enable further evaluation of threats and the adversaries behind them.

3.2.1 Defence-in-Depth

Defence-in-depth has become prevalent within ICS security, mainly due to the lack of built-in security features described earlier. Approaching security from this angle has been common within IT systems and has a long track record in preventing assets from issues such as theft both for physical and virtual assets (Smith, 2003). The principle is based on protecting assets with multiple successive barriers such as the usage of multiple firewalls on the network in combination with host-based security systems as well. However, this also incorporates other security methods such as policies and application security systems. This strategy is often shown by using layers, as it focuses on multiple security measures within the network and system. An example of this representation can be seen in Figure 3.1[1], which shows the device that is secured is protected with several layers of security measures.

Within ICS there is a lack of security on the device therefore the security has to be implemented on the other layers. Reflecting back on the Purdue model (Figure 2.1) and the airgapping of these systems from the IT environment we can see that the network layer is an

[1]Image under Creative Commons Attribution 4.0 License

important part of the security of OT environments. Hence the usage of network firewalls and intrusion detection system (IDS) designed for OT environments is prevalent (Hu et al., 2018). However, these systems are generally reactive and can not necessarily identify novel attacks. An IDS requires to understand changes in the environment and given the focus on safety uncertainty is frowned upon. The result of this can be seen in systems not always being capable of raising appropriate alarm such as seen with Process Comprehension at a Distance (PCaaD) (Green et al., 2021). In addition, as identified by (Miller et al., 2021) insider attacks are also common within critical environments and social engineering attacks (Green et al., 2015) are difficult to detect once they are successful.

Therefore, to be able to identify and defend against these new systems need to be used. Honeypots present an interesting candidate for this as they can provide pro-active data before systems are infected and are capable of presenting real-time data when an attacker is in a network where honeypots are deployed. Building upon defence-in-depth, honeypots are proven to be able to be a part of this (Weiler, 2002). We explore exactly how honeypots could be helpful within ICS attacks and to obtain data further in the thesis.

3.2.2 Shodan

As part of ICS security it is also important to gauge the systems that are available online, especially honeypots. There are many scanners available that scan the Internet on a regular basis, such as Shodan (Shodan, 2020) and Censys (Censys, n.d.). Shodan forms a key part of this thesis as the Internet scanner of choice.

Shodan is a popular Internet scanner that weekly scans the publicly accessible systems on the Internet over different protocols Matherly (2015). It allows users to search through their database to find systems running on IP-address or IP ranges. It can also list vulnerabilities detected on the system and the type of device running (e.g. Industrial Control System). Shodan can be used by network administrators to monitor their network but is also used by actors with less legitimate purposes Bada and Pete (2020). There is also a Shodan script for the popular penetration testing tool Nmap which integrates Shodan results within the Nmap scan.

3.3 Frameworks, Guidance Documents and Standards Relevant for ICS Security

Despite the increasing cyber threats against critical ICS infrastructures, many ICS operators appear hesitant to adopt security standards and best practices due to increased cost and

management overhead (Knowles et al., 2015). Given the criticality of ICS facilities to national security, many governments decided to mandate security measures and regulate their implementation through legislation, such as when the US attempted to pass the Cybersecurity Act 2012 (Harrop and Matteson, 2015). However, regulation remains the most common form. For example, the US Cybersecurity and Infrastructure Security Agency (CISA) operates the Chemical Facility Anti-Terrorism Standards (CFATS) (Klessman et al., 2011) to regulate the security of high-risk chemical facilities, while the security of Nuclear Facilities is regulated by the policies of the US Nuclear Regulatory Commission (NRC) (US Nuclear Regulatory Commission, 2010). Such regulation often describes the high-level security requirements and procedures, but not the actual techniques through which security measures should be realised.

Due to this heavy level of regulation within the environments this thesis is focused on, they need to form a key part when designing any security mechanism. Including honeypots. Therefore, it is important to consider how security systems can help organisations to adhere to these regulations and their relevant standards and guidance. This not only provides a strong argument why organisations should implement systems, but also how useful these systems can be in general. As such, we must understand the most comprehensive and well used of these regulations.

In this section, we provide an overview of some of the most comprehensive international ICS cyber security best practices, and we explore where honeypots are included in these. We give an overview of US, Canadian, Spanish, French and German best practices and give an in-depth analysis of the UK Cyber Assessment Framework (CAF). Although these are best practices, several of these guides and frameworks are used as a baseline for adherence to regulations such as the EU NIS Directive, which is implemented by the UK NIS Regulation. Afterwards, we introduce several well-known guidelines of organisations that are often referred to within governmental documentations.

3.3.1 UK Cyber Assessment Framework

In an effort to harmonise cybersecurity regulations across the European Union (EU), the European Commission introduced the Network and Information Security (NIS) Directive (The European Commission, 2016). The NIS Directive is an EU-wide legislation on cyber security, which means that every EU state has to adopt it in their national legislation. Several countries adopted a different strategy for their critical infrastructure (Austrian Government, 2013; NCSC, 2019) since cybersecurity needs differ from other sectors of the economy. It was adopted in 2016, and all members had to transpose it by 2018 (The European Parliament and The Council of The European Union, 2019). The UK has produced the Cyber Assessment

Framework, one of the most thorough guidance documents related to the NIS Regulation, which is the UK implementation of the NIS Directive. Although the CAF is very extensive, it does not discuss the application of honeypots as a defensive technique.

The UK Cyber Assessment Framework (NCSC, 2019) is compiled by the UK National Cyber Security Council (NCSC), to assess the security of critical national services and infrastructure and functions as guidance for organisations to adhere to the UK NIS Regulation. The framework further notes that "cyber threats to UK CNI represent an area of particular concern for the government, and consequently the cybersecurity and resilience of the thirteen CNI sectors is a high priority for the NCSC." One of the examples is the civil nuclear sector, which has its own cybersecurity strategy (Department for Business. Energy & Industrial Strategy, 2017), SyAPs (ONR Security Assessment Principles (SyAPs), n.d.) and Technical Assessment Guides (TAG) such as the Preparation for and Response to Cyber Security Events TAG (Office for Nuclear Regulation, 2021). Although the EU NIS Directive does not require this, the UK still puts emphasis on the importance of the critical infrastructure sector. This shows the commitment of the UK to assess and advise the industry on their cyber security and demonstrate the critical impact potential incidents might have on the country.

The CAF is divided into four main objectives, which are, in turn, broken down in several principles.

Managing Security Risk

This objective provides companies with information on how to manage cybersecurity risks. It assists them to have appropriate policies, structures and processes in place to mitigate and manage risks to the systems. It is further divided into Governance, Risk Management, Asset Management and Supply Chain.

A definition for risk has to consist of multiple elements, as risk relies on several factors within cybersecurity. The National Institute of Standards and Technology (NIST), one of the leading standard organisations regarding cyberspace in the US, defines it as "A measure of the extent to which an entity is threatened by a potential circumstance or event, and typically a function of: (i) the adverse impacts that would arise if the circumstance or event occurs; and (ii) the likelihood of occurrence." (NIST, 2012) The International Organization for Standardization (ISO) and The International Electrotechnical Commission (IEC) state within the ISO/IEC 31010:2019 standard (ISO and IEC, 2019): "risk is often described in terms of risk sources, potential events, their consequences and their likelihoods." And the UK NCSC defines risk as: "possible future outcomes that we can describe in terms of their chances of occurrence, and what impact they would have on us." (NCSC, 2018) These three definitions put emphasis on the impact or consequence of an event and its likelihood.

Reducing risk should, therefore, focus on reducing the probability of an event occurring and the potential impact it might have on the organisation.

Protecting Against Cyber Attack

Within the second section of the CAF Principles and Guidance, the objective is to protect the business against a cyber attack. It aims to have proportionate security measures in place to protect the systems. This objective is further broken down to Service protection policies and processes, identity and access control, data security, system security, resilient networks and systems and staff awareness and training.

When securing a system, the NCSC discusses three main ways vulnerabilities emerge: flaws, features, and user errors. Keeping software up to date is vital in limiting vulnerabilities through defects in the program. As mentioned before, it can be difficult keeping critical infrastructure up to date as powering off the system can have significant ramifications (Cavusoglu et al., 2008). Protection against cyber-attacks does not stop at the security of data and systems but also covers the actions after a system fails or is compromised. Preparations to ensure critical business functions can continue to have to be in place when accounts are compromised, or systems have been infected.

Detecting Cyber Security Events

Objective C of the Cyber Assessment Framework covers the capabilities an organisation regarding cybersecurity event detection. When all security measures fail, the detection of the malicious user and his actions within the network or system as quickly as possible is key (Mukkamala et al., 2005).

An organisation should monitor the security status of its networks and systems in order to detect potential security problems. After the collection of logs and potential security problems, an organisation should use appropriate tools and analysis to detect any indicators of compromise within them. The NCSC outlines the continuous activity required to maintain the security of the organisation and an effective ongoing change within the operational security of an organisation is vital. Apart from monitoring, a proactive approach to discover cyber events is necessary. Flagging deviations from regular interactions, such as users logging in outside of working hours and unexpected traffic should be a trigger for further investigation.

Minimising the Impact of Cyber Security Incidents

The final objective set out in the Cyber Assessment Framework focuses on minimising any adverse effects a cybersecurity incident might have on the network or organisation. As incident are almost unavoidable, even with top-of-the-line security in place (Wells et al., 2014), restoring the regular operation of the business is vital to minimise the financial and reputation losses.

When an incident has occurred, aside from reporting it to the regulator, the organisation should take steps to understand the root causes and make sure suitable mitigating actions have been taken. The aim of investigating the cause of an incident is to be able to prevent the root cause on a business wide-scale rather than only patch the affected system itself.

3.3.2 US NIST Framework for Improving Critical Infrastructure Cybersecurity

The US National Institute of Standards and Technology (NIST) published the first version of the Critical Infrastructure Cybersecurity Framework in 2014 and revised it in 2017 and 2018 to reflect the evolving cybersecurity landscape and incorporate feedback from organisations (Barrett, 2018). NIST is a well-known organisation for standards and guidance, and several of its guides are referenced in the CAF. Although NIST implementation is not subject to enforcement, organisations such as the North American Electric Reliability Corporation (NERC) does provide Critical Infrastructure Protection (CIP) standards that are enforced. Many of these references back to NIST and their Special publications for further details on actions an organisation can take to incorporate the standards. The NIST Cybersecurity Framework is addressed to the organisations that rely on networked devices, including the sectors of Information Technology, Industrial Control Systems (ICS), Cyber-Physical Systems (CPS), and the Internet of Things (IoT). In addition to the NIST Framework, the Department of Homeland security has also published a recommended practice to improve ICS security (Fabro et al., 2016).

The NIST framework focuses on five functions:

- *Identify*: Understanding of business needs, critical resources and risks.
- *Protect*: Implementation of necessary protection mechanisms to safeguard critical operations and services.
- *Detect*: Detection of anomalous activities and attacks, identification of attack targets and methods, and monitoring of external service providers to determine external threat vectors.

- *Respond*: Actions were taken to stop and mitigate the impact of potential security-related events.
- *Recover*: Recovery to normal operations and restoration of any services impacted by a cybersecurity incident.

While the main framework does not specify how the above functions should be implemented, it refers to NIST Special Publications (S.P.s) that provide implementation details. It is also interesting to note the similarity of these functions to the objectives set out within the CAF.

NIST SP 800-53 (Joint Task Force Transformation initiative, 2013) provides a comprehensive catalogue of tools that can be used to support the cybersecurity functions of federal information systems and organisations. The tools are divided into 18 different categories of security controls that can be used to satisfy the functions of the NIST

NIST recommends the use of honeypots only by specialised entities using non-operational equipment in highly isolated network partitions because potentially misconfigured honeypots can allow attackers to circumvent other security measures through lateral movement attacks. However, NIST SP-800-160 (Ross et al., 2019) on Developing Cyber Resilient Systems recommends maintaining a full-scale deception environment that encompasses honeypots, honeynets and decoy files. According to NIST, deception to create false targets combined with analytic monitoring to detect traffic to those targets could have hindered the 2015 attackers from opening the substation breakers and disrupt power distribution. Such a measure could have been implemented by developing honeypot Human-Machine Interface (HMI) screens integrated with an Intrusion Detection System (IDS) both for the O.T. and the ICS of the power plant. A similar deception strategy could have been effective in misdirecting the malware in the 2016 attacks from providing the intruders with a Command Line Interface (CLI) and interactive services on HMIs.

The NIST guidelines on the proper use of honeypots underline the importance of understanding the complexities of deploying honeypot-based defences in sensitive ICS. While deception and misdirection can be critical in slowing down catastrophic attacks or preventing them altogether, the potential risks necessitate meticulous planning and expertise.

3.3.3 Other National Guidance

Public Safety Canada ICS Cyber Security: Recommended Best Practices

Public Safety Canada released TR12-002 to provide SCADA and ICS professionals with both administrative and technical best practices related to the cybersecurity of industrial facilities (Public Safety Canada, 2012). Their best practices start by understanding the risks

an organisation faces, which includes cyber threats. To this extent, it is important to gain awareness of these threats and which actors are actively targeting the systems. The areas covered are similar to the ones found within the NIST Cybersecurity Framework and UK Cyber Assessment Framework.

Spanish National Cybersecurity Institute

The Spanish National Cybersecurity Institute or INCIBE, has published several guides for industrial control systems. One of them covers protocols and network security in ICS infrastructures (INCIBE, 2017), and another one describes the implementation of low-interaction honeypots for ICS security (INCIBE, 2019). The honeypot implementation guide focuses on the requirements and implementation of ICS honeypots. However, it is limited to low-interaction honeypots. We do feel it is a step in the right direction to introduce ICS professionals to the use of honeypots within their environments and hope for increased guidance from all appropriate bodies.

Portuguese National Cybersecurity Framework

The Portuguese National Cybersecurity Centre published their National Cybersecurity Framework to allow organisations to reach a mature level of cyber security (Portuguese National Cybersecurity Centre, 2020). It highlights the same five domains as the NIST guidance: identify, protect, detect, respond and recover. And similar to the UK CAF it identifies subareas such as risk management, asset management, monitoring, detection, response and recovery. Akin to INCIBE, it also mentions honeypots for anomaly detection.

ANSSI Managing Cybersecurity for ICS

The French Agence National de la Sécurité des Systèmes d'Information (ANSSI) published their guide to Managing Cybersecurity for Industrial Control Systems (Agence nationale de la sécurité des systèmes d'information, 2014). The purpose of the guide is to support organisations by providing good practices when implementing security measures. Within the guide, several myths are examined; one of those is that the isolation of ICS devices means they are protected. Similar to the other guides, it mentions areas such as asset management and risk analysis, monitoring and detection, and incident handling.

German Federal Office for Information Security k

The German Federal Office for Information Security or Bundesamt für Sicherheit in der Informationstechnik (BSI) has several resources for ICS security, which includes general

recommendations, recommendations for operators and recommendations for manufacturers (Bundesamt für Sicherheit in der Informationstechnik, n.d.). The general reference guide for ICS security is the ICS Security Compendium (Bundesamt für Sicherheit in der Informationstechnik, 2013), which establishes a general framework for the industrial sector. It acknowledges the mixture of ICS with traditional I.T. systems and the Internet as a significant change in the operation and security of industrial control systems. There is a focus on lack of monitoring, lack of awareness, malware, maintenance laptops and phishing.

3.3.4 Other Well-Known Cyber Security Standards and Guidance

The following are examples of other standards and guidance referenced within the CAF that are ICS focused or are generic guides for system security. We have selected these based on their relevance to the potential use of honeypots within the objectives of the discussed national guidance, which we explore in more detail later in this Chapter.

ISO 27001

The ISO 27000 certification range is one of the most known certifications within cybersecurity, and companies often pursue them. Within these, the 27001 (ISO/IEC, 2013) covers the information security management aspect. It has been designed to help organisations with the implementation and continuous improvement of their information security management system. When linking back to the CAF, the ISO 27001 standard fits in with objective A (managing security risks).

ISO 27002

ISO 27002 builds upon ISO 27001 and is designed to aid organisation in the selection of controls to implement an information security management system or to guide organisations into implementing standard security controls. It also focuses on the development of information security management guidelines within an organisation. ISO 27002 is referred to for objectives C and B of the CAF.

ISO 27019

As part of the ISO 27000 range of guidance, ISO 27019 (ISO/IEC, 2017) is based on ISO 27002 and provides guidance for process control systems that are used within the energy industry. These systems include PLCs, sensors, field devices, advances metering

infrastructure and many others that are used to control and monitor processes involving electricity, gas, oil and heat. It is used within objective B5 of the CAF.

IEC 62443-2-1:2010

The International Electrotechnical Commission's 62443-2-1 (IEC, 2010) standard focuses on the establishment of an industrial automation and control system security program. The IEC recognises the weaknesses in ICS due to the adoption of commercial off the shelf technologies, which tend to be more vulnerable to cyber attacks Jenney (2013). This standard can be used within objective A and B (Defending systems against cyber-attack) of the CAF.

NIST Information Security Continuous Monitoring (ISCM)

The focus of the ICSM (SP 800-137) (Dempsey et al., 2011) lays in maintaining the ongoing awareness of information security within the organisation; this relates to vulnerabilities, threats, and management decisions. The Information Security Continuous Monitoring publication fits under Objective C principle C1 of the CAF, which focuses on security monitoring.

NIST Computer Security Incident Handling Guide

The Computer Security Incident Handling Guide (SP 800-61r2) (Cichonski et al., 2012) is aimed at the handling of cyberattacks that even with proper security measures have been able to succeed. Incident handling is vital in reducing loss and destruction and mitigating exploited vulnerabilities. Linking this guide back to the UK CAF, it can be used within both principles of Objective D (Minimising the impact of cybersecurity incidents).

3.4 The Use of Honeypots in ICS Security

3.4.1 Honeypots in the Context of ICS Cyber Security Standards and Guidance

As explored previously, ICS attacks could have devastating effects. In the previous section, we have explored several pieces of ICS security standards and guidance, and this section further explores where honeypots can fit within these. Although we have surveyed several different documents, only NIST and the Spanish National Cybersecurity Institute mention honeypots, briefly. This is an indication that, particularly within an ICS environment, honeypots are yet to be implemented widely. This is surprising as honeypots can fit within many of the

objectives outlined and can provide a unique approach to security. Within this subsection we explore how honeypots can fit within areas mentioned within these guidance.

New approaches to the security of ICS that are able to predict and protect against new attacks are necessary. These approaches should rely on real-time data that can be used to detect malicious traffic automatically. This information can then be used to update firewalls, IDS, etc. A possible concept to gather the data is through the use of honeypots. A recent example of the capabilities of honeypots in an ICS environment is the ICS honeypot deployed by Cybereason, which alerted us of the dangers of multi-stage ransomware (Barak, 2020). The goal of the honeypot was to gather information on tactics, techniques, and procedures used by state-sponsored groups. As evaluated previously, there have been publications from governmental organisations which covers the use of honeypots for ICS. However, this guide focuses mainly on the deployment of low-interaction honeypots, which are generally easier to detect. We agree that low-interactive honeypots are generally used within production environments, however, we feel high-interaction honeypots and other honeypots that would generally be considered *research honeypots* could be beneficial within an organisation as well. Other guidelines from governments or international organisations do not include honeypots specifically, although they can fit within several areas.

In the example of NIST, Figure 3.2 breaks down the security functions and sub-functions of the NIST framework that we believe can benefit from the security controls linked to deception and virtualisation technologies related to honeypots. These security controls include *Concealment and Misdirection* to reduce the targeting capabilities of adversaries, and *Information System Monitoring* to detect events occurring both at the perimeter and within the protected information systems. Nonetheless, NIST does not include honeypot-based security controls in any of the three security baselines defined in the framework, even for high-impact information systems in terms of confidentiality, availability and integrity security objectives.

As stated previously, the NIST framework and the CAF have similar focuses, which are also found within the other national security guides we discussed earlier. We can also use honeypots to strengthen several of the objectives mentioned within these documents. An overview of how we have identified how honeypots can contribute to the CAF can be found in Figure 3.3. Unlike the NIST framework, the CAF does not directly mention honeypots. We will use the CAF as a guide to establish where honeypots can fit within regulation and guidance.

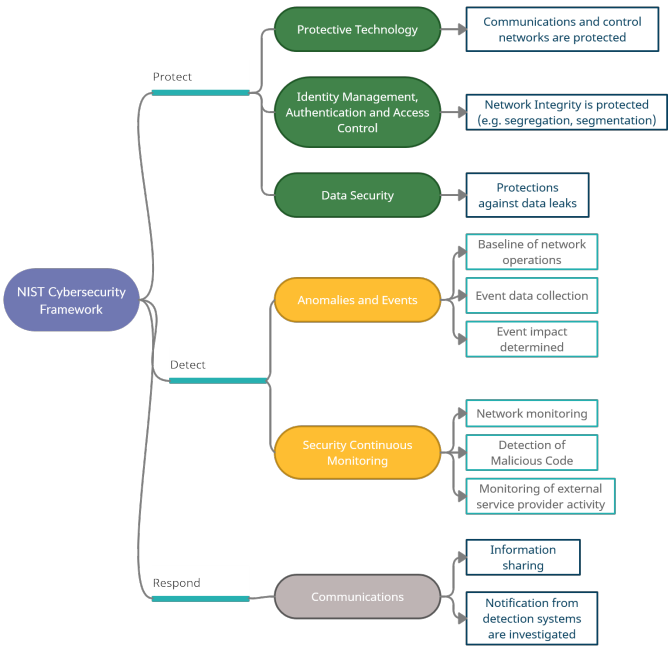


Fig. 3.2 The cyber security functions of the NIST framework applicable to honeypots

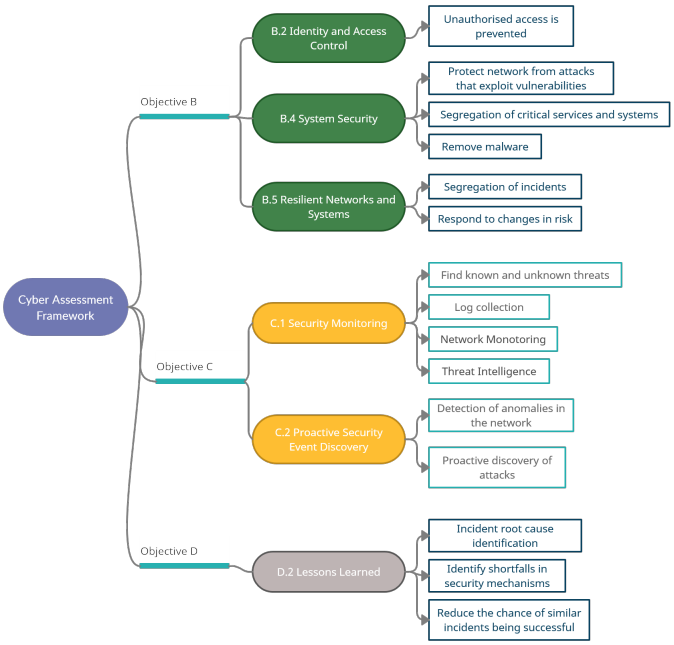


Fig. 3.3 The cyber security functions of the CAF applicable to honeypots

Managing security risk

The first section of the CAF, covering security risk, mainly focuses on policies and processes. Although honeypots can undoubtedly be part of these policies and procedures to, deploying them does not satisfy any portion of this objective. Therefore, we will not go in-depth on this objective.

Protecting against cyber attack

The second objective of the CAF and the protect function of the NIST framework cover security measures that are in place to protect the organisation against cyber attacks. Within the CAF, we have identified three sub-objectives within this area where honeypots can be of value. Honeypots can be leveraged within objectives B.2, B.4 and B.5. Due to their nature, they can aid in the detection of unauthorised access (Fabro et al., 2016), as people within the network could potentially try to access the honeypot. Within the system security, objective honeypots can be used to detect and remove malware, identify attacks that exploit vulnerabilities. A segregated network of honeypots can perform these functions without increasing, and can even be leveraged to reduce, the risk to the critical systems. When a honeypot gets targeted, it can also provide information that allows the organisation to respond to changes in risk.

Therefore, we can say that although honeypots, unlike other security systems, are not capable of protecting against attacks by themselves but require an analysis of their data which can then be used within security systems. Once you know who is attacking your network, and how they are doing it, you can more effectively defend against these threats.

Detecting cyber security events

As explored at the beginning of this Chapter, one of the primary purposes of a honeypot is to lure adversaries to them. Because of this, we can reasonably argue this objective is exceptionally suitable for honeypots. Honeypots inside the network can aid the organisation with the monitoring of the network, as a well-deployed honeypot should receive both adversaries inside the network as well as automatically mitigating malware. Setting up a separate honeypot network with the purpose of capturing threats to the organisation is a clear example of proactive security.

This objective encompasses the main strength of honeypots, their capability to detect security events. Generally, the more data points available, the more comprehensive the data captured is. Detection should therefore not only lie within the operational network, as this means the adversary is already inside the network, but should also include proactive

approaches. Honeypots are one of the most comprehensive forms of proactive event discovery, as they can be made to replicate many different systems and entire networks, which allows an adversary to behave like they were inside a real network. This information can then flow back into the previous subsection and help to inform possible actions that could be taken to improve the protection of the network.

Minimising the impact of cybersecurity incidents

After an attack, it is vital to learn how the attacker got into the system so any exploits can be patched. Although honeypots cannot aid with the response and recovery of systems that have been infected, they can provide a wealth of information relating to the incident. This can include further details on the adversary or even binaries used by the adversary during the attack. Having access to this binary can be of utmost importance. Attacks that happen within the honeypot network can also improve the operational network. Therefore it is essential to learn from attacks happening within this network as well.

A well-deployed honeypot, or network of honeypots, can also be used within digital forensics and incident response training. This further expands the capabilities of honeypots into a training environment and shows their flexibility. Additionally, once an organisation has been compromised, data collected by honeypots can be handed over to appropriate bodies such as law enforcement for further investigation. This means that even a small organisation that does not have the resources or expertise to investigate the logs captured actively can benefit from the deployment of honeypots. Therefore, honeypots are undoubtedly be part of the D2 objective.

3.5 Summary

This Chapter has presented a comprehensive survey of literature related to the deployment of ICS honeypots. Our survey began with an overview of industrial control system security and honeypots, afterwards we explored past attacks targeting ICS environments and the standards and guidance that are important within these environments. This all fed into a discussion on how honeypots can aid in adhering to these standards and guidance which gave the background for the survey of ICS honeypot deployments.

We analysed three ICS attacks, Stuxnet, BlackEnergy and Wolf Creek in the previous Chapter, which gave an overview of the attacks that these systems are facing. These attacks are only a small subset of ICS attacks, and many more have been and are being conducted. It is clear from these three attacks that the security of these systems is of great importance to our safety. Current ICS security lacks the ability to respond to new forms of malware quickly,

and the implementation of new patches is slower than necessary to mitigate vulnerabilities rapidly. Due to the nature of these systems implementing patches requires a vetting process to make sure the system will not be affected, and time slot has to be selected in which the system can be safely shut down and updated.

Due to the heavy regulations within the several sectors in which these devices are used, new forms of ICS security have to stem from legislation and guidance. For this reason, we explored several pieces of legislation and guidance from governments and international bodies. These included, and focused, on the UK Cyber Assessment Framework and the US NIST Framework for Improving Critical Infrastructure Cybersecurity. After introducing these, we explored them further in a honeypot context. This showed that honeypots could actively support several parts of many of these regulations, based on provided frameworks, and guidance.

Chapter 4

ICS Honeypot Implementations in the Research Literature

Now that we have covered how honeypots fit within ICS standards and guidance and they can be used within several objectives of the CAF and other guidance, we take a closer look at studies that have been done into ICS honeypots. We aim to look into the real data these studies have gathered and drawbacks of their approach. Several studies have been conducted into honeypot platforms which show that some perform better than others. The main difference between honeypot implementations is which data they can capture, for low-interaction honeypots that are linked to the amount of interaction available to the attacker and the quality of the emulation. When honeypots are poorly implemented, they can easily be identified by more experienced attackers and will therefore not be able to capture data from high-profile attacks or provide useless data (Krawetz, 2004).

Within this Chapter, we give an overview of low-, medium-interaction and high-interaction honeypots. The first subsection has an extensive overview of Conpot and other ICS honeypot implementations. During our research into these implementations, we could clearly see that there is significantly less research into ICS specific high-interaction honeypots than into their low-interaction counterparts. Based on our observations we also propsose a more comprehensive honeypot classification.

4.1 Methodology

For the purpose of this review, we have identified a range of ICS honeypots implementations that have been published over the years. Identification has been done by constructing a search query and using it on the following academic databases: IEEE Xplore, ACM Digital Library

and Google Scholar. Following search term was used to identify implementations of ICS honeypots in the literature:

("industrial control systems" or "ics" or "scada") and "honeypot" and "implementation"

The result set consisted of 302 papers; these have then been further distilled by eliminating papers that do not implement the discussed honeypot, are not discussing honeypots in-depth, or are multiple entries of the same paper. This resulted in 60 possible papers. Form these, several papers were not fully accessible, or did not evaluate the honeypot implementation. These steps resulted in 31 papers that were accessible, and implemented and evaluated an ICS honeypot.

4.2 Honeypot Implementations

4.2.1 Low-Interaction ICS Honeypots

The following 29 honeypot implementations are examples of low- and medium-interaction honeypots we have identified within our result subset. Several implementations use Conpot as a foundation, and others are developed for specific research purposes.

Jicha et al. (2016) performed an analysis of the effectiveness of Conpot by installing it in several separate AWS zones. Within the experiment, all ports were accessible, which would not simulate a real-world environment. We would always suggest configuring honeypots in a similar manner to a production device in order to gain the most accurate information. Overall, 12 Conpot honeypots were deployed, six Siemens S7-200 ICS and six Guardian AST gas pump monitoring system, over five different AWS locations. The authors noted that many more ports, such as 25 and 514, were found to be open through a Nmap scan, which resulted in them to believe Conpot is susceptible to Ubuntu default services. This again leads to an easy method for attackers to detect they are probing a honeypot instead of a real production system.

Another experiment involving Conpot by Kuman et al. (2017) which combined Conpot with OSSEC (OSSEC, n.d.) (a host IDS) and IMUNES (a network emulator) to create a honeynet that simulates an ICS. The combination between a simulated ICS and a simulated network could provide us with the opportunity to have a high-interaction environment with lower risks and without the need for significant investment in infrastructure. During the setup, the authors used the default Conpot template to emulate a Siemens S7-200 PLC and a modified version of the template to mimic an S7-300 PLC which had a vulnerability that was easy to reproduce. The experiment had a duration of two weeks and the bulk of registered activity consisted of port scans. No attempts were made to exploit the S7-300

vulnerability, and the port scans did not result in attacks on the system. The authors identified one possibility of the lack of attacks as the limited time of the experiment, which we noted as well.

The Beijing University of Posts and Telecommunications designed and implemented a more interactive ICS honeypot (Zhao and Qin, 2018) by improving on Conpot in two ways, Human Machine Interface (HMI) and industrial protocols. A simulation is used to provide the honeypot with data and activity to help it disguise itself further. Over 43 days they collected over 13000 requests, though which they managed to extract 244 IP addresses. Although their focus laid only on the improvement of the S7comm protocol, the study has managed to gain a significant amount of requests. Further improvements could be made, such as the implementation of more protocols and broader deployment of the honeypot. Overall, we can see movement in the right direction, improving interaction and simulating a production environment to fool attackers.

A Conpot implementation in combination with an IDS and a SCADA implementation was developed and tested by Ponomarev and Atkison (Ponomarev and Atkison, 2016). Conpot was used to represent an ICS and pymodbus was leveraged to implement the SCADA part of the ICS. Classification algorithms were used to differentiate between attackers and engineers, which was done at two different client-server separation stages to differentiate between outside and inside traffic. Overall, the purpose of this implementation lays more in the possibility to classify traffic than the perfect replication of an ICS in a honeypot for attackers to attack. Therefore, we cannot scrutinise how well implemented the honeypot is, but we can see where the benefits of a honeypot can lie with regards to data analysing. The IDS achieved an accuracy of 94.3% with no false negatives and 5.7% false positives. This shows a positive trend to implement honeypots combined with machine learning to gather and analyse data in order to warn system engineers when there is a possible attack.

In an experiment to study the fingerprintability, ways a honeypot can be identified as a honeypot, of ICS honeypots (Zamiri-Gourabi et al., 2019), Zamiri-Gourabi, Qalaei and Azad demonstrate the impact of the flaws found in honeypots by scanning the Internet to detect GasPot (Wilhoit and Hilt, 2015) and Conpot honeypots. These flaws range from the network delay and performance to further modifications such as keyloggers that can be detected and bypassed by attackers. Further, the limited amount of ICS protocols simulated can flag attackers they are interacting with a honeypot. The primary detection mechanisms are default configurations, missing protocol features, unusual behaviour and the underlying platform. A query ran on Shodan and Censys for one of Conpot its default configurations (PLC name: Technodrome) resulted in 214 hits on Shodan and 185 on Censys. As a side-note, it is possible that many of those are for testing purposes and not actively in use as honeypots. Although it is

certainly believable that there are active honeypots amongst those results. Another signature of honeypots was identified as the last modified value of "Tue, 19 May 1993 09:00:00 GMT", which returned a combined result of 373 hits. Unusual behaviour is identified as an unnatural pattern; an example listed by the authors included a steady -10% change of a value every hour. Discrepancies in the underlying platform are identified as the support of multiple ICS protocols on a single host when those protocols belong to different devices. The authors state that during their research, they discovered that their method managed to detect GasPots that were configured correctly, which shows that emulated devices can be detected even when not running default configurations. Overall, the tool managed to detect 17 GasPots, and after manual verification, they detected no false positives. This experiment proofs that the configuration and emulation of low-interactive honeypots is key to their operation. When emulation is lacking, protocols are not implemented which can be spotted by attackers and not properly configuring honeypots (keeping default configuration) is a significant error as attackers also know many ICS specific honeypots.

A set of Conpot honeypots was deployed by Ferretti, Pogliani, and Zanero (Ferretti et al., 2019) to analyse the interest towards ICS devices on the Internet. The authors note that Conpot is in its default configuration very easy to fingerprint, which lowers their value. Therefore, they have expanded upon and reconfigured the Conpot implementation of ICS protocols to make it more realistic. The implementation was verified by exposing it to Shodan, Shodan did not flag it as a honeypot and gave all instances a Honeyscore below 0.2. Shodan uses Honeyscore to give an indication if a device is likely to be a honeypot and uses characteristics of known honeypots to determine this score, the value is a range between 0.0 and 1 (Shodan, n.d.a). All honeypots were deployed behind a remote endpoint and connected through it over a VPN via a destination NAT rule. This method was used to deploy 11 honeypots. A further 20 honeypots were deployed on the cloud (10 in the US and 10 in Asia). Over the four-month testing period, the honeypots received a total of nearly 5000 connections of which most targeted S7, Modbus or EtherNet/IP. They have captured 1469 distinct IP addresses that specifically targeted ICS protocols and identified 97 distinctive actors amongst them. 17.72% of the IP addresses belonged to Shodan, and a further 6.63% to Censys. Of all the connections made by unknown scanners 60% came from Blackhost, a US hosting service, and nearly 90% of unknown actors (excluding cloud and hosting services) originated in China. The results of this study reveal that the top 10 actors made 92% of all ICS connections, which show that a handful of actors are responsible for most ICS traffic. Generally, the honeypots received only a handful of different request types, ranging from two to four per protocol. This is generally the case with automated attacks. We can see that the authors put effort into limiting the ability to fingerprint their honeypots, which pays off

by gaining a fair amount of connections. The implementation proofs to be useful to gather information on automated attacks, but we feel that the lack of variance in request types show that non-automated attacks are rare.

An evaluation of Conpot is made by Dutta et al. (2020) in which they examine its behaviour towards scanning attacks. A default template of Conpot was deployed inside an organisational network running on an Ubuntu machine. Both Zenmap and Nmap were used to scan for open ports on the machine and gave the same results. Other scanning tools were used but did not manage to produce similar results. None of the scanners was able to list all open ports on the honeypot. Interacting with open ports, such as FTP port 2121, created accurate logs within the honeypot. The authors note that Conpot is not as advanced as a host-based intrusion detection system, but provides adequate results. However, it misses features such as notifying the network administrator when an intrusion occurs. A possible solution to this problem is listed as the implementation of OSSEC, which would complement the logging with extra data. This evaluation of Conpot is relatively limited, mainly because it does not include a real attack on the honeypot itself, but does show that the ICS services can be found by scanning the open ports. The introduction of an intrusion detection system on the Conpot machine itself is a well-thought addition and should, in our opinion, be included in any low-interaction honeypot.

Wang et al. (2019) designed a honeynet-based IDS to capture traffic and study it using machine learning. The system uses several Conpot instances to capture the traffic on the system and extends on Conpot by implementing an intrusion detection module, which uses an SVM trained model to categorise the traffic as malicious or benign. After training the model, it has been implemented into the architecture and verified for accuracy. The model achieved a peak accuracy after being trained with 4% of the data set of nearly 90% with 90s training time and 80s testing time. However, with 1% of the data set training time is reduced to 3s, testing time is at 23.5s and accuracy remains on a similar level of 89.39%. Although we cannot comment extensively on the design of the honeynet, we can say that in a real implementation it would be relatively weak. Conpot, in its default configuration, does not present a wealth of possible interactions or obfuscation necessary to behave like a real PLC. Nevertheless, this paper shows the usefulness of honeypots in a more automated environment. Feeding the honeypot data through a machine learning algorithm and automatically categorising it as malicious or benign will significantly aid system administrators in the security of their network. Although the accuracy of nearly 90% is not perfect, and there is no data given on the false-positives generated, there is still room for further development.

A Gridpot honeypot, which leverages Conpot, to analyse the threats on the smart grid is deployed by Kendrick and Rucker (2019). The honeypot uses GridPot as an open-source

honeypot framework, and it uses a honeypot and modelling layer to integrate between GridLAB-D and Conpot. GridLAB-D is a power-distribution simulator that uses algorithms to model and test these systems. The system implements the following protocols for the attackers to interact: HTTP, Modbus, S7COMM, SNMP and IEC 61850. The environment used to deploy the honeypot consisted of an Oracle VirtualBox running GridPot on a Dell XPS desktop with a Windows 10 operating system. A test environment was structured in a similar way but ran on an Ubuntu operating system rather than Windows. The model deployed was the IEEE_13_Node_With_Houses, which consists of 13 nodes and 15 houses. In this setup, a node is a node on the network and a house represents a single family home connected to the smart grid. Conpot was obfuscated to lower the fingerprintability and present as a more enticing target for attackers. The experiment lasted for 19 days with minimal interference, aside from a broken link between Conpot and the modelling layer. In total, more than 9 million packers were captured, of much, the majority consisted of network broadcasts, ARP and other standard traffic. After filtering this data, 1.5 million packets remained. More than 50% of these packets originated from one cloud hosting company located in California, 3.6% from an IP address registered in Russia. Overall, HTTP was the most targeted protocol with Modbus being a distant second. Thirty-nine unique addresses had multiple interactions with the honeypots, ranging from repeat actions to further probing. After analysis with Shodan, the honeypot had a HoneyScore of 1.0/1.0, meaning that according to Shodan there is a very strong possibility the address hosts a honeypot. From the HoneyScore, we can see that the implementation was subpar to other honeypot deployments and did not manage to deceive Internet scanners. As a result, we can safely say that traffic to the honeypot will be of much lower value. Well-trained attackers will generally refrain from attacking a system like this, and automated tools might engage with it depending on the script running. More effort has to be put in the obfuscation of the honeypot, which should result is higher quality data.

A honeypot proposed and evaluated by Buza et al. (2014), but we will focus on the more improved implementation of Holczer, Félegyházi and Buttyán (Holczer et al., 2015), is based on a Siemens ET/200S PLC which is emulated on a Ubuntu virtual machine. The PLC generally offers three primary services, STEP7, HTTP(S) and SNMP. Both the STEP7 and HTTP(S) services are emulated in one emulated service, and SNMP emulated as a separate service. They deployed the honeypot on a public network in an attempt to obtain real traffic to the honeypot. It is noted that due to the deployment of the honeypot on a university network, attacks explicitly targeting ICS would be rare. After an eight-day test, no traffic was observed on either STEP7 or SNMP ports. Attempts were made to access the SSH ports but were blocked by the firewall. No specific PLC attacks were detected. The one-month extended test produces similar results, except for a limited amount of traffic to the STEP7 port. A second

short test, again, provided no different information than the other tests. This implementation provides us with interesting information. The use of a university network has a negative impact when trying to entice attackers and make them think they attack a real organisation.

Serbanescu, Obermeier and Yu (Serbanescu et al., 2015) deployed a research ICS honeypot over the public Internet using the Amazon EC2 cloud. They used software emulations of ICS/SCADA devices. However, they admit the goal of the emulation was not to mimic the devices perfectly but rather obtain information on the overall threat landscape. We would argue that to gain an in-depth and accurate overview of the threats to ICS devices. One should portray their honeypots as closely as possible to the device they want to mimic. Especially within the ICS environment where knowledgeable attackers closely scrutinise the devices, they attack. The deployment within the Amazon cloud infrastructure can also result in less valuable data, as attackers should be aware that PLCs would not be deployed in those environments. We agree with their statement that the cloud provides benefits in terms of scalability, but efforts have to be made to obfuscate the use of the cloud. It could also be argued that the deployment of multiple devices within one network, instead of some spread over several Amazon EC2 regions, can aid to entice attackers as a real production system would have multiple systems such as PLCs as well. Eighteen honeypots were deployed over eight EC2 regions, with most regions having either two or three instances. Overall, 1092 Modbus connections and 1040 requests were made, around 22% of both originated from Shodan, and more than 70% of other attacks originated from one other server. All IEC-104 requests and 23 out of 34 port scans were conducted by Shodan as well. These results show a lack of real threats probing and attacking the honeypot deployments.

Jaromin (Jaromin, 2013) designed and implemented an industrial control emulator that acts as a decoy field device. The PLC emulator is implemented on a Gumstix single board computer running a Linux distribution and emulates a web service on port 80, Modbus on port 502 and a host automation products (HAP) protocols service on port 28784. Several iptables rules have been set up to filter and queue the packets. Due to the focus on higher-level protocols, only web and HAP protocols were evaluated for an accurate representation of the target PLC. In general, we see that efforts have been made to emulate a PLC accurately, but there are multiple shortcomings to make us confident that the honeypot is a detailed representation of a PLC. The results showed a high level of packet-level accuracy for the implemented services; the honeypot also performed well in the scenarios conducted to evaluate the accuracy at both scanning and attack levels. An area where the emulator fails to perform similar to the PLC is response times, which were more than 98 times slower for some workloads on the Gumstix compared to the PLC. In standard honeypot scenarios, the

slower response of the Gumstix was negligible. Based on this research, we can identify a possible well-performing honeypot if further development is conducted.

To investigate the extent of malware and attacks targeting critical infrastructure, Berman and Butts (Berman and Butts, 2012) presented a honeypot based on Gumstix technology which emulated an ICS field device. The Gumstix was programmed to support Modbus communication following RFC specifications to incorporate standard function codes. When the honeypot receives a message, it will respond in an appropriate manner, e.g. valve closed. Unrecognised function codes are responded to by an unrecognised function code error code. All non-Modbus traffic (any port aside from 502) is logged but not responded to. Nmap scans returned the expected results but failed to recognise the honeypot was running on a Linux OS. This approach has its merits and provides a low-cost and low-maintenance PLC honeypot, but may not be effective against attackers with ICS knowledge, which the authors also acknowledge. It could be useful to capture scanning attacks and malware that automatically propagates throughout the network aiming to infect PLCs. To accurately implement all the necessary function codes to respond to Modbus communication, the configuration would require a significant amount of time. It has to be evaluated in the time spend to implement the honeypot is worthwhile compared to the information it would capture.

You et al. (2019) explored the use of honeypot data to characterise Internet-wide automated ICS attacks. To this regard, they have implemented a minimal interaction honeypot, MirrorPot. The honeypot generates a response based on the request, but the honeypot never parses the incoming requests. Their goal is to capture the actions of automated scripts, not specific attacks executed on the devices themselves. They deployed seven instances over the world hosted on static ISP addresses with 26 ports running ICS services. The most extended deployment lasted for 477 days, the shortest for ten days. Five out of seven honeypots were active for less than 40 days, which we deem insufficient to gain usable long-term threat intelligence. For example, it can take Shodan several days to fully index devices, with the initial scan potentially taking days (Bodenheim et al., 2014). At the end of the experiment, 2.6% of all requests (56,643,490) contained payloads, and only 5.3% of attacks were targeted at ICS-related ports. Less than 20% of IP addresses were spotted more than twice, which shows that the number of targeted attacks was low. Although the experiment resulted in some impressive results, such as more than 20 common ICS-related attack patterns, we still believe that the experiment overall lacks on the implementation side. Many ICS attacks are targeted, and attackers with knowledge of ICS easily spot weak emulations. Due to the short lifespan of most honeypots (less than 40 days), we feel that there was not enough time for those honeypots to be appropriately indexed and scanned. There is also a discrepancy in the data, showing 477 days as the most prolonged duration, but the authors claim a 418 days

duration. Further, they claim they ran seven honeypots for 418 days when the majority of those ran for less than two months.

HosTaGe (Vasilomanolakis et al., 2016) is a honeypot developed by Vasilomanolakis et al. and is categorised as a lightweight low-interaction honeypot for mobile devices aimed at detecting malicious devices. They have adapted HosTaGe to support ICS networks by supporting the emulation of ICS protocols (Modbus, S7, HTTP, SNMP Telnet, 5MB and SMTP). The implementation of these protocols looks extensive, but we still doubt that the implementation would have been perfect and therefore, attackers could notice missing features, vulnerabilities and others. HosTaGe has a built-in detection mechanism to detect attacks directed at a single protocol, multiple protocols (based on the same source) and based on the payload sent to the device. It captures the packets and connection requests it receives and generated signatures when it detects an intrusion. Those signatures can be sent to a Bro IDS. In comparison with Conpot, HosTaGe generally received more traffic over the same period on all protocols except Modbus and was also able to detect unique malicious IP addresses. We would like to point out that the experiment was set up without any firewalls between the honeypots and the Internet, which would be a significant red flag for experienced attackers. However, the goal of the experiment was to identify automated attacks towards ICS devices and therefore, it might not have made a significant difference. The authors note that Shodan was able to scan the honeypots during their tests but only managed to identify Conpot as a honeypot. This approach presents us with an interesting aspect, the generation of signatures and evaluation of malicious data on the honeypot, which is subsequently sent to a connected IDS. The purpose of honeypots is to capture data. Because of this, they are only as useful as their data and how the data is used. When data can be evaluated in real-time, it is at is the most-valuable point, as administrators are then able to mitigate an attack while it is happening.

In an attempt to discover which actors are scanning the Smart Grid, Mashima, Li and Chen (Mashima et al., 2019) deployed five low-interaction honeypots on geographically different AWS instances. Their emulation was relatively limited and included TCP listeners and simple server programs for IEC 60870-5-104 and IEC 61850 emulation. It was verified that Shodan did not categorise the systems as honeypots. This shows a lack of vetting on Shodan its part, as the limited amount of interaction and services in combination with the deployment on an AWS IP address should be a significant red flag. The honeypots were active for over six months and captured 6 GB of ICS network traces. Weekly packet count was in the same range with the exception of some spikes due to some of the honeypots receiving a significant amount of traffic for a short period. Multiple attacks showed an identical approach, which leads to the conclusion that a similar tool must be used. In our opinion, this is also an

indicator of automated attacks with limited intelligence behind it. The honeypots received a denial-of-service attack, and DNP3 and Modbus TCP scans. An interesting result of this experiment is that there is little correlation between the daily traffic intensity for the different geographical locations. This could be an indicator that scanning tools scan the Internet in geographical segments, but this brings us little more information unless further analysis is conducted. Overall we think this implementation is lacking effectiveness and the data analysis provides little results to be used to increase ICS security.

TrendMicro (Wilhoit and Hilt, 2015) implemented a GasPot honeypot to evaluate the attractiveness of SCADA honeypots for attackers. GasPot is designed to not look like a honeypot. Every deployed instance is unique, which makes it harder to fingerprint. It supports six commands and allows users to change values in the simulation. They deployed GasPot instances on physical IP addresses (no cloud services) in seven countries: US, Brazil, UK, Jordan, Germany, UAE and Russia. Some deployments were designed to be detected by Shodan to collect data on automatic attacks. After deployment, they discovered a Pastebin post where people shared information about the vulnerable instances deployed in this research. This is very interesting and shows that there is a real community amongst hackers. Attackers managed to exploit the honeypots, change their names and perform other actions. Looking at the source of the attacks, they were spread all over the world. The implementation shows a lot of promise as useful data was captured and presents possible avenues such as purposely leaking information about a honeypot within hacker communities to gain more data. All the honeypots were configured to resemble a real device, and although these were emulated, the attackers did thoroughly engage.

In an effort to evaluate the efficiency of honeypots in the detection and evaluation of advanced threats, Wade (Wade, 2011) implemented a test system based on the Digital Bond framework. The honeypot is deployed on the university network, behind a Honeywall, and a Nmap scan correctly identified the services running on the open ports (21/tcp, 80/tcp and 502/tcp). The services running on the honeypot include HTTP, FTP, Telnet, SNMP, VxWorks Debugger and Modbus. VxWorks Debugger is a real-time operating system for embedded systems and designed to run on top of another operating system. HTTP, FTP, SNMP and Telnet services are partially implemented to lower the vulnerability of the system. On the network, Snort is deployed for intrusion detection. The honeypot was running for 38 days, with seven days non-consecutive of malfunctioning resulting in 31 days of data. Nearly 2 million Snort alerts were generated over the deployment period, the majority of the alerts (97%) were generated by UPnP malformed advertisements which only affects older Windows systems. Of all other alerts, the majority were UDP port scans and port sweeps (2.8% of total alerts). The experiment concluded that there was no specific interest in the SCADA

services, and most attackers were not aware of what operating system was running. The first area we would look in an aim to discover there was no ICS specific interest would be the deployment of the honeypots on a university network, which are well known for hosting research honeypots and not known for hosting ICS devices. Further, the emulation was adequate and should have presented itself in a similar way than other low-interaction ICS honeypots.

4.2.2 Medium-Interaction ICS Honeypots

JPCERT, the Japan Computer Emergency Response Team, has implemented Conpot in a honeypot called THS to trace attacks and prevent further infections within the network (Abe et al., 2018). Their design uses Honeyd, to receive and classify packets, and perform actions based on the request it receives. ICS requests are forwarded from Honeyd to Conpot to provide attackers with interaction. The level of interaction, therefore, is limited to what Conpot delivers and does not allow for full PLC interaction. One of their evaluation cases involved the introduction of a computer infected with Havex RAT malware. The malware successfully identified the honeypot, due to the imitative ICS data sent from Conpot. Outgoing scans to the infected PC provided information related to the OS and the open port. THS is able to detect scans directed to the honeypot itself, but would not be able to detect attacks that affect other devices or do not send scan packets. When malware has affected another PLC, which then sends commands to other PLCs, the malicious activity would be revealed by THS by observing commands send within the network. If attackers were to write malicious software to the honeypot, THS would detect the malicious payload and logs created can provide further information to identify the activity. Even actions to overwrite the payload to erase any evidence, would be logged by the honeypot. We view this implementation as promising, and it shows the opportunities honeypots can provide to defend the network. When an infected device enters the network, the honeypot can detect the abnormal activity sent to it and can provide useful information related to malware used to target PLCs. Although, we still have to acknowledge that attackers with a background in ICS would be able to spot the honeypot and likely refrain from interacting with it. Therefore, targeted attacks by experts could still succeed without the honeypot capturing any data related to it.

Pliatsios et al. (2019) implemented an interactive ICS honeypot based on Conpot that has the ability to replicate traffic as if it was a real device. Their setup consisted of a real and virtual HMI, a Saitel RTU and Conpot to replicate a Saitel RTU. Both Conpot and the virtual HMI run on a VM. To increase the level of emulation, Conpot uses the traffic data from the real RTU (by feeding pcap files from the real RTU), and the virtual HMI generates requests for the Conpot honeypot. The purpose of the real HMI is to allow an operator to monitor the

status of the real RTU. Assessment was done by implementing the honeypot in a real-world hydropower plant. The honeypot managed to emulate the behaviour of the RTU, and the virtual HMI successfully generated realistic traffic. Although the assessment did not include any interaction with the honeypot from an attacker perspective, there were some interesting approaches. Utilising real data to feed into a Conpot honeypot and emulating traffic based on actual ICS traffic undoubtedly makes the honeypot more realistic. However, one has to esure the data presented does not contain sensitive information or will not aid the attacker in their activities to, for example, obtain process comprehension. The emulation itself would also still have flaws, such as not supporting all ICS functions and not allowing for high-interaction; it is a step in the right direction. We could definitely see a similar approach for a honeypot that is situated in a production environment, which possibly could capture malicious data sent to all devices inside the network. It is questionable if an attacker would be deluded that they are interacting with a real device.

Mimepot (Bernieri et al., 2019) is, according to its developers, a cyber-physical honeypot that is able to simulate physical processes and leverages SDN to provide a future-proof approach. It consists of two modules, Mime Plant and Mime E&C. MimePlant is the network node which simulates the PLC, whereas Mime E&C simulates a SCADA workstation which regulates the plant behaviour. Both modules have to be implemented separately, so they are able to produce network traffic similar to a production network. SDN is used to redirect malicious traffic to Mimepot, and to obfuscate the IP addresses of the real devices to fool the attacker into thinking they are attacking the real device. Their evaluation consisted of attacking Mimepot and changing the configuration. Attacks are successfully redirected to the honeypot, and attackers are able to see changes based on their requests. The introduction of SDN shows an excellent example of how other techniques can benefit honeypots, the redirection and obfuscation could fool an attacker to a certain level and protect the infrastructure. We believe that the general evaluation is lacking as the attacks were not carried out by attackers without knowledge of the system. Therefore they do not correctly replicate a real attack. The authors know what functions are supported by the honeypot and would have shown a different approach. More research has to be done into the robustness of Mimepot, as it is not clear how in-depth the implementation of the ICS protocols is.

In an attempt to develop a new medium- to high-interaction PLC honeypot Lau et al. (2016) developed XPOT to simulate a Siemens S7 314C-2. They achieved high interactivity by supporting more than 100 MC7 different instructions and allow an adversary to load PLC programs on the honeypot. The authors note that due to the fact that XPOT has to compile the bytecode, adversaries can easily spot the delay in execution compared to a genuine Siemens PLC. An evaluation of the honeypot was done by allowing six participants

of a PLC programming and hacking course to distinguish a real PLC from XPOT. Although participants were not always sure what the correct behaviour of the PLC should be, all participants correctly identified the XPOT when given access to all tools and features. We can be reasonably sure that experienced attackers would be able to identify XPOT as a honeypot without many issues. This, again, shows the need for a near-perfect implementation and the benefits of using a real PLC as a honeypot to trick attackers into believing they managed to penetrate a real device.

Simões et al. (2015) propose an emulated PLC honeypot that is situated between operational PLCs (physically or logically). It aims to divert attackers to attack it and actively report on the suspicious activity targeting it to the IDS. The honeypot focuses on Modbus and runs both simulated and complete services that are generally found on a PLC. Central to this honeypot is the Modbus API simulator, which provides the necessary Modbus functionality. Other modules included are FTPD, SNMP and a module to detect probing activities (port scans) on the other TCP/IP ports. The location of the honeypot is undoubtedly beneficial in our opinion as they would be in an environment that has real PLCs and the honeypot should be in a prime position to capture malware that is propagating throughout the network. Due to the programmability and configurability of Modbus API, they are able to mimic a wide range of real PLCs and provide unique behaviour. There is also an event monitor integrated within the honeypot to analyse the data captured. A module for remote management is also included to allow security staff to monitor the honeypot and allow for remote actions. To prevent the attacker from using the honeypot as an attack vector, a firewall is implemented to deny connections from the honeypot to the other systems but allow all incoming connections. We can see this is a comprehensive emulation and provides a lot of useful features. The location of the honeypot is one of its key strengths, especially with the lowered risk to other systems due to the firewall. Because there is no evaluation, we cannot comment on the attractiveness of the honeypot and if attackers would be able to detect it.

IMUNES is a well-known network simulator and was used by Haney and Papa (2014) to simulate a SCADA honeynet. The honeynet framework consists of two areas, the honeynet and the organisational network, and the former includes a honeywall and the latter a standard firewall. Both areas are therefore strictly separated. Within the honeynet, there is a system dedicated for log collection and an IMUNES on FreeBSD Cluster running virtual PLC/RTU nodes. Each of those nodes runs a JAMOD PLC simulator, honeyd and Sebek for data collection. JAMOD is a Java Modbus library that supports TCP, UDP and serial connections and is used to emulate the PLC functions. To further enhance the emulation honeyd is leveraged to simulate other operating systems and services such as a login shell, and a management interface over HTTP(s). Snort is deployed to monitor and capture traffic

within the honeynet, and generate alerts when signatures match a known attack. Sebek is deployed within the honeypots to capture keystrokes and system interactions, to further analyse how attackers are exploiting the system. This proposed framework presents us with some interesting approaches. First, using IMUNES to simulate the honeypots and second to emulate a PLC through Java. Using IMUNES adds an extra level of virtualisation that, in our opinion, can increase the fingerprintability of the honeypot and we would encourage anyone implementing honeypots to refrain from adding layers of virtualisation when it is not necessary. Especially when working within ICS environments. Although Java is an excellent programming language, in this case, the authors are doing something difficult when there is a tested alternative available. Honeypots like Conpot are designed to emulate ICS devices and used in a wide range of implementations already. Further experiments have to be done to verify the capabilities if the honeynet performs well, it would open possibilities for large honeynets with a small footprint.

Antonioli et al. (2016) have proposed a realistic virtual ICS honeypot that, according to them, allows for high-interaction by the attacker. We can see that the honeypot mimics a real ICS architecture near perfectly, with a simulated network environment, PLC, HMI and processes. Within the honeypot, they allow the attacker to fingerprint the device (e.g. Nmap, xprobe2) and obtain necessary information, including IP addresses and ports, and protocols used. There is no prevention for the attacker to obtain escalated privileges on the system, but the authors have limited their attacker model only to include reasonable scenarios. The implemented honeypot is able to capture Man-in-the-Middle attacks, port scanning and DoS attacks amongst others. Within this honeypot, we can see the benefits of emulating a real environment, as attackers have more ways to interact and engage. Overall, we can see this is a comprehensive implementation of a honeypot that should be able to fool even higher level threat actors. However, some decisions such as the use of weak SSH credentials and plaintext telnet authentication might be a red flag. Nevertheless, we would not call this an actual high-interaction honeypot, based on its virtual nature, but it does result in comprehensive attack data. We would like to see a similar honeypot based on real devices to achieve true high-interaction, with a possibility to emulate the processes and network.

A honeynet that is capable of emulating an entire smart grid field communication infrastructure was designed by Mashima et al. (2017). Substations are simulated on a virtual machine connected to a substation LAN, and OpenMUC is used to implement the communication interface. The substation network is emulated through Mininet, which runs on its own virtual machine. Mininet virtual hosts are camouflaged by making them look like IED devices, for this all packets are forwarded to a SoftGrid virtual machine. Physical components of the power grid are simulated on a PowerWorld simulator. Traffic latency is

close to a real implementation, and frequency changes are handled dynamically to emulate realistic changes to attackers. The honeynet is deployed on an Amazon IP address and utilises a ring topology as it is most frequently used. Fingerprintability was tested through means of Nmap to detect the operating system, and deployment of multiple virtual machines on the same hosts to evaluate the effects a high-load on one machine has on the others. The authors note that running multiple virtual machines on the same hardware can lead to the discovery of the honeypot and potential mitigation for this is to host virtual machines with their own dedicated processor core. It does not seem that the implementation was utilised for data capturing purposes. We can clearly see the authors aimed to create a honeypot that is comprehensive and difficult to detect. Implementing a honeynet, instead of a single honeypot, dramatically increases the interaction and emulation, which in turn should be more enticing for attackers. A statement was made that the deployment on an Amazon IP address does not increase the detectability of the honeypot, we disagree with this statement as real ICS infrastructure would not be deployed in the cloud. The use of AWS may result in data from automated attacks, but these are generally of lower complexity than targeted attacks. Our advice is to deploy ICS honeypots in an IP range that would generally see ICS deployments.

iHoney (Navarro et al., 2018) is an ICS honeypot designed by Navarro, Balbastre and Beyer to mimic a real ICS infrastructure as close as possible. The authors decided to simulate a water treatment plant, and therefore the design of the infrastructure followed the same process as it would have done in an actual plant. iHoney consists of three modules, the ICS system, simulation system and monitoring infrastructure. The ICS system consists of a SCADA server, PLC control network, and the associated ICS protocols. The plant is simulated within the simulation system to generate realistic outcomes in real-time and allows for interaction as a real plant operator would have. Monitoring is done by a Network Intrusion Detection System (NIDS) and Host-based Intrusion Detection System (HIDS) on the network and exposed SCADA server respectively. To further lower the fingerprintability, the HIDS is hidden behind a legitimate program which checks the communication status of the PLCs. The authors note that several concessions had to be made; for example, they had to be a balance between the complexity and realistic simulation of the system. This shows one of the main drawbacks of low-interaction honeypots, but this emulation looks well-thought and well-implemented. The honeypot was exposed on the Internet for over two years and was attacked on a daily basis. Most of the attacks were automated, which we would expect on a low-interaction honeypot, and some attacks involved social engineering, which is a sign of advanced attackers. Between July 2015 and September 2016, the initial exposure period, monthly IDS alerts mostly ranged between 1,000 and 10,000. However, August, September and October peaked at more than 600,000 alerts monthly (with August generating

over 1.3 million), the authors identified a problem with the remote desktop protocol used which was substituted with a Virtual Network Computing service afterwards. We can see a considerable amount of data gathered by iHoney, which looks like a well-designed honeypot. The capturing of social engineering attacks show that attackers were fooled by the honeypot and dedicated time to infiltrate it, which proofs the detailed design and implementation was beneficial.

The Symbolic Cyber-Physical Honeynet or SCyPH was presented by Redwood, Lawrence and Burmester (Redwood et al., 2015). It is designed to entice attackers, enhance the screening and coalescence of attack events and more. They emulate cyber-physical systems (CPS), implement a SCADA HMI, and provide logging and anomaly detection. The primary target is the HMI, which is integrated with the CPS and uses a web interface to allow interaction. The whole framework is modular and allows for the implementation of other systems. All actions taken on the HMI are represented on the CPS so that the attacker can see the result. Gridpot was used in conjunction with the framework to demonstrate its capabilities. MMS, GOOSE and Modbus protocols in IEC 61850 were emulated as services for the attacker to interact with. An attack was launched with specifically written malware, which successfully changed the state of an intelligent electronic device switch in one of the simulated substations. One interesting action taken by the authors was the exploitation of a vulnerability within the HMI software, which shows the importance of correctly emulating software within honeypots as attackers would have knowledge of such exploits and actively try to use them. All actions and binaries used within the system are logged and available for a forensic replay of attacks. This framework is promising and adaptive due to its modular nature. For proper emulation of devices, a significant amount of work has to be done, and non-discovered exploits would not be able to be implemented. This might allow attackers to fingerprint the honeypot. In general, we view this framework as promising but note the hard work needed for proper implementation.

In an aim to create resilient cyber-physical systems (CPS) Bou-Harb et al. (2017) propose an approach that unites both cyber and physical environments. The main aim of the study is to leverage real threat intelligence into the security of CPS, and the proposed architecture consists of both a cyber and physical layer. Within the cyber layer, dynamic malware analysis as an active measure and a Conpot honeypot as a passive measure. The physical layer is similar to a generic CPS environment. However, it is extended upon by implementing a CPS monitor to tap, gather and amalgamate data flows and coordinate with the threat detector to react to an attack. A cyber-physical threat detector is implemented and receives data from both physical and cyber layers to monitor all data and detect attacks to the systems. We will focus on the honeypot implementation of this framework. To enhance Conpot and provide

a realistic CPS, the authors have implemented further emulation to include CPS protocols. After a one month deployment, Conpot managed to capture about 500 unique attackers which generated thousands of events. Further analysis showed that 10 000 packets contained TCP and UDP scanning attempts and 2000 were TCP DoS attacks on CPS protocols. Two of the three case studies done within the environment included the honeypot as an attack vector. The first one consisted of an attacker attempting a privilege escalation attack on the honeypot by exploiting the HMI session manager. Successful mitigation of the attack was done by blocking traffic originating from the IP address. In the second case study, the attacker exploited the SNMP to gain an overview of all operational services. When this was successful, the attacker generated malicious Modbus request to cause damage to the system. The countermeasure to this attack was dropping the malicious requests and blocking traffic from the IP address that exploited the system. We can clearly see that the honeypot managed to capture a significant amount of data, which could contain valuable threat intelligence, during its short deployment. Further, we can see that attackers trying to exploit the honeypot were successfully caught, and the threat was mitigated. This is a perfect example of how honeypots can be of extreme value when combined with other systems. Although the honeypot implementation was fairly basic, the combination with the other aspects of the system would make it an enticing target.

4.2.3 High-Interaction ICS Honeypots

The following three honeypot implementations are high-interaction honeypots we have identified within our result subset. High-interaction ICS honeypot implementations are clearly less common than low-interaction variants, which can be explained by the high costs of equipment, and the level of knowledge required to deploy and maintain them.

In a recent research paper by Trend Micro (Hilt et al., 2020), a fake factory consisting of honeypots was set up to attract and capture real threats. Within the company, there were cellular routers, protocol gateways, servers and HMI (virtual), and physical industrial control systems. Four PLCs were implemented, one Siemens S7-1200, two Allen-Bradley MicroLogix 1100, and one GE Fanuc. They also included a Phoenix ContactILC 131 inline controller. One comment we have to this approach is that it is unlikely for companies to implement many different brands of PLCs which, for us, would be a clear indicator that the network might consist of honeypots. Nonetheless, the honeypots managed to gain attention and capture comprehensive attacks which resulted in system shutdowns, fraud and more. We can see quite a bit of non-ICS related activity, such a fraud, crypto miners and ransomware. ICS specific attacks are generally attackers playing with the HMI and the factory infrastructure. This might be linked back to our previous point. The implementation

does show that even when we set up a comprehensive ICS honeypot, we have to take into account other actors and their malicious intents.

Aside from an emulated PLC honeypot, Simões et al. (2015) proposed a high-interaction honeypot architecture where the attacker interacts with a real PLC. However, it is not linked with any industrial processes. All traffic to the PLC is forwarded to an IDS, which is more accessible due to the unencrypted nature of the Modbus protocol and generates security events. The main advantage the authors' list is the implementation of the real infrastructure for the attackers to interact. The cost and complexity of the implementation, especially when there are multiple honeypot deployments, is listed as one of the main disadvantages. Real PLC honeypots make it harder for attackers to spot the honeypot if they never interact with the monitoring systems themselves. Attackers can interact with the PLC in the same ways they would when they attack real production infrastructure, which lowers the suspicion they might be targeting a honeypot. The architecture is proposed in limited form and not evaluated.

An operational technology honeypot designed and implemented by Piggin and Buffey (2016) consists of four major components: control systems and process simulation, situational awareness and forensics platform, the attacker's infrastructure and the remote monitoring infrastructure. They have designed a high-interactive honeypot that allows the attacker for detailed interaction with the honeypot and specifically designed it for forensic investigations. The honeypot was designed to attract attackers and capture valuable, for which they developed an application to resemble real automated processes. After deployment, the honeypot managed to capture ICS attacks related to the disruption of PLC data communications, an anonymous attack against the PLC originating from the TOR network and password attacks using default vendor credentials to delete directories on the SCADA PC, amongst others. This implementation shows the need to lure attackers to honeypot implementations that closely resemble production systems.

4.2.4 Summary of Discussed Implementations

We can identify the essential characteristics of ICS honeypots that are important to adhere to in order to capture valuable data. An overview can be found in Table 4.1. This is a high-level overview of some of the protocols implemented within the discussed honeypots, this list is by no means exhaustive and could contain errors. Some implementations do not specifically discuss all implemented protocols, and these have been deducted from background knowledge. To aid in the comprehension of data we provide some information on how the data is presented in the table:

- ✓: one or all of the protocols listed in the column are implemented

- Data Captured relates to the quality of data captured related to the attacks on the system
  - **Limited** e.g. No specific ICS data
  - **Basic** e.g. Only initial reconnaissance on the ICS honeypot
  - **Intermediate** e.g. A significant amount of data that can aid in the investigation of attack vectors
  - **Comprehensive** e.g. Extensive ICS interaction on the honeypots and potentially around the honeypot

Looking into the low-interaction honeypots, we can see that the data gathered by them is of relatively low value. The data captured by these honeypots are generally limited to Internet-wide scans or initial reconnaissance. When low-interaction honeypots are connected to the Internet, they can be used for high-level threat intelligence purposes and to gain information on how ICS devices are scanned on the Internet, which can enable organisations to limit the exposure of the organisation to these scanners. Within the network, they could be able to spot the automatic propagation of malware that has already managed to infect a device on the network. However, when deploying low-interaction honeypots for any purpose, it is important to obfuscate the default signatures that may be included in the platform used. For Conpot, the deployment that is most popular, there are multiple signatures we have evaluated in previous work, which range from signatures on the HTTP emulation to information seen on the S7Comm protocol (Maesschalck et al., 2021). Due to the limited information that is generally available within the papers we have evaluated we cannot provide an in-depth overview of the fingerprintability of the platform itself. This is why, within this survey, we have mainly focused on the environment the honeypots were deployed in.

Author(s)	Interaction	HTTP(S)	Telnet/SSH	(T)FTP	SNMP	Modbus	IEC-104	IEC 61850	S7Comm	Data Captured
Jicha et al. (2016)	Low	✓	✓	✓	✓	✓			✓	Basic
Kuman et al. (2017)	Medium	✓		✓	✓		✓		✓	Basic
Zhao and Qin (2018)	Low	✓		✓	✓	✓			✓	Basic
Ponomarev and Atkison (2016)	Low	✓		✓	✓	✓			✓	Basic
Abe et al. (2018)	Medium	✓		✓	✓	✓			✓	Basic
Pliatsios et al. (2019)	Medium	✓		✓	✓	✓	✓		✓	No Eval.
Ferretti et al. (2019)	Low	✓	✓	✓	✓	✓	✓		✓	Basic
Dutta et al. (2020)	Low	✓		✓	✓	✓			✓	-
Wang et al. (2019)	Low				✓	✓			✓	No Eval.
Kendrick and Rucker (2019)	Low	✓			✓	✓	✓		✓	Basic
Bou-Harb et al. (2017)	Medium	✓			✓	✓			✓	Intermediate
Holczer et al. (2015)	Low	✓			✓				✓	Limited
Serbanescu et al. (2015)	Low	✓	✓	✓	✓	✓	✓			Basic
Jaromin (2013)	Low	✓	✓	✓						No Eval.
Redwood et al. (2015)	Medium					✓		✓		Intermediate
Berman and Butts (2012)	Low					✓				-
You et al. (2019)	Low		✓		✓	✓				Basic
Bernieri et al. (2019)	Medium					✓				Intermediate
Vasilomanolakis et al. (2015)	Low	✓	✓	✓		✓				Basic
Navarro et al. (2018)	Medium					✓			✓	Intermediate
Mashima et al. (2019)	Low					✓	✓	✓		Basic
Mashima et al. (2017)	Medium		✓				✓	✓		No Eval.
Haney and Papa (2014)	Medium	✓	✓			✓				No Eval.
Wilhoit and Hilt (2015)	Low									Advanced
Wade (2011)	Low	✓	✓	✓	✓	✓				Limited
Antonioli et al. (2016)	Medium	✓	✓		✓	✓				Comprehensive
Hilt et al. (2020)	High	✓	✓	✓					✓	Comprehensive
Simões et al. (2015)	Medium			✓	✓	✓				No Eval.
Piggin and Buffey (2016)	High	✓	✓		✓	✓				Comprehensive
Lau et al. (2016)	Medium				✓				✓	Basic
Simões et al. (2015)	High					✓				No Eval.

Table 4.1 High-level Overview of ICS Honeypot Implementations

High-interactive honeypot deployments are rarer than low-interaction ones. This is mainly because of the higher cost, maintenance and development time. Nevertheless, they are able to provide a more realistic environment for attackers to exploit. Because attackers can perform the same actions as on real systems, they are less likely to notice they are attacking a honeypot and are more likely to use all the tools they have to attack it. This should result in higher-value data, as we can see from the University of Toulouse experiment. The main downside, aside from cost, is the risk they pose when they are successfully exploited. When this happens, the attacker can use the system to exploit other devices on the network. The production devices can also be compromised if the honeypot is deployed within the same subnet as the operational network. Therefore, we would advise not to deploy high-interaction honeypots within an operational environment or internal business network but to deploy them in an isolated network within the IP range.

4.2.5 Deployment Considerations based on the literature

When looking to deploy honeypots, we can find some foundational recommendations in existing literature (Dodson et al., 2020), we build upon this with the following suggestions based on the review of the literature in this Chapter. The aim of the configuration should be to mimic a real device as closely as possible. This does also include how the honeypots are portrayed to the outside. From the discussed implementations, we can see that honeypots deployed on university or AWS IP addresses captured less valuable data. A failure to consider the practical deployment environment can outweigh any improvements made to create a more realistic environment.

Because of the specific knowledge required to successfully exploit ICS devices, we believe that the people targeting these devices are generally more aware of how these devices are implemented. This also links back to the necessity of the device's realistic configuration and functionality. Current research tends to focus on general cybersecurity threats, often overlooking the unique challenges posed by ICS-specific attacks and threat actors. To extend on the honeypot itself, further development and the implementation of more data collection/monitoring systems is encouraged. Most monitoring systems should be implemented within high-interaction honeypots by default (e.g. HIDS and NIDS), and these can also improve data collection on low-interaction honeypots. These systems also have bespoke vulnerabilities that cannot be easily patched or mitigated which presents a unique opportunity for honeypots to provide a significant benefit to the security of these systems. Not acknowledging the distinct environment and knowledge required within this environment can lead to the deployment and development of honeypots that can easily be identified by adversaries targeting the systems.

We would argue that because low-interaction honeypots generally capture less valuable data, extracting as much data from them as possible is crucial. The deployment of monitoring systems should always accompany the deployment of honeypots. These systems should be implemented both on the honeypots themselves and around them.

4.3 Enhanced Honeypot Classification

Based on this study on the level of interaction of honeypots, we can see that high-interaction honeypots provide a more significant data set than low- and medium-interaction ones. What we do observe is that emulated honeypots that are deployed in a more interactive environment do provide more data than stand-alone or weakly implemented ones. With this research as a foundation, we provide a new form of classification for honeypots. Aside from the standard low, medium and high classification of the honeypot itself, we focus further on the environment where it is deployed. This environment that we call a honeynet (Spitzner, 2003) should be described with the same categories to avoid the usage of many different categories. Disregarding their own level of interaction, Honeypots can be situated within a low-, medium- or high-interactive environment. This is determined not only by the number of honeypots in the environment but also by the number of different services and how closely it represents a real organisational network. For example, deploying multiple high-interaction PLC honeypots in an environment with HMI (Human-Machine Interface) honeypots, sensors, and other systems commonly found within these networks would be a high-interaction honeypot within a high-interactive environment. A limited amount of high-interaction PLC honeypots in a stand-alone network would be a high-interaction honeypot within a low-interactive environment. Although we discuss the interactivity of honeypots, depending on the monitoring systems on, around, and within the honeypots, the value and amount of data might differ.

The environments can be defined as followed:

- **Low-Interactive Environment**
  An environment which has a limited amount of additional services and systems and does not represent an actual organisational network. Mainly consists of one or more low-interaction honeypots or one high-interaction honeypot which offer limited interactions outside of the honeypot.

- **High-Interactive Environment**
  An environment which has a number of additional services and systems and does represent an actual organisational network more closely. Mainly consists of a mixture

between low-interaction and high-interaction honeypots or multiple high-interaction honeypots which offer an level of interaction similar to that of an operational network.

This further classification provides us with more information about the honeypot implementation at first glance. It also allows for a better understanding of how extensive the amount of data captured by the honeypot implementation might be. We feel that the environment in which the honeypot is deployed gives a better understanding of the honeypot itself, and this classification can limit misunderstandings about the extent of the implementation. A graphical overview of how this classification can be used to classify honeypots can be found in Figure 4.1.

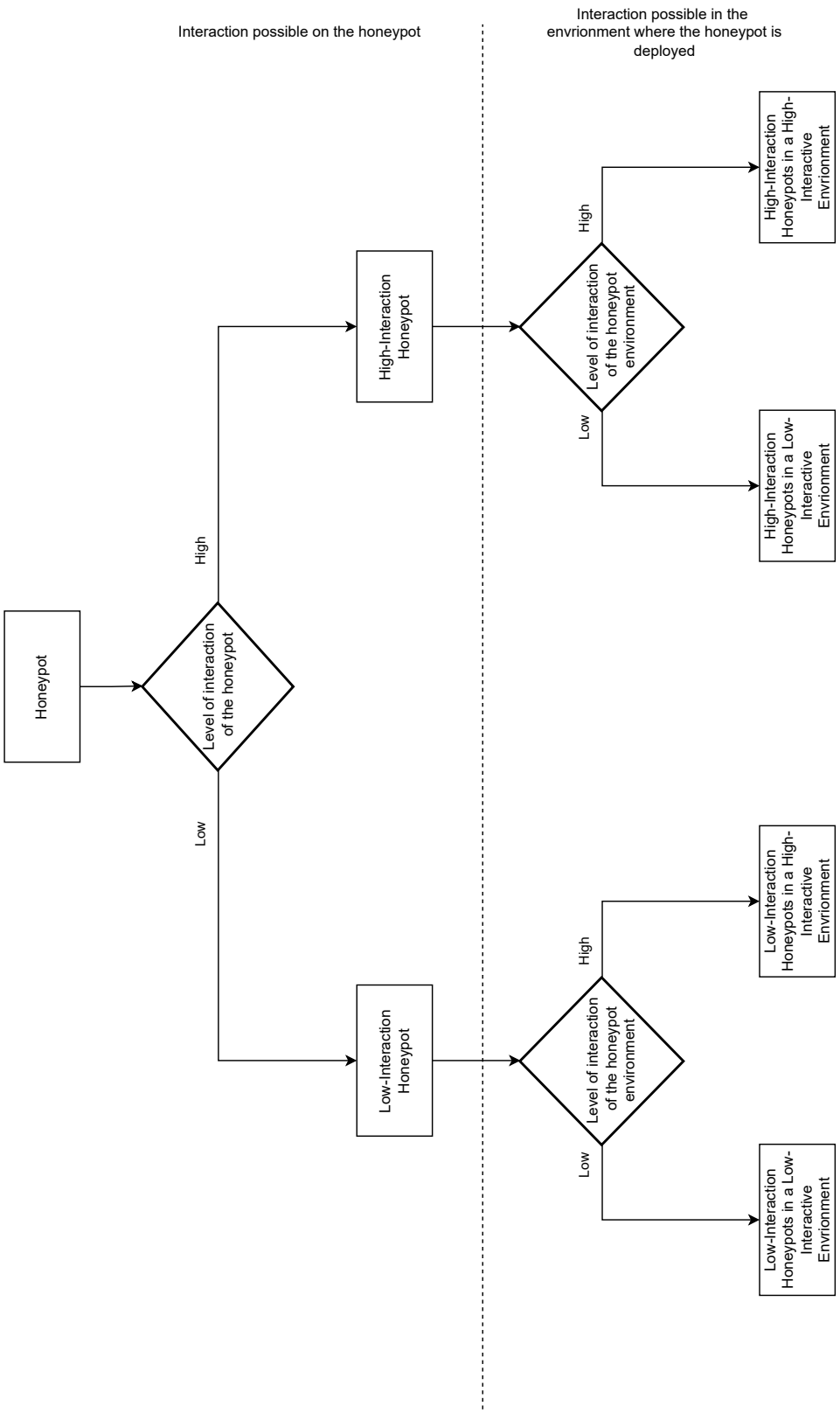


Fig. 4.1 Enhanced Honeypot Classification - Classification Process

4.4 Summary

In this Chapter we further explored the capabilities of different low-, medium- and high-interactive honeypots, by conducting an extensive survey of published ICS honeypot implementations. The outcome of this survey was an extensive overview of honeypot capabilities and their effectiveness in capturing threat intelligence. However, this also uncovered a need for a better classification system as the current three levels of interaction are limited in capturing the actual abilities of the honeypot. Further, we could see that many authors call their honeypots high-interactive because they realistically implement one or some protocols. We would argue that the threshold for a honeypot to be high-interactive also lays providing an attacker with and environment similar to a device within an operational environment. This includes fully implemented protocols but also the possibility to interact with other systems such as the operating system. It is clear that a high-interaction honeypot allows for more interaction than its low-interaction counterpart, but this classification does not require a high-interactive environment around the honeypot. Therefore, we introduced an additional set of classification that focuses on the environment within the honeypot is deployed. Deploying one device as a honeypot does not result in an environment similar to an operational system. Therefore, we would like to call such honeypot networks a low-interactive honeypot network or honeynet, ones that provide an attacker with an interactive environment would be called a high-interactive honeynet. A vital aspect of this implementation lays in providing a better understanding of the implementation of the honeypot as a whole, when discussing a high-interaction honeypot within a high-interactive honeynet we immediately understand the completeness of the implementation.

Chapter 5

An Investigation into the Deployment of ICS Honeypots

In the previous Chapters we have explored the literature on honeypots and investigated how these can fit within regulations applicable in this space. We have seen honeypots can be useful to underpin many aspects of regulation, especially within the NIST Framework and CAF, which we have explored in-depth. Regarding the available literature on honeypot deployments we have seen that there are many ways to deploy a honeypot or multiple honeypots. This now builds upon this Chapter where we investigate how different deployments of a honeypot can lead to different results in data obtained.

5.1 Introduction

Honeypots deployments are commonly classified into two, and sometimes three, types: low-, (medium-) and high-interaction. The difference between these types lies in the deployment, maintenance and, most important, data capturing capabilities. Generally speaking, the higher the level of interaction, the more data the honeypot can capture (Spitzner, 2002). This is because a high-interaction honeypot is, in essence, the same system as would be running in the operational environment but with the addition of monitoring systems such as host and network intrusion detection systems. Therefore, high-interaction variants are less likely to be identified as honeypots (Chamotra et al., 2011). Within the ICS environment, these differences are even more important. As attackers are generally more knowledgeable, they could easily spot anomalies within the honeypot. Advances in process comprehension (Green et al., 2021) not only reduce cyber attack cost (Derbyshire et al., 2021b) but also allow modern adversaries to identify functions running on PLCs by reading the underlying memory structure. This means

that honeypots do need to take the mechanism of process comprehension and the memory structure of a real system into account. Automated attacks would also be interrupted as they would not receive the appropriate response (Rowe et al., 2020), which is easily built in a script. Therefore, a honeypot needs to resemble a real system as closely as possible.

Further, due to their critical functions, any attack can have significant effects. However, from a security perspective, legitimate requests should not be blocked as that can have an equally dangerous effect. Therefore we need to find a balance. Honeypots can aid in this as they can try to redirect malicious requests to those systems instead of the operational infrastructure and provide us with threat intelligence to be used in other security measures.

This Chapter investigates the deployment of honeypots and discusses the strengths and weaknesses of these deployments. Further, we use well-known Internet scanners to scan the Internet for ICS infrastructure and honeypots. We utilise this data to provide an overview of the differences between real devices and Conpot honeypots. These weaknesses result in red flags for attackers targeting ICS systems and should divert them from these honeypots. Scanners such as Shodan, and its Honeyscore, should be able to pick up on these discrepancies (Rowe et al., 2020).

5.2 Honeypot Deployments

5.2.1 Low-Interaction Deployments

Due to the nature of low-interaction honeypots, they tend to be easier to identify. A study analysing the fingerprintability of GasPot concluded there are generalisable fingerprinting schemes (Zamiri-Gourabi et al., 2019). There are several issues that are more linked to the nature of low-interaction honeypots, but others can be easily avoided. An Automatic Tank Gauging devices running on an IP address owned by Digital Ocean is strange. Not changing default configurations leaves traces that knowledgeable hackers will identify. These are two examples of clear misconfigurations that should not happen. Further, discrepancies such as increased delay can also have an effect on the believability of the honeypot (Tsikerdekis et al., 2018). One simple spelling error can lead to the attacker identifying the honeypot (Mokube and Adams, 2007).

One of the most popular low-interaction ICS honeypots that also encompassed a significant amount of the low-interaction honeypots reviewed in the previous Chapter is Conpot (MushMush Foundation, 2018), which is capable of mimicking a range of devices. Further, Conpot can be reconfigured to fit the purpose of the honeypot better. Through a search for Conpot deployments, we can see that there has been a wide range of studies that

deployed Conpot in different environments (Maesschalck et al., 2022). There have been other low-interaction honeypots such as SCyPH (Buza et al., 2014) and Mimepot (Bernieri et al., 2019). We can see that SCyPH is promising but lacks the capability to emulate all functions of ICS devices properly. Mimepot does use SDN to redirect malicious traffic to the honeypot and provides the attackers with responses to their requests; this reduces the chances of the honeypot being discovered. However, the study did not include a thorough evaluation.

5.2.2 High-Interaction Deployments

High-interaction honeypots are based on real devices or systems. They can be virtualised such as a Linux system running on top of a hypervisor (Sochor and Zuzcak, 2016), but for ICSs they are physical devices. These physical systems, such as PLCs, are deployed with monitoring systems installed around them. For IT systems this can also include the usage of tools on the system, such as keyloggers, to monitor activity on the system as encryption can pose difficulties to obtain all the data from network traffic (Balas, 2005). Fingerprintability of these systems differs enormously as there are no software packages available to readily deploy a physical PLC as a honeypot. Similar to low-interaction honeypots the IP address is important to consider.

Within the honeypots deployed there are issues to be found. Trend Micro (Hilt et al., 2020) deployed several PLCs as a high interaction honeypots, however, they used multiple different brands whereas it is more common to see the same brand of PLCs deployed within one environment. Another honeypot by Simões et al. (2015) was deployed without an actual industrial process connected to it, resulting in the data on the PLC being static. These are a few of the examples of how the deployment of a high-interaction honeypot can hinder their full potential. Although, it is important to note there has not been a significant amount of research into high-interaction ICS honeypots. Possibly due to the cost and specific knowledge required to operate and deploy then.

5.3 How do Honeypots Compare with Real ICSs

In this section we aim to establish how a well-used and well-known ICS honeypot, Conpot, compares with a real ICS system. This is important to consider how close a low-interaction honeypot gets to the actual operation of an ICS, in this case a PLC, it is supposed to represent. We do this by comparing a baseline Conpot configuration with an actual Siemens PLC, given the default configuration should represent a Siemens S7-200 PLC. Further, we investigate both systems from an attacker perspective using NMAP.

5.3.1 Setup

To have a baseline configuration to assess Conpot, we deploy a default Conpot configuration within a lab environment. This is done on an Ubuntu 20.04 machine. Further, we deploy a real Siemens S7-300 PLC within the Lancaster ICS lab (Green et al., 2020) in its basic configuration. To verify the behaviour of both devices we utilise Snap7, which is a library for interfacing with Siemens S7 PLC.

5.3.2 Methodology

To evaluate the obfuscation capabilities of Conpot, we investigate how we can differentiate the default Conpot configuration from the default configuration of a real PLC. We will be looking for common signatures of Conpot and discrepancies in services running on the device. To assess both systems from a reconnaissance phase, we will be running several NMAP scans. The first NMAP scan is an adapted slow comprehensive scan:

```
   nmap -sS -sU -T4 -A -v -PE -PP -PS80,443 -PA3389 -PU40125 -PY -g 53
-script "default or (discovery and safe)" IP-ADDR
```

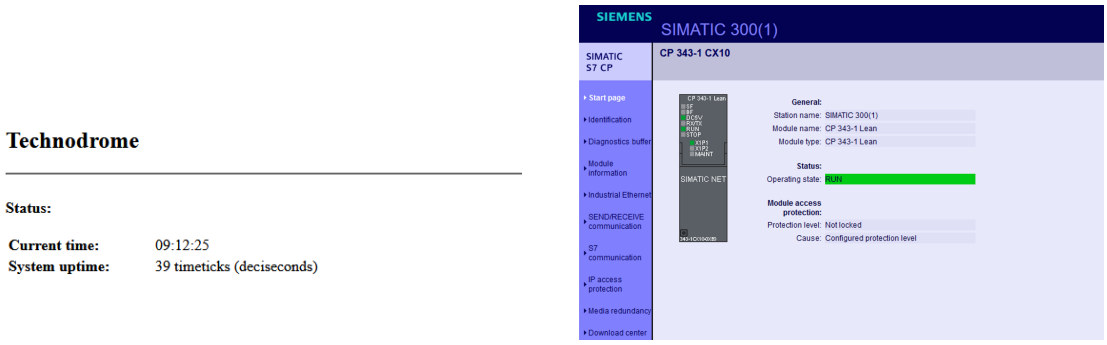
Then we run an normal intense scan:

```
   nmap -T4 -A -v IP-ADDR) and a scan targeted on port 102 (nmap -sS -sU
-p 102 -T4 -A -v -PE -PP IP-ADDR
```

Afterwards, we will run an Snap7 script, attempting to establish a connection, upload a data block and retrieve the CPU info.

5.3.3 Conpot

When looking at the services running on a Conpot deployment, we can see FTP running on port 21, S7 on port 102, Modbus on port 502, TFTP on port 69, BACnet on port 47808 and in the default configuration, a web page running on port 80. The OS detection scan guesses three Linux distributions with 96% probability and five other Linux distributions with either 94% or 95% probability. Port 21 shows a default FTP server running. The webpage (Figure 5.1a) running on port 80 contains many references to Conpot, such as ‘Technodrome’ and ‘Last-Modified: Tue, 19 May 1993 09:00:00 GMT’. S7 is one of the most important services for this Conpot deployment. However, the built-in s7-info script fails execution,



(a) Default Conpot Web Page (b) Siemens Simatic 300 Dashboard

Fig. 5.1 Comparison Between Conpot and S7 PLC Dashboard

which might be due to the emulator not supporting all functions. SNMP, running on port 161, is identified as pysnmp and lists Siemens, SIMATIC, S7-200 as system description. We can already see multiple failures within the default Conpot configuration that have to be resolved to properly obfuscate the honeypot.

Attempting to connect to the Conpot, we can successfully establish a TCP connection and a further COTP connection. Both of these messages get an acknowledgement. However, when we request data from the device, there is no response. Querying the device for CPU info results in a TCP: Connection Timed Out exception, as does sending a request to upload a data block. This leads us to think that there is no real environment connected to the device and would be a red flag to attackers.

5.3.4 PLC

When interacting with real PLC systems, we see a similar result within the NMAP scans. Unlike with the Conpot honeypots, of which NMAP consistently guesses Linux as one of the possible operating systems, real PLC systems provide a more mixed result. With one of the real PLCs, NMAP guesses Thomson TG712 DSL Router, Microsoft Xbox game console, HP PSC 2400-series printer and Wyse ThinOS 6 as possible operating systems with 90% or higher possibility. From this result, it is clear that NMAP does not have extensive experience with PLC systems and their specialised operating systems.

A comprehensive NMAP scan further affirmed that NMAP cannot determine the exact OS. A more in-depth port-specific scan revealed the iso-tsap service with version Siemens S7 PLC running on port 102. Further specific information gives a module serial number, version, system name and copyright saying “Original Siemens Equipment”. Retrieving this data on a known Conpot system resulted in an error. There are no indications that this system might be a honeypot. Further, another real PLC gives us a dashboard (Figure 5.1b) when

interacting with it, which is very different from the Conpot page seen in Figure 5.1. This allows us to gain more information about the network and device; this is not available on a Conpot system.

If we sent the same pattern of messages as we did with the Conpot deployment: a TCP connection followed by a COTP connection and a data request, we get more information. The first two messages got the same response as with the honeypot, but if we request data, then we do got a response from the PLC. We also received a response when asking the device for information on its CPU. When we tried to upload a data block, an exception was thrown stating the function is not authorised for the current protection level. This shows that the PLC is managing another device, or there is at least simulated data on the system. However, based on all the other characteristics, there is no clear indication of a simulated environment.

5.4 Improving Upon Honeypot Configurations

5.4.1 Low-Interaction Honeypots

From looking at the Conpot configuration files, we immediately see that the S7Comm implementation is limited. Nine of the eleven requests listed result in a 'request_not_implemented' exception. Within those nine requests, we could find read, write, download and upload. These are basic requests for any PLC; not implementing these results in a severe lack of features.

To improve upon the Conpot configuration it is important to determine the purpose of the honeypot. This includes reducing the protocols that are emulated, such as (T)FTP, and aiming to enable the protocols generally seen with a real PLC deployment. Although every protocol can improve upon the data collection of Conpot, there could also be flaws in the emulation as they do not respond as expected. Another fault we see on several deployments is the SSH service that is sometimes running on the honeypot, which both NMAP and Shodan identify as the Ubuntu version of SSH as it stems from the operating system. In regards to the HTTP service, it is vital to change the appropriate configuration files before deploying the honeypot. This configuration file includes all of the common signatures we have listed beforehand.

Experiment

We verify these changes by deploying three Conpot honeypots (Table 5.1) within our threat lab and aim for them to get scanned by Shodan. The first of these honeypots is the default Conpot template with SSH disabled. For the second deployment, we disabled SSH, (T)FTP and changed all the common signatures from the HTTP service, which helped us determine

the importance of the common signatures for Shodan to identify a Conpot honeypot. This way the honeypot does not bear any signs of signatures such as 'Technodrome', or the default last modified date. The last honeypot has only the S7Comm emulation was activated to replicate a Siemens PLC running only S7Comm.

We ran all three honeypots twice within our threat lab, once over a 54 day period and once over a 47 day period. During the first experiment we received a combined total of 627 connections over all deployments. Of those connections, there were 308 (49.1%) distinct IP addresses. Overall, Conpot 1 received 257 connections (166 distinct), Conpot 2 received 367 connections (227 distinct), and Conpot 3 received 6 connections (4 distinct) (Figure 5.2). From this overview, we could see the deployment with only S7 received the least connections by far, and the deployment where we obfuscated the default Conpot deployment received the most. The Conpot 2 deployment received more than one third more connections than Conpot 1, making it the most popular honeypot by a significant margin. Looking at the unique IP addresses that only scan one honeypot in Figure 5.3 we could see a slight difference between Conpot 1 and Conpot 2. However, Conpot 2 has received a significant amount of returning IPs.

The second period of data collection saw a total of 1,151 connections over all deployed honeypots. Of those there were 560 (48.7%) distinct IP addresses. Across all deployments, Conpot 1 received the most connections with 632 total (303 distinct), Conpot 2 received the second most connections with 491 total (240 distinct) and Conpot 3 came in with 28 total connections (17 distinct) (Figure 5.4). Similar to the first period we ran these honeypots, Conpot 3 received the least connections by far. However, it received significantly more connections compared to the Conpot 3 from the first experiment. Unlike previously, Conpot 2 received less activity than Conpot 1. The main reason for this is the 163 (T)FTP connections seen by Conpot 1. When we only look at the shared protocols between Conpot 1 and Conpot 2, Conpot 1 has 469 connections which puts it slightly behind Conpot 2. Overall, Conpot 1 and Conpot 2 received a similar amount of connections on the shared protocols with only

	Conpot 1	**Conpot 2**	**Conpot 3**
(T)FTP	✓		
HTTP	✓	✓	
BACnet	✓	✓	
MODBUS	✓	✓	
CIP	✓	✓	
S7Comm	✓	✓	✓

Table 5.1 Overview of Conpot Deployments

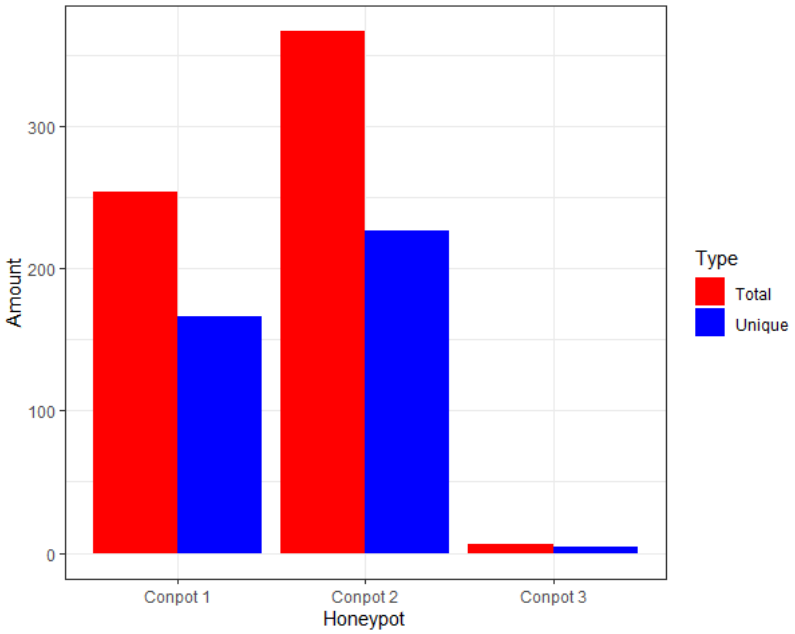


Fig. 5.2 Total Connections for Each Honeypot - First Period

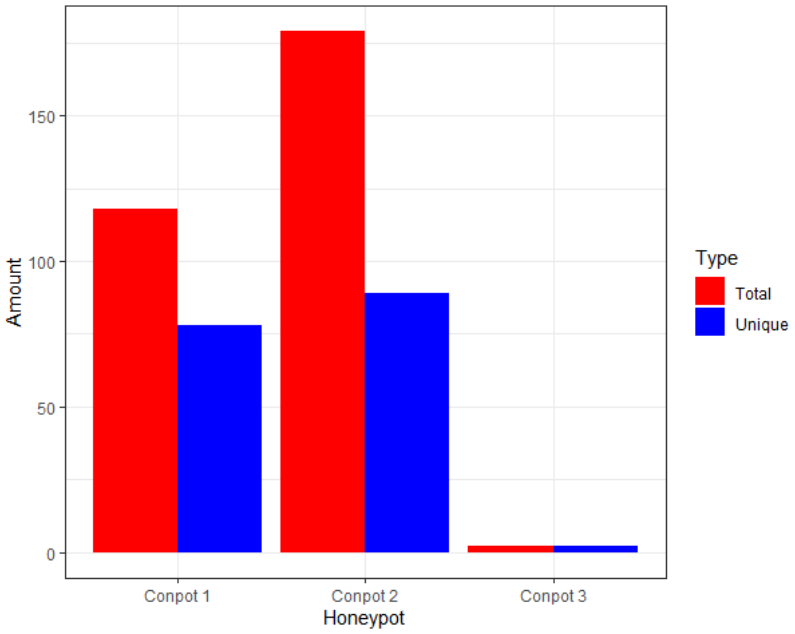


Fig. 5.3 IP Addresses Only Seen on One Honeypot - First Period

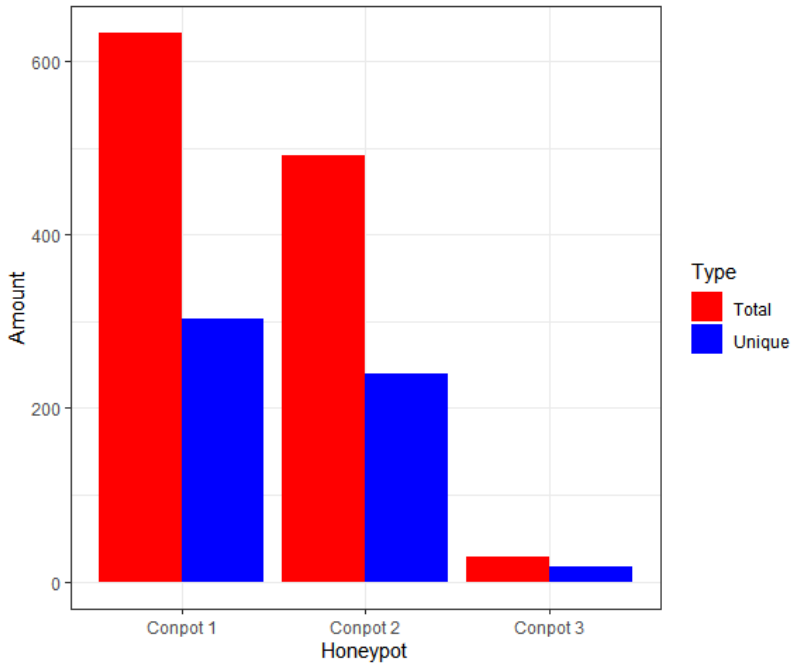


Fig. 5.4 Total Connections for Each Honeypot - Second Period

CIP and MODBUS showing a slight difference in favour of Conpot 2. Looking at the IP addresses that are only seen in one honeypot as shown in Figure 5.5, Conpot 2 (164) received slightly more than Conpot 1 (139) but again has a significantly higher amount of those that are returning (369 vs. 208). Conpot 3 received 1 IP that is not seen by another honeypot, which was only seen once on the Conpot 3.

Throughout both experiments, two honeypots were discovered by Shodan, Conpot 1 and Conpot 2. The time when both honeypots are discovered is different between both periods. Within the first period the time of discovery differs between both, with Conpot 1 being discovered after 35 days and Conpot 2 being discovered after 45 days. Additionally, Conpot 1 was discovered without any further flags, whereas Conpot 2 was discovered and flagged as an ICS. Looking at the average daily connections between all three deployments (Figures 5.6 & 5.7), we could see Conpot 1 saw a significant increase in activity after discovery, whereas Conpot 2 saw a decline of 35%. The time of and way of Shodan discovery over both deployments differed. Conpot 1 was scanned by Shodan over FTP and HTTP protocols, whereas Conpot 2 saw scans on HTTP and CIP (Common Industrial Protocol) protocols. Conpot 3 saw no scans from Shodan.

During the second period, both Conpot 1 and Conpot 2 were discovered quickly. With Conpot 1 being scanned on the CIP protocol after 1 day and Conpot 2 being scanned on the CIP protocol after 2 days. This was not the only difference between both experiments, as

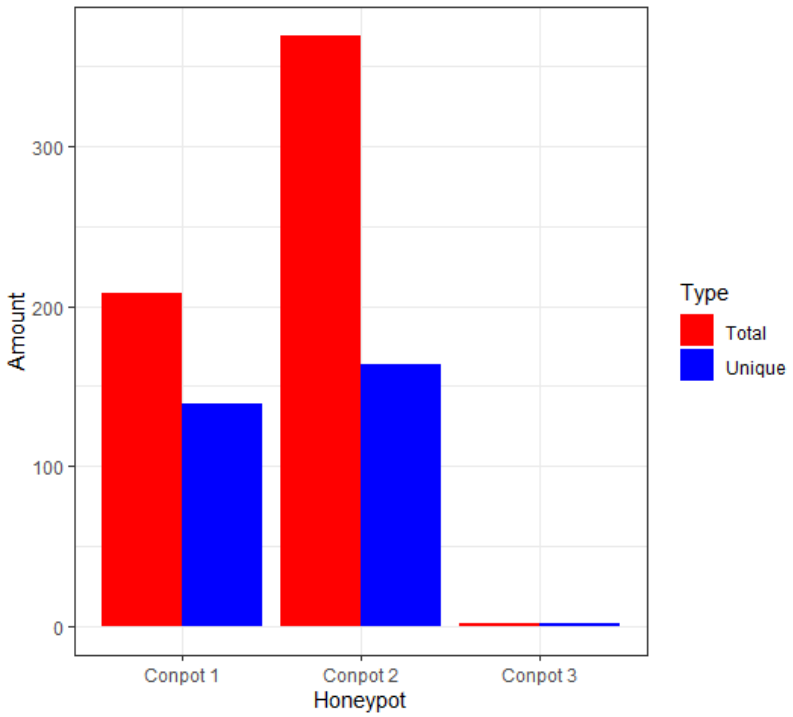


Fig. 5.5 IP Addresses Only Seen on One Honeypot - Second Period

this time Conpot 1 was also flagged as an ICS. Due to the quick discovery the comparison between the average daily connections before and after discovery is not as interesting. The daily connections itself however is. As we can see in Figure 5.8, the connections do not drop to zero anymore on Conpot 1 but are more stable. Compared to the first period, Conpot 1 was also scanned over the CIP, BACnet and TFTP. The only protocol that saw no Shodan scans on Conpot 1 and Conpot 2 in this second period was S7comm. Conpot 3 did not receive any Shodan scans.

Overall, we could see a wide variety in connections on each protocol (Figure 5.9 & Figure 5.10). During the first period we could see BACnet (33%) and CIP (31%) being the most popular and HTTP (19%) coming in third. Looking a the second period, BACnet (24%) and CIP (36%) were still the most popular, with HTTP (18%) again coming in third. However, Conpot 1 saw the most connections over (T)FTP during the first period, but saw the most activity on CIP during the second. For Conpot 2, CIP is the most popular during the second period with BACnet coming second with a significant difference, whereas the first period saw BACnet being the most popular with CIP being slightly behind. Conpot 3 received the most S7 connections of all three deployments during both experiments. When looking at the IP addresses scanning each protocol, we observed during both experiments that each IP only scans one protocol. This could indicate we mainly got internet scanners

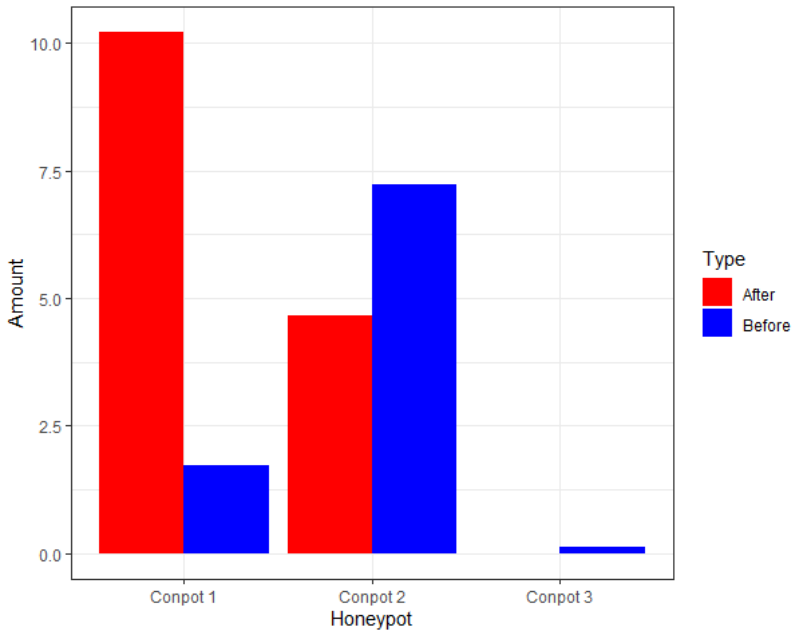


Fig. 5.6 Average Daily Connections Before and After Shodan Discovery - First Period

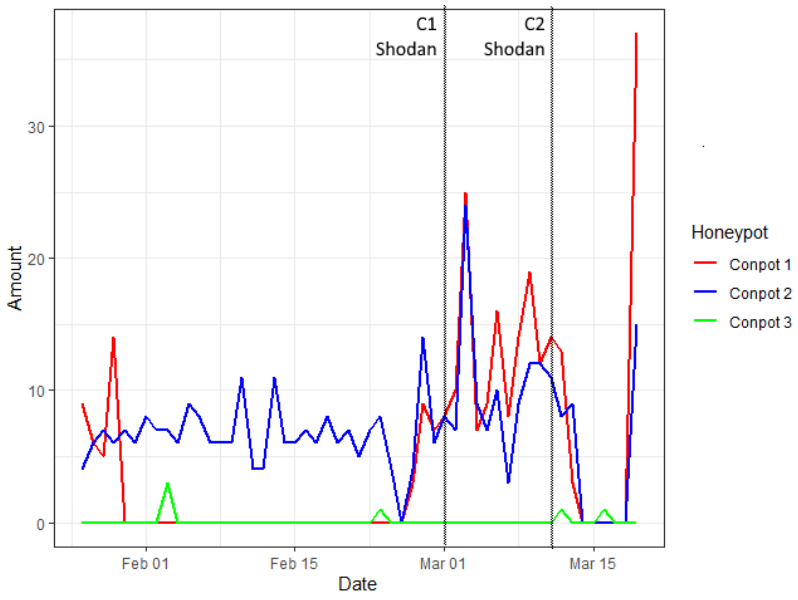


Fig. 5.7 Daily Connections for Each Honeypot - First Period

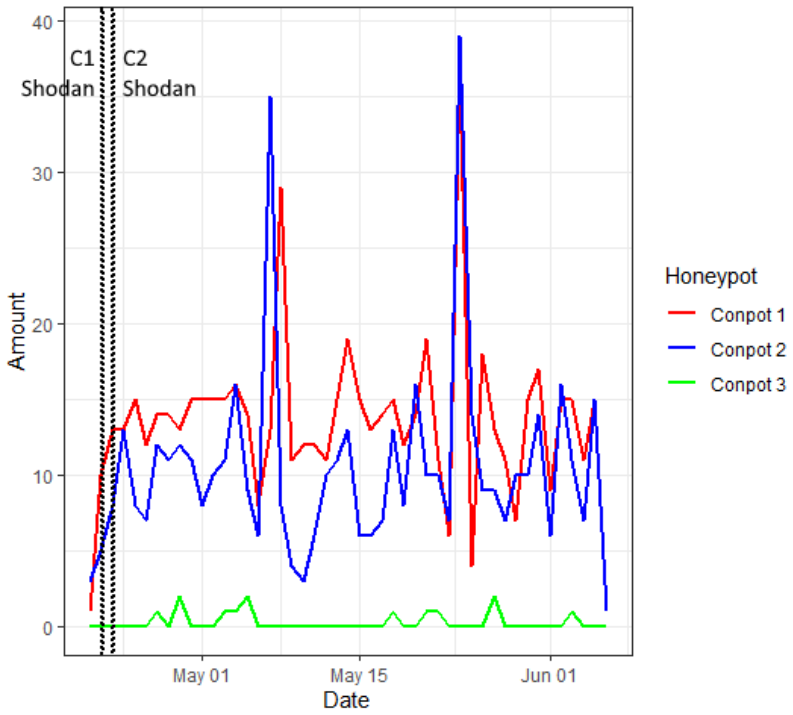


Fig. 5.8 Daily Connections for Each Honeypot - Second Period

that scan one protocol with each address. During the first period, 74 (24%) of the unique IP-addresses that have targeted our systems belong to Censys, this seems to support that statement further. Another 26% of unique IPs belonged to cloud provider Digital Ocean, which makes it difficult to assess who is behind these requests. For the second period, 81 (21.2%) of the 382 unique IP-addresses belong to Censys and 78 (20.4%) belong to Digital Ocean.

Results

Looking back at the activity we have seen on all three Conpot deployments over both periods, there are several conclusions we can make. The main goal of the experiments was to improve upon the default Conpot configuration. Given that Conpot 2 received significantly more traffic than the default configuration in the first period, and of the IP-addresses that only targeted one honeypot, it saw more returning connections during both periods, we have achieved this goal. Disabling (T)FTP and changing the default signatures on HTTP resulted in more activity on protocols linked to ICSs (BACnet, CIP). With regards to S7comm the difference was very small at both experiments. Therefore, we are reluctant to draw a conclusion on S7comm connections. All these factors combined make a strong case that our Conpot 2 deployment is a significant improvement upon the default configuration. Sadly, we saw not a

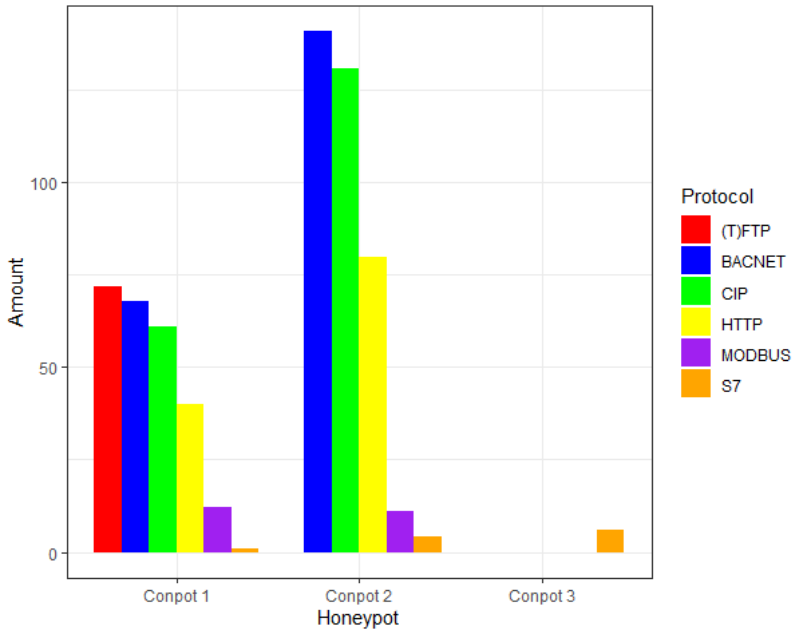


Fig. 5.9 Protocol Connections per Deployment - First Period

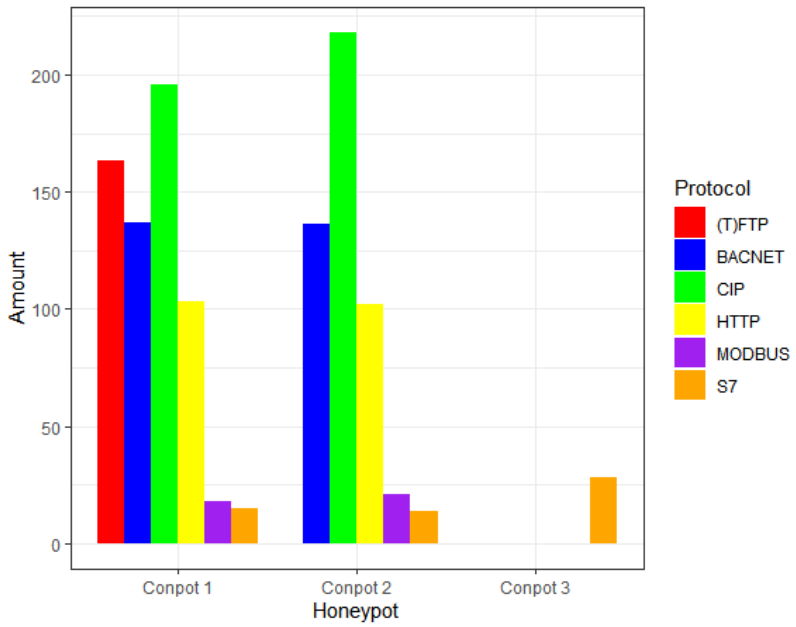


Fig. 5.10 Protocol Connections per Deployment - Second Period

lot of activity on the Conpot 3 deployment in order to draw a conclusion on its effectiveness. However, during both periods it received the most connections on the S7comm protocol, with a relatively big difference during the second experiment. The lack of traffic can be the result of several factors, including not a lot of activity looking for S7comm protocols in general, as we did not see a lot of S7comm connections on other deployments. Further, we acknowledge that the deployment on a university-owned IP-address range can have an impact on attracting real ICS adversaries.

5.4.2 High-Interaction Honeypots

Improving high-interaction honeypots is both similar and in some points more straightforward than low-interaction variants. However, each honeypot can differ from any other honeypot and so does their deployment. Therefore, these improvements are generic to most deployments. Due to the fact these systems are real systems they possess all capabilities to present themselves like a real system. They possess full implementations of protocols used within industrial environments and do not carry honeypot such as running Ubuntu or having a default Conpot configuration. However, these systems can have honeypot characteristics when deployed without due care (Maesschalck et al., 2022). To properly deploy a high-interaction honeypot it is important to make the system portray itself as being actively used. Feeding in data that changes the memory of the PLC, e.g. showing the water level in a tank going up or down, and configuring the system with actual ladder-logic will make the system more interesting for an adversary. Although, the most important consideration to make is if the system is representative of a real system from the outside.

When deploying a realistic high-interaction honeypot this has to be done within a reasonably plausible environment (López-Morales et al., 2020). It is wise to avoid cloud services or educational networks as the autonomous system number might reveal the device is deployed within such an environment. Using a network that is linked to an actual organisation that would be expected to deploy an industrial system, like a PLC, is advised. Being within the university address space was also one of the drawback within the experiment with the Conpot deployments. Furthermore, surrounding the system with other systems makes it more realistic and part of a bigger environment. As an industrial control system is meant to control systems on lower levels of the Purdue model and communicate with systems on higher levels, it would be odd to see them deployed in isolation. Deployments such as the one by Piggin and Buffey (2016) are promising and show the need to properly deploy high-interaction honeypots to present themselves as an alluring target for attackers. Although there are many upsides to high-interaction honeypots the deployment itself requires an equal amount of due

care and diligence compared to low-interaction systems. Particularly due to the risk posed when they are infected, due to their large attack surface (Antonioli et al., 2016).

5.5 Shodan ICS Discovery

One of the goals of this Chapter was to have our deployments scanned and indexed by Shodan. As two of our honeypots were indexed twice, we can claim to have achieved this goal. More interestingly, Conpot 2 was discovered and flagged as an ICS both times, whereas Conpot 1 was only flagged as an ICS during the second period. This seems to be because Conpot 1 was not scanned on the CIP service during the first experiment. During the second period both Conpot 1 and Conpot 2 were also scanned on BACnet, which they were not during the first period. Neither the first or second period saw Shodan scans on MODBUS. Because the first scan before the honeypots being classified as ICS happened over the CIP protocol, and the first period did not see any BACnet scans, they had to have been identified over CIP. This further makes us think CIP is used by shodan to classify ICS devices, but that does not conclude it is the only protocol Shodan uses. As none of our systems were classified as honeypots, more can be done to improve upon honeypot detection, especially the limited behaviour of the protocols can be utilised for this. Looking at the connections Shodan made to both honeypots, we can see Conpot 1 received both FTP and HTTP connections during the first period and (T)FTP, HTTP, BACnet and CIP during the second. Conpot 2 received scans on HTTP and CIP during the first period, and CIP, BACnet and HTTP during the second. We set out that discovery by Shodan would result in an increase in traffic to the honeypot. Although this is the case for Conpot 1, which saw a substantial increase in connections, Conpot 2 saw a decline in average traffic per day after discovery during the first experiment. One caveat we have to make with this is that Conpot 1 was discovered by Shodan 10 days before Conpot 2, which could skew the data slightly. Due to the quick Shodan discovery in the second period we cannot infer any trend from that dataset. In research done by Bodenheim et al. (2014) activity did not increase after indexing in Shodan. In our data we observe both an increase and slight decrease during the first period. However, given Conpot 1 did receive zero activity over a period of time until indexed by Shodan during the first test but did receive constant activity during the second test we can make a preliminary conclusion it does increase activity.

5.6 Conpot Deployments on the Internet

Looking at our research into Conpot deployments we can establish that low-interaction honeypots such as Conpot provide a limited amount of data to investigate and are generally incapable of presenting themselves convincingly as a real system. There is a clear improvement when low-interaction honeypots are deployed in a more realistic environment or as part of a realistic environment. Further, we can see that high-interaction honeypots gather significantly more useful data (Hilt et al., 2020; Simões et al., 2015). However, across interaction levels, we can see that the deployment of the honeypot plays a vital role in the success of the honeypot.

Methodology

To better understand Conpot deployments connected to the Internet, we have used popular Internet scanners to find ICS devices. We will investigate several of the systems returned by Shodan, Cencys and ZoomEye. Unlike Cencys and ZoomEye, Shodan does identify systems as honeypots within the results. This is one of the methods to determine if a system is a honeypot easily. Aside from this, we will be investigating common signatures of Conpot and discrepancies from a real PLC device.

Results

When investigating the PLC devices that can be found through Shodan, we immediately see some honeypots that are wrongly configured. Hundreds of honeypots are found with one or more default Conpot configurations. These include the name 'Technodrome', 'Mouser Factory' as plant information and a serial number of '88111222' (Table 5.2). Some of the PLCs are also deployed on Digital Ocean, a well-known cloud provider, several IP addresses related to the University of Maryland, Nippon Telegraph and Telephone (NTT), and Amazon AWS IP addresses (Table 5.3). Aside from NTT, none of these organisations would not generally deploy a PLC on their network; particularly cloud deployment is unseen. For an experienced attacker, any of these signatures should be a red flag. In the case of The University of Maryland, we get 119 results when searching for Modbus devices within one of their network ranges (129.2.27.0/24).

Some organisations do change some parts of the default configuration but still leave obvious red flags on the system. One of these examples can be seen in Figure 5.11, which is a Conpot honeypot deployed in Poland hosted by OVH ISP. We see they changed the PLC name on the webpage but failed to change the information linked to the FTP service running on port 21. The FTP server still shows 'Technodrome' and 'Mouser Factory', which is a

Conpot Signature	Results
PLC name: Technodrome	319
Plant identification: Mouser Factory	319
Serial number of module: 88111222	336
Last-Modified: Tue, 19 May 1993 09:00:00 GMT (HTTP)	187

Table 5.2 Common Conpot Signatures on Shodan

Organisation	Amount
University of Maryland	241
NTT	34
Choopa, LLC	10
Amazon.com	6
Digital Ocean	3

Table 5.3 Top Organisations for 'PLC Name: Technodrome' on Shodan

clear sign of a Conpot deployment. We could also see some deployments from the Australian Academic and Research Network, where the TCP server also still contains clear indications of a Conpot honeypot.

In Table 5.4, we can see that Censys produces a similar result when searching for these common signatures. However, when using ZoomEye (Table 5.5), these results are significantly increased. From this, we expect that ZoomEye uses another mechanism than both Censys and Shodan. As it focuses on the discovery of ICS devices Xu et al. (2018), it might find more than others, but further investigation has to be done to find the reason for this discrepancy.

Conpot Signature	Results
Technodrome	385
Mouser Factory	327
88111222 & port:102	348
Tue, 19 May 1993 09:00:00 GMT & port:102	244

Table 5.4 Common Conpot Signatures on Censys

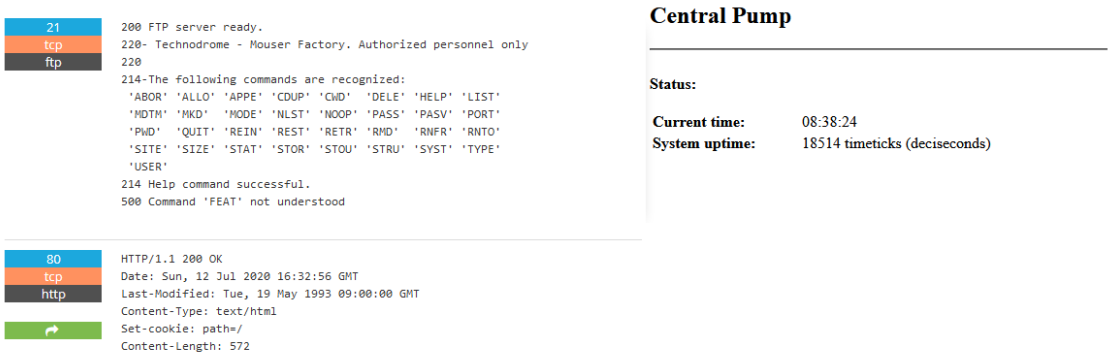


Fig. 5.11 Example of Obfuscation Attempt

Conpot Signature	Results
Technodrome	4,442
Mouser Factory	2,951
Serial Number: 88111222 & port:102	1,015
Last-Modified: Tue, 19 May 1993 09:00:00 GMT	3,442

Table 5.5 Common Conpot Signatures on ZoomEye

It is clear that these Conpot deployments we have found are not representing real PLC devices. They are wrongly configured and easy to spot. We, therefore, are sceptical about their usefulness. There are quite a few common Conpot signatures, and these are scattered across the honeypot. We can see in Figure 5.11 that even though there was an attempt to obfuscate the honeypot, there were still common signatures found in other parts of the configuration, such as the FTP server. Going to a non-existing web page also gives an error referring to Linux Apache. The slightest hint to one of these signatures should be a red flag for any attacker. Nonetheless, these deployments might still capture automated scans and attacks which could be one of the main reasons to deploy low-interaction honeypots. However, we believe that even these attacks could quickly check the deployment for honeypot signatures or check if Shodan has it categorised as a honeypot. To mitigate these common signatures, the default template should not be used or should be reconfigured to mitigate these shortcomings.

5.7 Related Literature

There have been several studies into the identification of honeypots, however, not many of these focus on the improvement of existing honeypots to decrease the risk of being identified.

A framework by Srinivasa et al. (2021) has been developed to aid with the fingerprinting of several honeypots, including Conpot. The framework leverages both a metascan- and probe-based pipeline utilising portscans, banner checks, Censys and Shodan for fingerprinting. They have deployed a Conpot 0.5.2 honeypot running HTTP, Modbus and S7 protocols in a lab environment to evaluate their tool. The authors have detected 884 Conpot honeypots out of 21,855 honeypots identified on the Internet using a ZMap scan. The study also did not investigate the impact of decreasing default signatures on the fingerprintability and detection of the Conpot honeypot by Shodan.

Another study by Vetterl and Clayton (2018) scanning the Internet for several different honeypots, including Conpot, has identified 87 Conpot instances. This study was limited to HTTP detection of Conpot and did not interact with its OT protocols. Therefore, presenting a more limited study in relation to Conpot. This study also did not look into the effect of a reduction of default signatures on the fingerprintability and discovery of the Conpot honeypot by Shodan.

A study by Zamiri-Gourabi et al. (2019), focused on ICS honeypots fingerprintability, has identified a similar number of Conpot honeypots based on default signatures. Identifying 240 honeypots on Shodan and 133 on Censys for the ‘last modified’ signature and a similar amount for other signatures. However, the main study was focused on GasPot and did not go in-depth in the effect of reducing fingerprints and the identification by Shodan.

5.8 Conclusion

Within this Chapter the importance for proper honeypot deployment has been reiterated multiple times. The experiment has been shown through experimentation that small changes in honeypot deployments can result in significant changes in data capturing. It can be seen that there are many inconsistencies between a default Conpot deployment and an actual PLC device. The honeypot package investigated within this Chapter should have raised many questions to real attackers. For one, there are a lot of common Conpot signatures, which should have been obfuscated before deployment. There are many references to the actual operating system of the device, Linux, which will put off any knowledgeable attacker. Deployments such as these would not provide valuable threat intelligence to the organisation

deploying them. If an attacker were to attack a similar deployment, they would notice that no data is returned when requested, although a connection can be set up.

For a Conpot deployment or any low-interaction ICS honeypot, there are many obstacles to overcome. Due to the simulated environment, there is a general lack of interaction. This does extend to the operating system, which is a major red flag for any attacker. ICSs are hosted with specific hardware and are situated within environments that the general population does not generally interact with. Because of this, an attacker that would typically target these environments would be knowledgeable and able to spot these inconsistencies. The rise of search engines as Shodan only increases the importance of obfuscation of the honeypot, as once it is classified as a honeypot, the system would see less valuable interactions. Our own experiments showed that our slightly obfuscated Conpot deployment received more connections on ICS-related protocols and more returning traffic. In contrast, the default Conpot received many connections over (T)FTP, with (T)FTP being the most and second most popular protocol in the first and second experiment, respectively. We also established Shodan leverages CIP as one of the protocols to identify a system as an ICS, but does not perform an in-depth scan that identified any of our discovered systems as a honeypot.

With the inclusion of Shodan within NMAP, it becomes even more important for honeypots to be configured appropriately. It is easy for attackers to detect Conpot within their NMAP scan, a default step in the reconnaissance phase. Further integration with and improvement of Shodan should provide even more information and makes the discovery of many honeypots even easier. We believe that for proper obfuscation of a low-interaction ICS honeypot, a lot of time and resources would have to be spent to simulate an actual device accurately. Nevertheless, fooling a real attacker that is targeting ICSs, remains unlikely through a low-interaction ICS honeypot. We would recommend a high-interactive variant to gain the most valuable data. However, a high-interactive honeypot by itself needs a lot of care when being deployed and throughout its operation. It eliminates many of the issues related to low-interaction honeypots, but does not mean it is easier to deploy. Both types have their benefits when properly deployed and should be considered based on the purpose of the honeypot.

The most important aspect of deploying honeypots has to be the deployment of the right honeypot for the right reasons. Depending on the data the organisation wants to obtain, the risk they want to take and the resources they have available. In many cases it would be more beneficial to deploy a well configured low-interaction honeypot than a badly deployed high-interaction one. Especially when risks are high within an environment such as the operating environment where a high-interaction honeypot, no matter how well deployed, poses many risks. In this case a low-interaction honeypot would satisfy goals of being alerted

when malware is targeting ICS protocols, when there is a malicious user scanning ICSs and other situations. A high-interaction honeypot would be more adequate within a separate honeypot network where it can allow adversaries to propagate and continue interacting with the system for the complete cyber kill chain (Assante and Lee, 2015) and capture zero-day attacks (Portokalidis and Bos, 2007; Ranger, 2020).

Chapter 6

The HoneyPlant Framework for ICS Honeypot Deployment

As we have identified in the previous Chapters, ICSs can be deployed in highly regulated environments such as energy and utility sectors or in environments that require a high amount of up-time. It is clear that honeypots have the capabilities of improving the security of such environments when properly deployed. However, for those highly regulated environments, honeypots can provide another benefit as they can be leveraged to (partially) comply with several of these regulations and guidance. We have yet to see a clear framework to deploy honeypots and use them to be compliant with standards and guidance. This Chapter presents HoneyPlant, a high-level framework for the implementation and deployment of honeypots within an ICS context. This framework can be used in line with regulations, such as the UK NIS Directive through the Cyber Assessment Framework, and as a part of standards such as the NIST Cybersecurity Framework.

6.1 Introduction

The main aim of the HoneyPlant is to support ICS operators with the deployment of honeypots within their environment. To provide a clear overview how their deployment can assist with regulations and provide context on which type of honeypot fits a certain environment to provide increased security with limited risks. Further, we investigate if the implementation of our framework could have made a difference in the protection against historic ICS attacks. For avoidance of doubt, the main goal of the honeypots lie within their detection capabilities. However, we will investigate if detection of events by honeypots deployed according to the framework could have lead to protection against these attacks. Although this framework

can be used as a basis for any honeypot network, we mainly focus on ICS environments and regulation. The security of these environment differs from traditional security in several objectives (Neitzel and Huba, 2014). For ICSs, security should not affect the availability and integrity of the device, unlike IT, where confidentiality is the key focus. We do not want to block any legitimate traffic to the devices. There are differences in physical components, network topology a segmentation between ICS and IT environments that have to be taken into account when deploying an ICS honeypot.

6.2 Framework

The framework can be broken down in two parts, the organisation's network and the external organisations' networks. The latter contains the blacklists and honeypots situated all over the Internet whilst the former one is owned by the organisation itself. These external blacklists are leveraged to gather intelligence on threats on the Internet which have yet to reach the organisation. This will be part of proactive threat discovery and will aid in the creation of resilient network and systems, and general system security. This proactive security aspect is also in-line with the DHS recommendations for ICS security (Fabro et al., 2016), as reactive approaches can be expensive and disruptive to the operations. Within the organisation, there are two distinct parts, a separate honeypot network and the default internal organisational network to which all employees connect. By placing the honeypots inside the organisation its network, we aim to make it look indistinguishable from the other parts of the organisation network and encourage attackers that have entered the network to attack the honeypots. This part of the framework focuses on the threats an organisation faces. It can be used in relation to objectives B, C, and D of the CAF and the protect, detect and respond functions of the NIST framework. Aside from these, it can be used as part of implementing guidance such as ISO 27002, ISO 27019 and NIST Guide to Intrusion Detection. A high-level graphical overview of the framework can be seen in Figure 6.1.

Several considerations have to be made within the organisation when looking to deploy a honeynet according to this framework. First, the high-interactive honeynet has to mimic a field site (Cell/Area Zone (levels 0 - 2)) like the ones deployed within the organisation as closely as possible. This includes the type and amount of devices that are specific to the levels within this zone, and should also take the deployment of commercial ICS software packages (such as ClearSCADA or WinCC) in consideration (Green et al., 2020). Some examples of architectures can be found in the ICC working paper on Establishing Zones and Conduits from the Industrial Cyber Security Center and Kaspersky Lab (Industrial Cyber Security Center and Kaspersky Lab, 2019). Further, to enhance the honeynet, other characteristics

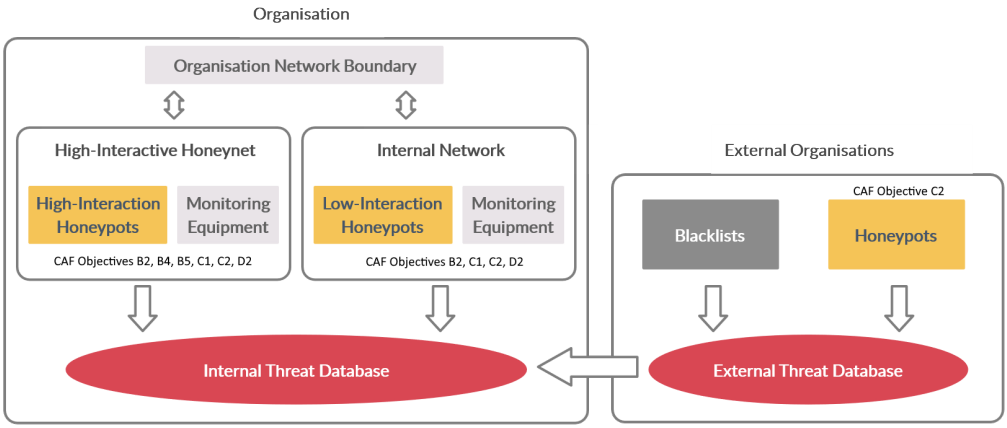


Fig. 6.1 Overview of HoneyPlant

such as latency and external IP address have to be taken into account to create an environment that resembles a field site as closely as possible.

6.2.1 Internal Organisation Honeypots

High-Interactive Honeynet

The high-interactive honeynet is located in a separate network, and its purpose is to trap attackers and capturing data, this will provide more in-depth information as to who is attacking the organisation. Although these honeypots would be deployed within an organisation they would more align with the goals of research honeypots. This network consists of several high-interaction honeypots to provide the organisation with a comprehensive data set. Looking back at the previously discussed implementations, we can see that a higher level of interaction allows the honeypot to capture more data. It also enables the deployment of all ICS protocols supported by the PLC within this environment. To increase the potential of these honeypots further, there should be pre-programmed activity in the network to simulate not only the devices but also the network activity of an operational network. The aim is to make this environment as highly interactive as possible for which we recommend the deployment of a range of honeypots (both ICS and non-ICS) such as Windows honeypots to encompass different levels from the Purdue model. A benefit of some of these honeypots, such as Windows servers and clients, is that multiple honeypots can be deployed on a virtual machine (Alata et al., 2006), without losing the high-interaction aspect. To further emulate a real network environment, several servers (such as FTP or websites) should be deployed as a high-interaction honeypot. By providing both clients and servers, an attacker has more

opportunities to transpose between devices which will provide more valuable data. These high-interactive honeypots could be improved by having them engage with attackers and actively interact with phishing attempts (Li and Schmitz, 2009) as phishing is still one of the most popular ways to gain a foothold in a network. We are calling them bi-directional honeypots, due to the engagement of both parties.

Although high-interaction honeypots pose a significant risk to the network, this is partially mitigated by the segregation between the honeypots and the operational network. All traffic within the honeypot network has to be monitored by an IDS, and hosts must have a host-based intrusion detection system (HIDS) to observe the host. Examples of these HIDSs are AT&T's AlienVault Unified Security Management (AT&T, n.d.) and FireEye's Endpoint Security (FireEye, n.d.). Specific intrusion detection or prevention systems for ICS environments include Check Point IPS (Check Point, n.d.), FortiGate (Fortinet, n.d.), Dragos Industrial Cybersecurity Platform (Dragos, n.d.) and Claroty Platform (Claroty, n.d.). All data captured has to be monitored by the security department to enable them to take appropriate actions. Generally, this part of the framework is flexible, and various amounts of honeypots can be deployed in this location. We agree that this network will require high-maintenance, but it can also provide valuable threat intelligence to be used to lower overall risk and increase security. In terms of legislation, this honeynet would help with Objectives B2, B4, B5, C1, C2 and D2 of the CAF and NIST functions detect and respond.

Internal Network Honeypots

The second location within the organisation consists of low-interaction honeypots and is deployed in an effort to capture threats within the network. Because of this, these honeypots are better aligned with the purpose of production honeypots. In a similar attempt to Simões et al. (2015), we believe that a good low- to medium-interaction honeypot in this location would be less detectable due to being surrounded by real systems. Additionally, small changes it the default configuration of readily available low-interaction ICS honeypots can lead to more valuable activity (Maesschalck et al., 2021). Because these honeypots are low-interaction, they do not require many resources. Therefore we advise organisations operating critical national infrastructure to implement PLC honeypots. Further, we propose the use of operational PLCs to provide the low-interaction PLC honeypot with data. This data could be obtained, and stored in a database, by sending requests to the operational PLCs at startup. We suggest sending the top 20 requests typically requested to a PLC and storing the response to later use when asked by a malicious user, this can be done from a testbed to ensure no sensitive data is leaked. Exposing some data can also lead to process comprehension which would aid the attacker in their goal to infiltrate the systems. This data will make the honeypot

resemble a real PLC more accurately and reduce the chances of discovery. To further increase the chances of discovery, we propose a periodic connection of legitimate internal systems (such as engineering laptops) to the low-interaction honeypots, which can lure attackers to those systems. As with the separate honeypot network, both intrusion (IDS) and host-based intrusion detection systems (HIDS) should be put in place to monitor the network and the systems that host the low-interaction honeypots. The security department should closely monitor the data captured by these honeypots, as any form of malware captured at this stage has potentially already infected other systems inside the network.

Another aim of these honeypots is to provide forensic investigators with more information about the attack that occurred inside the organisation as a honeypot is capable of storing binaries and other attack-related information for further forensic analysis. Once the internal network is infected chances are high that multiple systems are infected or will be breached soon. These honeypots can give system administrators the opportunity to detect propagating malware or attacks quicker and decrease the chances systems are infected for several months or years. Mapping these low-interaction honeypots to the CAF, they would fall under Objective B2, C and D2. Within the NIST Framework they would fall under the detect and respond functions.

6.2.2 External Organisations

The external organisation part of the framework consists of several honeypots deployed and maintained by several external organisations with the purpose of monitoring and capturing malicious traffic on the Internet. These honeypots will capture large Internet-wide scans and attacks. When malicious traffic is detected, threat databases will be updated with newly discovered infected IP addresses and signatures of malware. IT departments can then leverage this data to update their security measures. These honeypots can be located within the same industry as the organisation or can reside in organisations that focus on threat intelligence and honeypots such as BadPackets. Therefore an organisation deploying honeypots according to our framework can also be an external organisation to another organisation's deployment, and vice versa.

The organisation has no control about this part of the framework, but it can still be an important way of gathering threat intelligence. Cybersecurity can and should be a collaborative effort between multiple organisations. If one organisation experiences an attack, other organisations should learn from it and be prepared if the adversaries might target them as well. Due to the nature of ICSs, and the required knowledge, attacks might be re-purposed within other attacks. These external honeypots would fit under CAF objective C2 and the NIST detect function.

6.2.3 Hypothetical Use Cases

The aim of the HoneyPlant Framework is to provide some guidance where honeypots can be deployed. This does not mean an organisation has to follow the framework as a whole, but rather needs to decide what is useful and achievable. For example, a smaller company might not have the adequate resources or capabilities, or a need to deploy a high-interactive honeynet. They could opt to rely on external data or deploy some honeypots within their internal network or both. This needs to be properly evaluated before the deployment of honeypots. Within this subsection we will cover several attack scenarios that parts of the HoneyPlant Framework can help with.

Low-Interaction Honeypots

The purpose of the low-interaction honeypots is mainly to obtain data on automated attacks. They usually require little resources and knowledge to deploy, as some low-interaction honeypots (e.g. Conpot) are available as a prepackaged deployment. Due to their limited deployment it would be unlikely that a skilled-attacker would interact with these systems directly but any tools that would search for specific services or protocols will. For example a script that targets all systems in a network that have TCP port 102 open and run S7Comm would also hit a low-interaction honeypot. These systems will be able to notify the security operations centre of activity within the network they are deployed in. In addition, these systems could be targeted by less skilled adversaries or used to emulate bespoke vulnerabilities. When deployed for dedicated vulnerabilities, the chances a skilled-adversary targets the system might be higher as the services are better emulated and the system is more targeted. These honeypots mainly have a place within the internal network but can also be deployed within the high-interactive honeynet environment. Especially honeypots designed for specific vulnerabilities can be useful within the high-interactive environment.

When an adversary interacts with these systems they are more likely to provide a hard-coded response, depending on the quality of the simulation/emulation. When the quality of the emulation is high, the adversary will receive proper responses and can interact more with the system, potentially revealing more of their capabilities. If this is low, they will not be able to deploy their skills to the full extent, nor will they be able to use many of the vulnerabilities, including zero-days, they would expect in the system. Therefore, it is less likely that they will be interacting with these systems for a long time. However, as previously stated, there is a much lower risk of an attacker taking over these systems which makes them safer to deploy. Strict monitoring of these honeypots is required, particularly when they are deployed in the internal network.

High-Interaction Honeypots

The high-interaction honeypots are deployed to obtain in-depth data on adversary behaviour. As stated earlier, these honeypots provide full capabilities to attackers and are actual systems, such as PLCs, deployed as honeypot. This allows full interaction, and means all vulnerabilities that are inherent to the device, software, operating system etc. are available for exploitation. A properly deployed high-interaction honeypot is capable of fooling skilled-attackers, however, a poorly deployed honeypot can be leveraged for malicious purposes. Within the framework they should only be deployed in an isolated network, i.e. the high-interactive honeynet environment. Deploying multiple high-interaction honeypots strengthens the environment and all honeypots deployed within it as they can be configured to provide a realistic network for an adversary to propagate.

When an adversary interacts with a high-interaction honeypot the system should mimic the responses of an operational system, if deployed correctly. This allows for the organisation to identify patterns of attacks used by adversaries and enabled them to utilise all their tools, including zero-day vulnerabilities. Which means they can be invaluable to obtain important data on threats targeting the organisation, enabling pro-active defence. As established earlier these honeypots can also be leveraged by attackers to further their goal, which is why they need to be carefully monitored and maintained. Therefore, the deployment of these honeypots might not be useful for smaller companies to deploy due to the risk and knowledge necessary for deployment.

External Honeypots

External honeypots are honeypots that the organisation has no ownership over as they are deployed by other organisations. These organisations can be within the same sector, different sectors or even specialising in threat intelligence providing threat intelligence to other organisations. They can consist of both high-interaction and low-interaction honeypots but should include similar systems as the ones used by the organisation. For example, obtaining threat intelligence from systems that are non-ICSs will be less valuable to assess threats to the OT environment. Although the data obtained will not be specific to the organisation, using external organisations can provide benefits without the need for in-house capabilities. It allows for financial planning, and can portray a general overview of threats on the Internet.

6.3 Discussion

We have discussed ICS attacks, ICS regulation (through frameworks, standards and guidance), honeypots and given an overview of ICS honeypot implementations. With that information, we have proposed a new framework. Now we explore how HoneyPlant could theoretically be used to improve the security of an organisation. We do this by giving an example of how honeypots are used to capture malware and link the capabilities of HoneyPlant back to the ICS attacks we previously discussed.

6.3.1 Honeypots and Malware Discovery

As previously stated, honeypots can be used to capture binaries and provide a wealth of threat intelligence. Several honeypots have already been explicitly designed to capture malware. These allow for further investigation of the binaries used and even the detection of previously unseen forms of malware.

The New Zealand Honeynet Project implemented and evaluated Nepenthes (Riden, 2006). They explored how the low-interaction honeypot can be used to alert administrators when there is a network compromise and its effective malware detection rate. The developers of the honeypot claim a detection rate of 73 to 84 per cent for new forms of malware (Baecher et al., 2006). Riden used the Norman Sandbox included with the honeypot environment to perform run-time analysis. The honeypot was listening on 255 IP addresses for five days and collected 74 unique samples of which 48 were identified by the anti-virus software used in the test.

When Dionaea captures malware, it calculated the MD5 hash and uses this as the file name; this way, it does not store the same binary twice (Skrzewski, 2012). Other honeypots can hold the same form of malware multiple times which is a waste of resources. Skrzewski ran Dionaea for over nine months, recorded more than 169 000 attempted connections and captured 537 unique malware samples. Out of 1189 attempted connections, 181 were recorded more than once with the top five being recorded over 150 times.

6.3.2 HoneyPlant and Discussed ICS Attacks

At the beginning of this thesis, Stuxnet, BlackEnergy and Wolf Creek were discussed. In this section, we will theoretically discuss if the implementation of our proposed system could have made a difference within these attacks.

Stuxnet entered the Iranian nuclear facility via a USB drive and then spread through the network. As the malware was not yet circulated over the Internet chances are low, it

would have been detected by the honeypots. Therefore it would not have been instantly blocked within the network. When the malware started distributing itself across the network, it will have attempted to infect the internal honeypot, which then will have detected the malicious nature, and enable the security team to block further infections and respond to already infected systems. In the case of Stuxnet, the malware would have been spotted within the network. This would have allowed administrators to react before any damage was caused.

BlackEnergy initially infected systems via Word documents within emails and was active on the Internet before it infected the Ukrainian facilities. Because of this, honeypots would have detected it in the wild; accordingly, this data would have been used to update blacklists and other security tools. This piece of malware could have been discovered by built-in anti-malware software within mail filtering services like Microsoft Exchange Online Protection before entering the network. It would also have been identified by network monitoring systems or an internal honeypot when it spread within the network. NIST has also recognised this possibility in their Special Publication 800-160 (Ross et al., 2019). Further, emails received by the honeypot could be investigated or automatically executed within a sandbox.

Although the Wolf Creek attack was not successful in infecting the nuclear part of the facility, it still presented a significant risk. If more systems were in place to limit infection and increase the detection of malware, the risk would have been significantly lower. Similar to BlackEnergy, the initial infection occurred through email, which could possibly have been prevented through a mail filtering service if it was aware of the malware. The malware would have been caught by honeypots situated on the Internet or Internal network and spreading over the internal network would have been prevented. In this case, if an infected device were to be connected to the nuclear network, network monitoring systems would then have restricted the infection of the ICSs.

Aside from capturing attacks launched on the honeypots by malicious users, system administrators could pretend to be a non-security conscious user and actively deploy malware within the honeypot environment. This could enable further research into the working of the malware, which can then be used to secure the system against novel malware. This can be done through email, active penetration testing, or USB sticks. These opportunities show that a high-interactive honeynet can be used in several ways and does not solely need to rely on real threats to provide useful intelligence.

6.3.3 Benefits of Honeyplant

As seen in the previous subsection, the deployment of honeypots in line with a framework such as HoneyPlant could have potentially stopped some of the adversarial actions during the Stuxnet, BlackEnergy and Wolf Creek attacks. Additionally, we have established clear

links between the deployment of the modules within the framework and where these fit with the UK Cyber Assessment Framework and the NIST Cybersecurity Framework. Although this is a high-level framework, with several more in-depth suggestions within each module, we believe it could be of great benefit to operational technology engineers when deciding to deploy honeypots within their environment. We have previously discussed the importance of regulation and legislation within ICS, and especially critical infrastructure, environments in which honeypots are rarely included. The establishment of links between honeypots as a security mechanism and as a mechanism to adhere to these regulations provides a clear merit for their deployment.

Previous work (Antonioli et al., 2016; Cifranic et al., 2020; Litchfield et al., 2016; Provos, 2003; Simões et al., 2013) has focused on in-depth ways to deploy honeypots within ICS environments, where the location and their impact on the regulation has not been considered. Within this framework we address the matter of guidance and regulation in relation to the deployment of honeypots, and highlight the importance of considering the deployment environment and then tailoring honeypots accordingly to suit that environment context. By following our general guidance specific to each module of HoneyPlant, and leveraging other resources related to the deployment of ICS honeypots, such as the Conpot documentation, that a tailor-made deployment can be achieved.

6.4 Conclusion

This Chapter presented a honeypot framework that can be used within an organisation to support the deployment of honeypots. This framework is divided into two sections, the organisational network and external organisations. Both these parts can provide a wealth of data, but if an organisation would only focus on one, we would encourage this to be the organisational network. Honeypots deployed by external organisations lack in capturing threats that actively target the organisation, but can capture general trends and attacks targeting similar organisations. The organisational network honeypots are deployed in two separate environments, within the operational network and honeynet that is separated from the operational network. Within the operational network, low-interaction honeypots should be deployed. The separate honeynet should mainly rely on several high-interaction honeypots but can also include low-interaction honeypots; this network should closely represent a typical field site within the organisation. We concluded that this framework fits well within the legislation and guidance previously discussed, and therefore presents both benefits from a security perspective and to support required adherence to legislation.

Chapter 7

Deploying Bespoke Honeypots

Within the previous chapter we introduced HoneyPlant, a framework to deploy honeypots, to help critical infrastructure to deploy honeypots in line with regulations and gain more data from them. Although we have covered which level of interaction to deploy within the areas identified, we have not provided an overview which honeypots. This is mainly because there are a wealth of possible honeypots available, and organisations should decide which of them aligns best with their goals. However, within this chapter we focus on one novel vilnerability we have identified and highlight how a honeypot could be deployed by an organisation to protect against exploitation. A type of honeypot that is designed to detect exploitation of bespoke vulnerabilities. As stated in chapter 2, critical infrastructure presents a intriguing target for nation states and can be targeted through cyberspace. Because these actors are highly knowledgeable they could exploit distinct ICS vulnerabilities.

7.1 Introduction

Over recent years we have observed an increasing level of sophistication in attacks targeting ICSs that underpin CNI (Miller et al., 2021). While legislation has been introduced to support the continued development of defensive strategies (European Commission, 2019), there remains a challenge in using available resources to accurately identify and assess risk. Where risk is viewed as the product of threat, vulnerability, and impact, understanding each element is of equal importance (Derbyshire et al., 2021a). Vulnerability and impact can present a significant challenge to be fully comprehended within an ICS context, as current tooling focuses on traditional vulnerability categorisation at a device level (Denial of Service, protocol limitations, etc.). This limits an assessor's ability to focus on how an attacker could impact operational process behaviour, which is of particular importance when one considers Advanced Persistent Threats (Ahmad et al., 2019). Understanding and exploring

ways to identify possible exploitation of inherent vulnerabilities is therefore necessary, and is something honeypots can assist with. However, we must first understand where bespoke vulnerabilities within ICSs can come from.

This Chapter takes the understanding of ICS device vulnerabilities a step further and investigates a PLC memory vulnerability and how honeypots can be used to mitigate it. More specifically, it covers vulnerabilities within control logic (program code) that is used to monitor, control, and automate operational processes (e.g., a water tank level). As control logic is underpinned by variable blocks (VBs), storing all of the variables used within the PLC, and analysing when they are manipulated provides a view on associated vulnerable control logic. The insights a honeypot designed and deployed for this bespoke vulnerability can provide risk assessors and security professionals more data and also allow for a more coherent picture on the potential impacts should the identified vulnerabilities be exploited, with direct links to specific operational process elements (e.g., a valve within a water tank level control process). We present the results of a practical use case study with commercial-grade PLCs and publicly available control logic, demonstrating the impact of this vulnerability and evaluate how a honeypot can be used to monitor exploitation. In this Chapter we have focused on vendor-provided library functions as these are widely used in practice and present a baseline for real operational systems (Ljungkrantz and Akesson, 2007).

7.1.1 PLC Program Structures

In setting out an approach by which vulnerabilities derived from current PLC programming practices can be identified, it is important first to understand the baseline construction of PLC programs.

While PLCs are available from multiple vendors, and are deployed across diverse industrial contexts (e.g., water, electricity, and gas) (Stouffer et al., 2015), they all execute a series of instructions, i.e., a "program". These programs are, however, more commonly referred to as "control logic" in the industry, and thus will be the adopted term throughout the remainder of this Chapter.

Control logic acts as the intelligence behind operational process monitoring, control, and automation, enacted through its ability to interface with sensors and actuators. While this may be its primary function, the way in which it can be achieved, and the range of additional capabilities it offers, is often overlooked; for example, it can also include the configuration and operation of network protocols, hardware interfaces, SMS alerting, PLC-to-PLC data exchanges, and so on.

Using BSI/IEC standard 61131-3:2013 (British Standards Institute, 2013) as a guide, five control logic programming languages are defined for use on PLCs (Ladder Diagrams,

Function Block Diagrams, Sequential Function Charts, Instruction Lists, and Structured Text). While vendors may choose to develop their own custom languages, they are typically subtle variations of those defined by BSI/IEC. Moving one step further into the control logic ecosystem, we have Program Organisation Units (POUs), applicable across all of the aforementioned languages. The following POU definitions are taken from Green et al. (2021) who use and expand on BSI/IEC 61131-3. We have opted to include this extended definition as it captures one critical element not well covered by BSI/IEC 61131-3, that of variables.

- **Programs**: are the highest level of organisational unit. They control the operation sequence, enabling responses to cyclic, time-based, or interrupt-driven events during program execution. They are composed of specific instructions but also Function Blocks and Functions.
- **Function Blocks (FB)**: contain code that stores their values permanently in memory, remaining available post Function Block execution.
- **Functions**: provide discrete common functionality, for example, ADD or SQRT. Function POUs can use global variables to permanently store data, but do not have their own dedicated memory (i.e., local variables).
- **Variable Blocks (VB)**: store program data in the PLCs memory and can be global (gVB) or local (fVB). The latter of these are associated with Function Blocks to provide long term data storage. The VB is an addition to the POU model as defined by IEC 61131-3. This standard describes the use of variables in a general way, with limited links to their storage.

Control logic can be written in three POUs: these are Programs, FBs, and Functions. Figure 7.1 depicts a basic Ladder Diagram example that is implemented in a FB with data linked to the variables stored within a fVB. This control logic represents two "if" statements. (1) If Example_Input_1 and Example_Input_2 == 1, then Set Example_Output to 1, and (2) If Example_Input_1 and Example_Input_2 == 0, then Reset Example_Output to 0. While this example was constructed using a Ladder Diagram, any of the five programming languages defined in BSI/IEC 61131-3 could have been adopted here (depending on device/vendor support). Furthermore, despite any variation in language, the inputs and output variables would have remained constant. In this example, the inputs and output represent three Boolean variables stored within a gVB.

Additional specialist control logic elements for accessing and addressing system components may also exist depending on the device/vendor; these include peripherals, timers, and pointers. Pointers are perhaps the most common and well understood, due to their use within conventional programming languages. They assist in the reduction of common

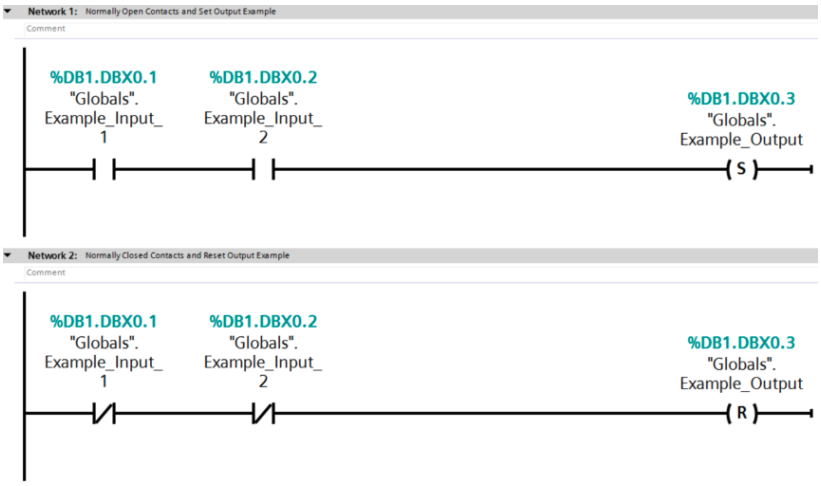


Fig. 7.1 Ladder Diagram Example

data replication across a control logic base (induced through other data access/processing methods).

To support the development of control logic, vendors often produce libraries of Functions and FBs (Siemens, 2019a); these are also referred to as "instruction sets" (Allen Bradley, 2008). Libraries contain pre-written control logic supporting a wide range of functions, from converter operations to communication exchanges. Due to the benefits brought about through the use of libraries (tested, trusted, standardised code snippets that provide significant time and performance efficiencies); their adoption is widespread (Ljungkrantz and Akesson, 2007).

7.2 ICS Vulnerabilities

Research into vulnerabilities within traditional IT systems/software has been at the forefront of security research for several decades. This is in direct contrast with work in ICSs, which has seen an increased focus only in recent years. There has been a shift towards more integrated IT and ICS environments, removing the isolation between the two, rendering ICS environments more vulnerable to external adversaries, including those from nation states (Derbyshire et al., 2018; Miller et al., 2021). Due to ICS deployment issues, there exists a consequent plethora of vulnerabilities. However, research within the ICS vulnerability space has tended to focus on operating systems, firmware, industrial protocols and circumventing traditional security measures (Abbasi et al., 2016; Biham et al., 2019; Drias et al., 2015; Nochvay, 2019; Wardak et al., 2016). This includes research and development into vulnerability scanning tools designed with ICSs in mind (Antrobus et al., 2016, 2019). These tools are critical in

providing information on exploitable vulnerabilities within operational environments during risk assessments.

Within the focused topic area of PLC vulnerabilities, the work of Serhane et al. (2018) provides an overview that spans multiple themes, from protocols to operating systems. Taking a more practical approach, Green et al. (2017) explore how PLC vulnerabilities can be exploited at a network and device level, to achieve operational processes manipulation. Collectively, these works provide a baseline understanding from both an attacker and defender perspective.

PLC programs are developed across five special programming languages (see Section 7.1.1), one of which is commonly referred to as "ladder logic" (ladder diagrams). Ladder logic was originally designed to document the design and construction of relay racks in a sequential process. Over time, ladder logic evolved into a full programming language, and with it, related research. Research on ladder logic by Kottler et al. (2017) introduces formal verification, where Eckhart et al. (2019) allows us to consider wider implications of vulnerabilities within systems through the use of a security development lifecycle. The work of Serhane et al. (2019) has taken research into this space one step further, examining a variety of control logic conditions that induce vulnerabilities. As with conventional programs, adversaries could identify and exploit issues within a PLC program. Therefore, the ability to identify and assess vulnerabilities within programs running on these systems is of crucial importance.

When evaluating PLC programs, specific programming practices have to be taken into account (Fluchs, 2020; Serhane et al., 2019). Current programming practice includes the use of library functions developed and provided by device vendors. This can be seen in Ljungkrantz and Akesson (2007), in which the authors explore programming practices and the use of library functions. We can assert that these functions, and consequently associated memory use, behave in the same way across all devices in which they are deployed. Additionally, not only do PLC programmers rely on these library functions, but they also create their own custom functions, written for use across an entire organization. Identifying vulnerabilities within these functions would indicate that all systems in which they are deployed are vulnerable. Vulnerabilities on this scale would be very considerable, given the international breath of their deployment.

Two notable vulnerability scanners have been developed by Antrobus et al. (2016, 2019). One, SimaticScan, focuses on vulnerability scanning of Siemens PLCs, with the other, PIVoT Scan, focusing on scanning Industrial Internet of Things (IIoT) environments. SimaticScan is designed to detect a range of vulnerabilities within a Siemens PLC, including Web vulnerabilities and unauthorised read/write requests. Within its evaluation, this tool identified

multiple vulnerabilities across the researchers' testbed environment, including one that was previously unknown. In the evaluation of PIvoT Scan it was seen to outperform a popular vulnerability scanner, Nessus. Although Nessus (Tenable, 2022) does contain plugins that support the scanning of ICS devices, it remains a tool primarily developed for use within traditional IT environments. OpenVAS (Greenbone, 2022), another well known vulnerability scanner, lacks ICS specific scanning capabilitites, limiting its value in a similar way to Nessus. While both OpenVAS and Nessus are able to find basic vulnerabilities (e.g., DoS, HTTP-based, and SMTP-based) (McMahon et al., 2018), they fail to identify vulnerabilities directly linked to operational process monitoring, control, and automation. Two additional ICS-specific port scanners that can be used to assess vulnerabilities are PLCScan (Searle, 2015) and ModScan (Bristow, 2008). These tools are not vulnerability scanners; however, their output (software and hardware focused) can be used in conjunction with vulnerability databases to identify the existence of known vulnerabilities. Therefore, they can not be used in the identification of new vulnerabilities that might affect the system under consideration.

Tools such as PCaaD (Green et al., 2021) take an initial step towards the identification of vulnerabilities within PLC programs and allow for the development of process comprehension (Green et al., 2017) without any pre-existing knowledge about the target system; however, they are highly focused and fail to assess previously unseen PLC programs due to their use of signature-based recognition. Thus, honeypots that can identify the exploitation of specific vulnerabilities can be extremely useful to mitigate shortcomings of current security measures.

7.3 Threat Model

Existing work has demonstrated that attacks targeting PLCs can be executed at both the network and hardware levels (Green et al., 2017). These have been witnessed across a range of real-world events dating back to the 1980s (Miller et al., 2021). One of the most commonly discussed attacks, operating at the network level, exploits the use of PLC network protocols to inject unauthorised command messages into a PLC's memory (VBs) (MITRE, 2021; Robles-Durazno et al., 2019). Figure 7.2 depicts normal/trusted operational use of a Human Machine Interface (HMI) by a system operator. In simple terms, an operator will use HMIs (these can be compact touch screen devices (Siemens, 2022), or larger desktop/server based systems (Siemens, 2020c)) to send requests to PLCs. These requests are stored in the PLC's VBs (gVBs and/or fVBs), which in turn are ingested and processed as part of the PLC control logic. The control logic will then operate underlying physical processes (e.g., starting/stopping a pump) via output cards (See the Digital Output "DO" card in Figure 7.2). It

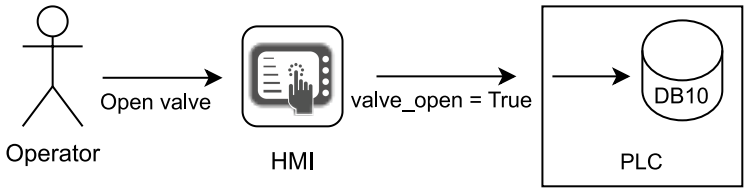


Fig. 7.2 HMI - PLC Interaction Process

is this trusted HMI functionality that attackers mimic when adopting unauthorised command message attack techniques.

As this type of attack can be challenging to mitigate at the network layer (Green et al., 2021), it offers a compelling argument towards identification and remediation at the device level (i.e., within the control logic), and therefore forms the basis for vulnerability identification through the remainder of this Chapter.

Attacking a PLC via the injection of malicious data into its memory requires network-level access via the device's supported network protocol and function (e.g., S7-Comm and Write requests (Molenaar and Preeker, 2013)). This can be achieved in a number of ways, with Figure 7.3 included as a point of reference across the following examples:

(1) The PLC has been configured with a public IP address. Examples of this type of deployment can be seen in Shodan (Shodan, 2020). As the PLC resides directly on a public IP address with no network-based defensive controls, an attacker with Internet access can directly inject malicious data into the PLC's memory with no restrictions.

(2) The PLC has been configured to be located behind a firewall that filters incoming requests. In this scenario, an attacker would be required to compromise a trusted device that is permitted to traverse the firewall and communicate with the PLC.

(3) The PLC is within a network segment that has a poorly protected WiFi network. Once access has been obtained, an attacker would have direct network connectivity to the PLC.

7.4 Proof of Concept

To facilitate practical proof-of-concept exploration of this memory vulnerability a Siemens 300 series PLC was used based on its global adoption (making it a representative use-case) (Siemens, 2020b), Siemens TIA v13 (Professional) platform as a programming agent

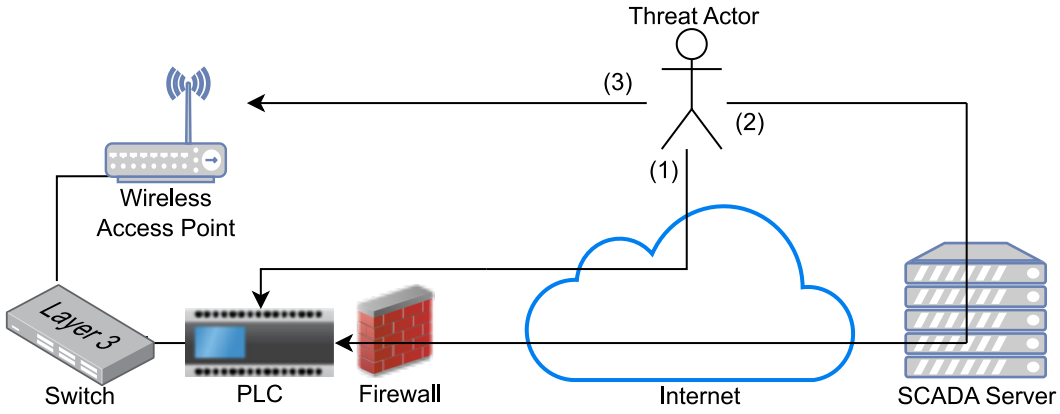


Fig. 7.3 Threat Model

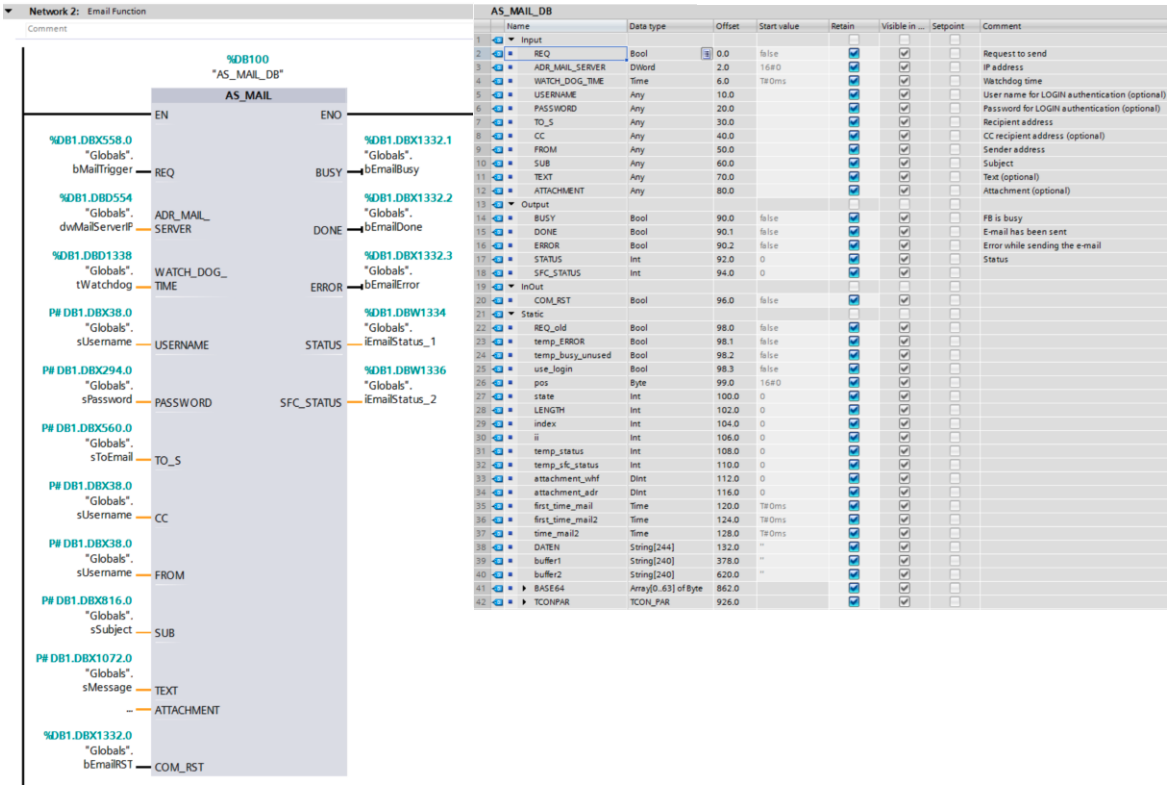


Fig. 7.4 Email Function and Associated VB (DB 100)

(Siemens, 2020a), and a specialised scanner (PLC-VBS) was developed. All evaluation was undertaken within a physical environment deployed following the guidance discussed by Green et al. (2020).

7.4.1 Siemens PLC Ecosystem

The Siemens 300 series PLC used during the evaluation supports Ladder Diagrams, Statement List (Instruction List), Function Block Diagram, Graph, and Structured Text programming languages. During programming, four primary blocks are used to build the control logic: Organisation Blocks (OB), Function Blocks (FB), Functions (FC), and Data Blocks (DB) (i.e., Variable Block POUs). These four blocks map directly with the POUs defined in Section 7.1.1.

Within these blocks, aside from DBs, an engineer can write control logic. As DBs in the Siemens Ecosystem represent VBs, they are used to store variables called OBs, FCs, and FBs. There are additional symbol types where data can be generated, output, and stored, such as I/O Signals (I and Q), Marker Memory (M, MB, etc.), Peripheral I/O (PIW, PQB, etc.), and Timers and Counters (T & C) (PLCDev, 2020). However, for the purposes of this work we will be focusing on DBs as these store the variables related to the linked function which we seek to evaluate. These variables can be of different types, such as int, dword and date. However, different types do not change our method as it checks the constituent bytes that are stored within the DB.

Figure 7.4 has been included to better explain the use of gVBs and fVBs within a Siemens context. To the left of this figure is a single rung of ladder logic. On this rung is the AS_MAIL FB, part of the Siemens TIA Communications library. Once triggered, this FB will send a pre-constructed email to a pre-defined recipient. This could be used as part of an emergency alerting system, notifying system operators of issues with the underlying operational process (e.g., a pump tripping). To the right of this figure is the AS_MAIL FB's associated fVB (DB100), where data for all of its local variables are stored. The left hand side of AS_MAIL in Figure 7.4 depicts a set of inputs, required in order for the FB to operate. These inputs can be specified using four techniques:

- gVBs - As can be seen in "REQ", where the value stored in "bMailTrigger" (gVB address DB1.DBX558.0) is copied/pasted into "REQ" (fVB address DB100.DBX0.0) during every control logic cycle.
- Pointers - As can be seen in "USERNAME", where the gVB address of "sUsername" (P#DB1.DBX38.0) is copied and pasted into "USERNAME" (starting at fVB address DB100.DBX10.0) during every control logic cycle. Here the value is simply read from

the gVB via the pointer address on every cycle, rather than being copied/pasted from the gVB into the fVB.

- Defaults - As can be seen in "WATCH_DOG_TIME" (T#0ms in a grey font). Here, instead of specifying an input value, the fVBs default value of 0ms has been left in place.
- Direct Inputs - Figure 7.4 does not contain an example of this input type; however, it would look the same as a default input, but with a blue font. Here the value is input directly within the ladder logic, and would be copied/pasted into "WATCH_DOG_TIME" (fVB Address DB100.DBD6) on every cpu-cycle, in the same way as if configured via a gVB.

This proof of concept scanner will focus specifically on VBs (i.e., DBs in the Siemens ecosystem), and the ability to manipulate data stored within them via unauthorised command message attacks. From the example provided in Figure 7.4, this would include DB1 and DB100 VBs.

7.4.2 Vulnerability Scanner

Within this subsection we first provide a high-level overview of PLC-VBS, before deconstructing it into multiple phases where we provide further in-depth detail about its functionality. Figure 7.5 provides an overview of the scanner's operation, as a high-level reference point.

When starting the scanner, the user is required to provide the PLC's IP address, the location of its CPU (rack and slot numbers), target VB (DB number), the time period between writing a new test value to a VB and then reading it back (**Read 2 Time**), and the wait-time between each byte being tested (**Next Byte Time**). Also, before starting the scanner, the user must first confirm the system is in a safe state. This typically involves removing any physical outputs to operational equipment (e.g., pumps and valves) and other devices (e.g., PLCs and HMIs), that could enact operational change based on its actions. Removing physical outputs is important, in order to ensure the scanner does not interfere with the physical environment, and does not affect the result. What should not be changed is the PLC program itself (i.e. the ladder logic), as any changes might result in an distorted output. Once a connection to the PLC is established, the scanner begins testing each byte sequentially within the VB. Because all data is accessible in byte form, the scanner does not need to distinguish between data types. The scanner uses S7Comm read and write requests, the same requests used by engineers and HMIs to communicate with Siemens PLCs (Green et al., 2021). For each byte it scans, the scanner attempts to invert each bit in the target byte, before reading it back to

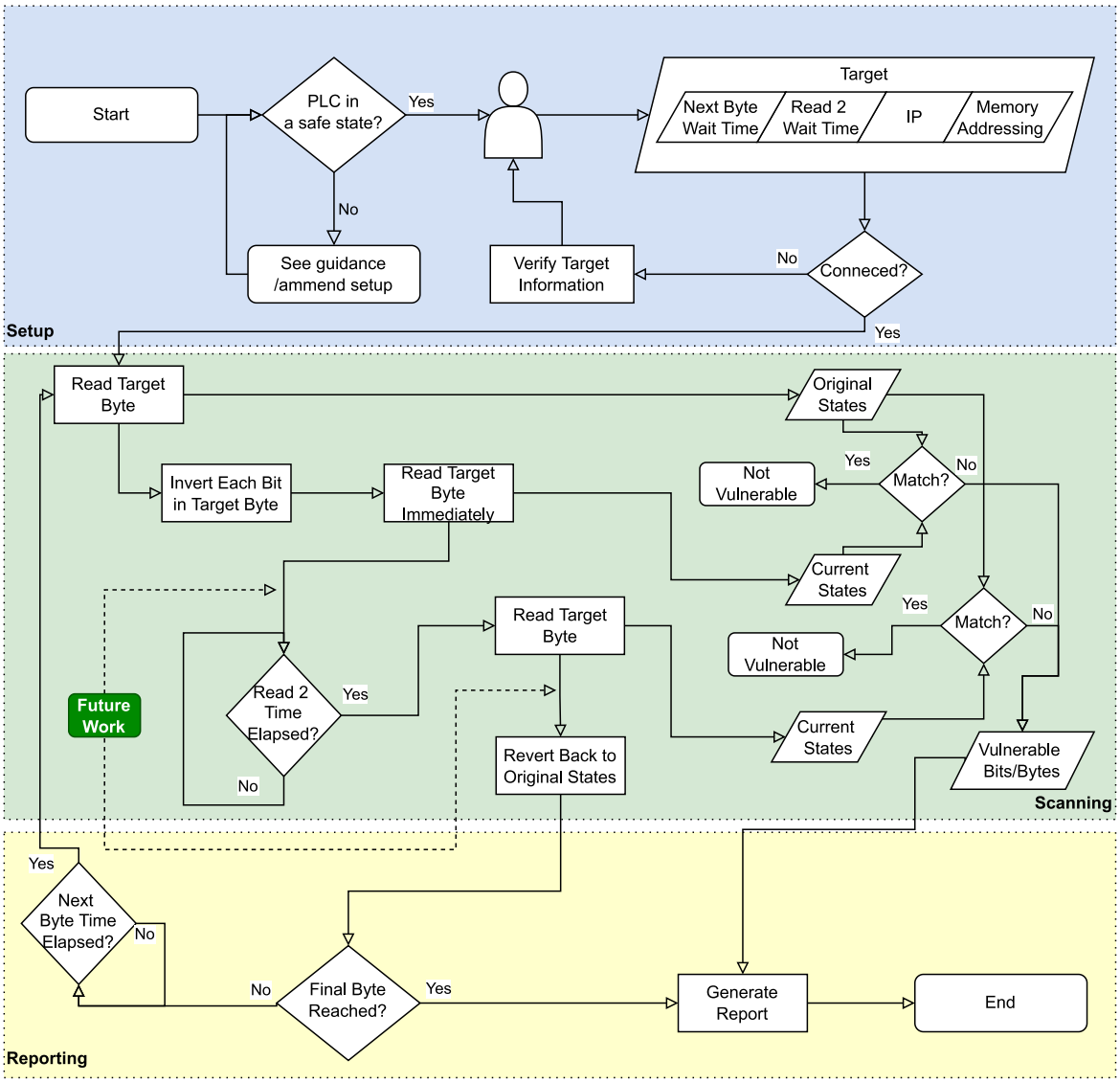


Fig. 7.5 Vulnerability Scanner Process

confirm if the inversion request has been accepted and retained by the PLC. If the states of bits within the byte have changed, they, and the byte itself, are classified as vulnerable. If the byte can be changed this will in turn change the variable that this byte is part of. This change will be visible regardless of the data type of the variable. For example, despite the word data type being 16 bits in length, changing one of these bits will result in a different value. However, for an attacker to accurately change the variable into a specific value, they would need to have knowledge of the data type and which bit belongs to which variable. The scanner then waits until the time specified by the user has elapsed (**Read 2 Time**), before reading the byte again. If the bits have remained in the inverted state, we know the inversion

request has persistence (i.e., you only need to send one write request to change the state of the bits). If the bits have reverted back to their original state, they have been overridden by the control logic, and therefore any attempts to change their state will not be persistent with a single write request. Finally, we revert the byte back to its original state and wait until the defined time period has elapsed (**Next Byte Time**) to begin testing the next byte. When the final byte has been reached a report is shown and the scanning ends.

Scanner Phase 1 (Setup)

This involves evaluating the current PLC state and preparing the scanner for use. Ensuring the PLC is in a safe state is important to reduce the risk of inadvertent operational impact. Once verified, the user can continue with the scanner's baseline setup, which determines what the scanner will target and how it will operate. Aside from the target PLC's IP address and CPU details, two timers must be configured. The first is a wait-time (**Read 2 Time**) between inverting each bit within a byte and then reading it back (to check for inversion persistence); second (**Next Byte Time**) is the wait-time between each byte under test (only a single byte is tested at any one time). Checking for persistence is vital, as some bits will be overridden by other control logic operations, which would then require a consistent flow of write requests in an attempt to achieve the desired level of persistence (an unreliable approach which may be more noticeable on network monitoring solutions). The wait-time between each byte under test is used to reduce load on the PLC, and setting this to 0 could impact the results as the PLC cannot respond in a timely manner or may fail to process scanner requests. We recommend this to be at least 1 second. Finally, the user is required to specify the VB under review, thus constraining the scanner to an approved area of PLC memory. Once the user has completed these actions, the scanner attempts to connect to the PLC, should the connection fail, the user is asked to verify the information provided.

Scanner Phase 2 (Scanning)

This begins once the scanner's initial setup has been completed and a connection to the PLC has been established. Within this phase, the scanner's main objective is to iterate through each byte within the memory space defined during its setup. The scanner starts by reading the first byte within the memory space (S7Comm read request) and stores its state (the byte's **original state**). Then, each bit within the target byte is inverted (S7Comm write request) and the byte is read again. The **current state** of the byte is stored, and used as a comparison against the original state. If these states match, there are no vulnerable bits within this byte, and the byte as a whole cannot be manipulated. This is because the inversion we have done

was overwritten by the control logic before the tool could read the byte again. Where the states do not match, the byte is identified as vulnerable. The scanner then logs both the vulnerable byte and the individual vulnerable bits.

Once the first wait-time (**Read 2 Time**) has elapsed, the scanner reads the target byte for a third time and stores its current state. Then, as before, the current state is compared with the original state. The results of this will determine whether or not the inversion has persistence, and if so, across which bits. The scanner logs both the vulnerable byte and the individual bits. Finally, the byte is reverted back to its original state (S7Comm write request).

Scanner Phase 3 (Reporting)

This has two functions. If the final byte of target memory has not yet been reached, the scanner waits until the second wait-time (**Next Byte Time**) has elapsed. It then returns the scanner back to phase two where it begins testing the next byte. Once the final byte has been reached, the tool generates a report identifying the vulnerable bits and bytes within the scanner memory space. The tool then exits.

7.4.3 Experimentation

Method

In order to evaluate the capability of PLC-VBS to identify vulnerabilities we require access to real-world control logic. This presents a significant challenge, particularly when seeking to identify vulnerabilities, as an organisation sharing internally developed control logic would be unlikely to provide approval for its discussion within the public domain. Therefore, we opted to use Library FBs written by Siemens, included as part of their PLC programming agent TIA Portal (Siemens, 2019b). As per our findings earlier, vendor provided control logic is extremely common, is context agnostic (it can be applied to water treatment, energy distribution, manufacturing processes, etc.), and is internationally adopted, making it an ideal candidate for exploration.

We selected ten FBs, offering a range of functionality, from basic counts, to more complex data exchanges. Each FB has its own associated fVB, as per the example shown in Figure 7.4 and discussions across Section 7.4.1. Furthermore, these fVBs are accessible over the network, so are directly applicable to the approach adopted by our scanner. Where possible, each FB was configured multiple times, adopting three of the four configuration input techniques described in Section 7.4.1:

- gVBs

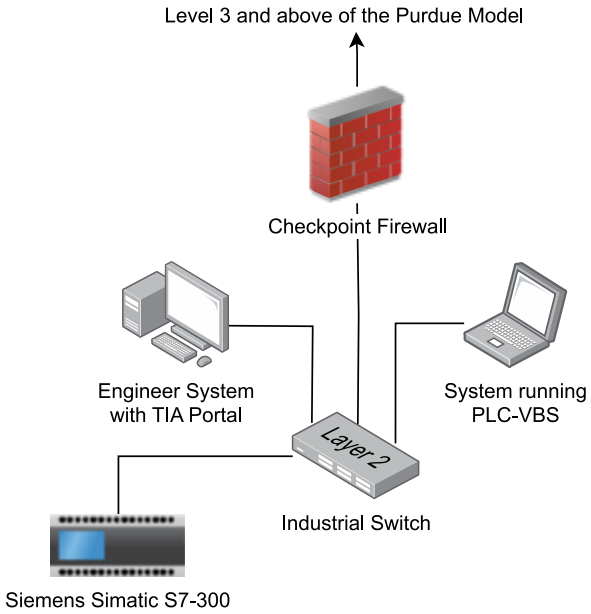


Fig. 7.6 PLC-VBS Evaluation Setup Within the ICS Lab

- Defaults
- Direct Inputs

As each FB dictates the type of input it accepts, it was only possible to use pointers where they were specifically requested (hence their exclusion here). This means if an FB requests a pointer as an input, no other input type can be applied, preventing us from assessing alternatives. Some fVB variables did not include default values, so where possible we left these empty (optional inputs), and some fVB variables did not allow for direct inputs. Where direct inputs were not permitted, we used gVBs. We ran our tool once per configuration setup, and logged the findings before moving on to the next FB.

We have deployed control logic based on these library functions through TIA Portal on a Siemens S7-300 series PLC that is deployed within our Industrial Control Systems Laboratory. The setup of the environment is presented in earlier work (Green et al., 2020). We are connected to the PLC by a system that is running TIA Portal and a secondary system that is connected to a switch within the industrial environment. This secondary device is used to run the scanner, nonetheless the scanner can be ran on any device that is connected to the network in which the PLC resides. A high-level view of the experimental setup can be seen in Figure 7.6.

Function	Bytes	Variables	Defaults	Direct Input	gVBs
MODBUSPN	1544	82	1484	1444	1470
AS_MAIL	990	42	977	889	889
CTU	9	10	6	4	4
CTD	9	10	6	4	4
TP	23	11	4	0	0
WRREC	26	12	20	4	4
TSEND	22	14	20	8	8
PUT	598	106	594	572	572
TCON	20	13	18	6	6
PORT_CONFIG	86	39	84	63	39
TOTAL	3327	339	3213	2994	2990

Table 7.1 Overview of Tested Library Functions - Direct Read

Results

We ran our scanner across ten fVBs. As per our method, each FB was configured three times, with associated results shown in Tables 7.1 and 7.2 (i.e., defaults, direct inputs, and gVBs).

High-level results for direct reads (checking if the bit state inversions have been applied immediately after the inversion request was sent) can be found in Table 7.1. High-level results for delayed reads (checking if the bit state inversions have been retained after the 5 second Read 2 Time has elapsed) can be found in Table 7.2. We can see that for each of the three configuration options, the quantity of vulnerable bytes differs. We can also see there is little difference between direct and delayed reads, aside from the MODBUSPN function. This means, in a holistic sense, that we can manipulate the behavior of FBs via their dedicated memory (fVBs). While our experiment was limited to inverting each bit within a byte, it highlights vulnerable areas of memory that an attack could target in a more elaborate way.

Overall, based on the FBs selected as part of our experiment, we can see that the use of direct inputs and gVBs does reduce the number of vulnerable bytes by around 6%, though this is not a significant difference, as 90% of bytes remain vulnerable. However, what is important to note is that for specific functions (GENERATE_PULSE, WRITE_DATA, and REMOTE_CONN) a significant drop in vulnerable bytes was observed when not using default values. The PORT_CONFIG_DB function also saw a notable reduction from 97.7% vulnerable bytes with default values, to 73.3% vulnerable bytes with direct inputs, and 45.3% vulnerable bytes with gVB inputs. Excluding the use of default values can therefore help in reducing vulnerabilities in control logic; this is a PLC programming practice that should be applied as part of baseline defensive measures.

Function	Bytes	Variables	Defaults	Direct Inputs	gVBs
MODBUSPN	1544	82	1482	1439	1465
AS_MAIL	990	42	977	889	889
CTU	9	10	6	4	4
CTD	9	10	6	4	4
TP	23	11	4	0	0
WRREC	26	12	20	4	4
TSEND	22	14	20	8	8
PUT	598	106	594	572	572
TCON	20	13	18	6	6
PORT_CONFIG	86	39	84	63	39
TOTAL	3327	339	3211	2989	2991

Table 7.2 Overview of Tested Library Functions - 5 Second Delayed Read

Taking the COUNT_UP FB as an example the output for one scan can be seen in Listing 7.7), when we examine the scan results in more depth (see Table 7.3), we can evaluate what variables the vulnerable bits and bytes are mapped to, and what impact this could have on its operations should they be targeted by an attacker. Within COUNT_UP several critical variables can be manipulated across the three configuration types, including CU (counter input), which is used as a trigger for the counter to count up, and CV (count value), which is the current count value. Only the PV variable changes from vulnerable to non-vulnerable when a direct input or gVB input is used compared to default values. An adversary could use these vulnerable variables to stop the control logic from counting, or modify the overall count value with a single request (S7Comms Write). An attack of this nature could prevent the control logic from performing additional critical operations, such as notifying an engineer with an alarm when a given count limit has been reached. Further to this, even though Q (counter output) is not vulnerable, an adversary could manipulate this boolean variable by manipulating the input bytes which are vulnerable. Fundamentally, once the scanner has returned its results, an analysis similar to this must be undertaken by the user to evaluate the list of vulnerable bits and bytes, and identify the impact of their manipulation. This is a critical task, where results are fed into risk assessment processes, with a goal of bridging the vulnerability-impact gap.

During our experimentation we established that pointer-based inputs are less susceptible to attack (e.g., MODBUSPN where variables storing pointer addresses could not be inverted). However, this does not mean the variable is not indirectly vulnerable, as it could point to a vulnerable area of memory. We have undertaken a proof-of-concept attack to examine this possibility further (see Figure 7.9), by first reading the pointer address (a gVB address)

```
==================================

XXXXXXX_ Byte 0 | 7 bit(s) vulnerable
XXXXXXXX Byte 1 | 8 bit(s) vulnerable
XXXXXXXX Byte 2 | 8 bit(s) vulnerable
XXXXXXXX Byte 3 | 8 bit(s) vulnerable
________ Byte 4 | 0 bit(s) vulnerable
________ Byte 5 | 0 bit(s) vulnerable
XXXXXXXX Byte 6 | 8 bit(s) vulnerable
XXXXXXXX Byte 7 | 8 bit(s) vulnerable

==================================
```

Fig. 7.7 Scanner Output for the COUNT_UP Function with Default Values

Variable	Vuln.	Type	Description
CU	✓	Bool	Counter input
R	✓	Bool	Reset
PV	(✓)	Int	Max count before Q is triggered
Q		Bool	Indicates if CV is greater than PV
CV	✓	Int	Count value

Table 7.3 COUNT_UP Function Variables

stored in a fVB linked to the REG_KEY variable (this stores a registration key for use by the MODBUSPN function). Once the gVB address has been obtained, we are able to target it and successfully invert the state of its bits. This bit inversion was then observed by the fVB.

Our scanner was able to identify these previously unknown vulnerabilities within the Siemens TIA libraries, which could enable an attacker to manipulate fVB data and therefore impact FB behavior. This could create wide-spread impact to physical operational processes, communication and alerting functions, and more.

We also ran Nessus and OpenVAS on the same PLC used for this evaluation. These tools identified vulnerabilities such as CVE-1999-0517 (related to SNMP); however, they did not offer insight into any control logic derived vulnerabilities, or the targeted manipulation of control logic behaviour. Additionally, the use of Nessus did cause our PLC to crash during one scan. The results provided by both tools offered basic high-level generic vulnerability information, thus failing to bridge the vulnerability-impact gap in the same way as our approach.

In addition to these tests we have done a preliminary evaluation on a Siemens S7-1200 PLC running the PLC-VBS tool, which showed the tool can be used on some other devices without modification and the S7-1200 PLC library functions have similar vulnerabilities.

7.4.4 Example Attack Scenario

Based on our proof of concept, performed on Siemens library functions, there are several attack vectors based on our identified control logic vulnerability that could be identified. As we are sending unauthorised command messages to the PLC in order to change the state of the memory and impact control logic, we can change the behaviour of the PLC. What can be changed depends on the function and how the variables are written to it. Looking at four library functions (COUNT_UP, AS_MAIL, GENERATE_PULSE and TCON), we can explore one of the potential attack scenarios.

If we have a program that utilises the COUNT_UP function to count (CV) how many times a valve has opened (CU) within a specific period which is reset when the GENERATE_PULSE function sends a pulse to the COUNT_UP function (R). When the counter reaches 10 (PV), a pulse (Q) is sent to the AS_MAIL and TCON functions. The AS_MAIL function, within this example, is configured to send an email to an engineer and the operations centre, and the TCON function writes data to a web server over TCP. An attacker would only need to interfere with the operation of the COUNT_UP function to interfere with the operation of the other two functions and by proxy the operation of the PLC and the associated alarm reporting processes. By continuously writing a FALSE value to the CU variable, the function might never trigger the counter input; therefore, the count value (CV) will not increase when it should. Given that the CV variable is also vulnerable, the attacker could continuously write 0 to this value so it would never reach the max count (PV) value that is required for the function to send a pulse to AS_MAIL and TCON. An attacker could also repeatedly write a TRUE value to the reset (R) variable, which would reset the count value to 0. Finally, the attacker could also interfere with the max count (PV) value if this value was left as default (0), which we deem to be unlikely as an engineer would set this value to enable the function to perform. An attack like this could allow other attacks to be executed without alerting the OT engineer, as reporting processes would be impacted and could eventually result in the activation of safety systems which could cause the plant to shut down.

However, an adversary with access to the PLC could adversely impact the function in multiple ways by exploiting this vulnerability. All three other functions have vulnerable bytes, which can also be used to affect the operation of this example program. For example, the BUSY variable for the TCON and AS MAIL function is vulnerable; this variable is used to identify if the function is busy, and if so, a new job cannot be started. If an attacker overwrites the BUSY variable with a TRUE value, the PLC will halt the function's operation. Some of the vulnerable bytes can become non-vulnerable if direct inputs or gVB is used to provide values. Therefore, good programming practice might mitigate some of the attack vectors.

7.4.5 Summary

As observed from the results of our experiment, we can see that there are ways for adversaries to manipulate PLC operations. If PLC memory can be manipulated via the use of unauthorised command messages, malicious actors are able to change the behaviour of control logic, leading to undesirable operational impact. From scanning 10 distinct library fVBs, we concluded that even vendor provided control logic can be vulnerable. This therefore raises further concerns; more specifically, if vendor provided library functions cannot be considered secure against memory manipulation attacks, what level of vulnerabilities could be exploited within code written by PLC engineers, who are less likely to follow secure programming practices? Not only would this code rely on vulnerable vendor written libraries, but also on self-written sections of code.

To mitigate a number of the identified vulnerable bytes discussed, we have identified that input parameters should always be set (not left as default), as this results in fewer vulnerable bytes. When left unset (default) these values are only set when the PLC initializes, which makes it easy for the values to be overwritten with persistence. The use of concepts such as 'stack canaries' and other similar alerts could also be adopted to monitor predictable data structures indicating the presence of unauthorized command messages targeting vulnerable memory areas. However, at a fundamental level, the control logic itself needs to be reviewed to better understand why certain areas of memory and their associated variables are vulnerable to begin with, and whether modifications can be made to mitigate the risk within the baseline code itself.

However, none of these mitigations can provide complete certainty that the memory vulnerability cannot be exploited. Therefore, it is important that systems are used to identify when there are adversaries within the network that exploit this. Honeypots deployed within the high-interactive honeynet can provide indications if adversaries are using this vulnerability, however, bespoke honeypots that allow memory manipulation can be deployed within the operational network as well. These honeypot can be low-interaction as they only have to be able to correctly represent the memory of a PLC and recognise the required protocol commands. Then the honeypot can be configured to notify the security operations centre when it identifies the memory has changed from the initial state. In a brief experiment done on a real PLC, we have been able to generate a hash of the PLC memory and continuously verifying if the memory of the PLC has been changed. When we subsequently change the values in the memory this could be identified by comparing the baseline hash with the current hash. This is not suitable to do an an operational system which is why honeypots are valuable in this instance. Even when a honeypot is configured to change dynamically, this change should not be unexpected and if the honeypot cycles through expected values all the hashes

can be incorporated as a baseline to compare with. The experiment shows that the honeypot is capable of monitoring for changes within the PLC memory and can alert the relevant personnel when changes occur.

7.4.6 Vendor Response

Siemens response to our vulnerability disclosure: "The flat addressing in PLCs like S7-300 and ET200S CPU is a design decision from the 90s and cannot be easily changed without breaking existing installations. Siemens recommends customers to restrict network access to any affected devices, to apply Defense-in-Depth measures that can be found in the Operational Guidelines for Industrial Security (https://www.siemens.com/cert/operational-guidelines-industrial-security), and to follow the recommendations in the product manuals. Siemens improved this behavior in the new PLC generation (S7-1200 and S7-1500) by creating the optimized Data Blocks and additional levels of protection to these PLCs."

Although the behavior of the new generation of PLCs is improved, as stated earlier, we have been able to find vulnerable library functions within the S7-1200 generation of PLCs. The use of the new security measures introduced in the new generation is not widespread and can interfere with the operation of legacy environments. This requirement for backwards compatibility is one of the difficulties that need to be overcome to effectively secure these systems.

7.5 Conclusion

In this Chapter, we have introduced PLC-VBS, a scanner that allows engineers to identify vulnerable bytes within PLC memory, and implemented a honeypot that can be used to identify exploitation of this vulnerability. We have shown that there can be many vulnerable bits and bytes within PLC memory, by evaluating vendor-provided PLC library functions, through which adversaries can interact with the device and its associated underlying operational processes. During the evaluation, we tested 10 vendor-provided library functions and identified that they are vulnerable to manipulation through the use of unauthorised command message attacks. This proves that anyone able to access the PLC over the network can change values within memory reserved for commonly used functions. This could result in a disastrous impact and allow adversaries to interact with the PLC while hiding alerts from system operators. For example, suppose operators are to be notified when a counter reaches a certain value. In that case, an adversary could keep resetting this to a lower value, and therefore prevent alerts from being generated, while also impacting other areas of control

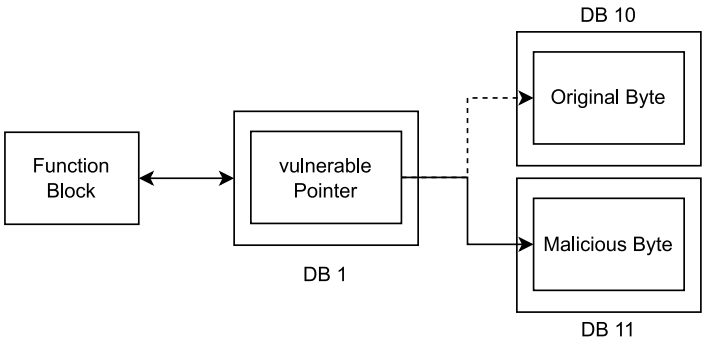


Fig. 7.8 Pointer Re-direction Example

logic that may require this value to operate (e.g., to start a pump when the count reaches a given threshold).

To begin mitigating the vulnerabilities discussed within this Chapter we would encourage PLC engineers to avoid reliance on default or hard coded (Serhane et al., 2018) input values and public libraries. That would reduce the attack surface but not necessarily mitigate it completely, as we have seen direct input and gVB values can still be vulnerable. This could form part of a DevSecOps approach, where security is an inherent part of the development and operational process. In addition, we advise the protection of VBs from direct manipulation over the network where this level of connectivity is not required for operational purposes (e.g., HMI access). While we have shown that there is a multitude of vulnerable bytes within the tested fVBs, the potential misuse of these is not necessarily avoided when a specific byte is not vulnerable. We can also identify pointer addresses within functions that are potentially vulnerable to being overwritten, which would then point to an address of the adversary's choosing, i.e., performing a pointer re-direction. This can be seen in Figure 7.8 where we show a function with a pointer in DB1 that used to point to an address in DB 10 but is now redirected to point to an address in DB 11.

Further to this, if the pointer itself cannot be overwritten, the address it is pointing to might be vulnerable. Hence, the pointer can be viewed as vulnerable by proxy. Furthermore, if we can identify multiple pointer addresses across VBs pointing towards the same gVB, we can establish the "primary gVB"/gVB of high importance across the control logic base. We can then investigate this VB (e.g., DB 10 in Figure 7.9) and monitor changes across multiple other VBs (e.g. DB 1, DB 2, DB 3, and DB 4) potentially interfering with previously non-vulnerable addresses. However, it is important to note that when we refer to pointers, we only have the pointer start address and would need to establish the length of data stored at that address. An example of this can be seen in Figure 7.9 where we depict a function that stores values in DB 1 and has a pointer that points to a value stored in DB 10.

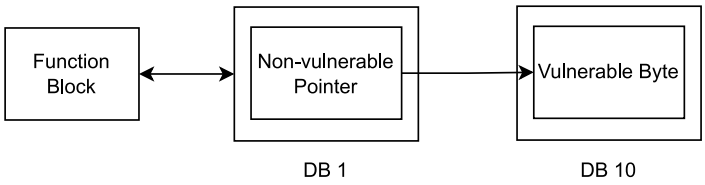


Fig. 7.9 Indirect Vulnerable Pointer Example

Other appropriate mitigation techniques, such as network-based detection and honeypots (Maesschalck et al., 2022), have to be deployed to deliver a defence-in-depth strategy. Given that the vendor-provided library functions are vulnerable and all PLC code is built upon these and bytes can still be vulnerable by proxy, it is not possible to mitigate this with proper programming practices. Therefore, other measures have to be implemented. For example, honeypots could be used to identify if adversaries are actively exploiting the identified vulnerabilities when configured with appropriate levels of interactivity (Maesschalck et al., 2021). We have evaluated this by deploying a honeypot that continuously scans for changes in its memory. Whereas we have deployed a real PLC as a honeypot, this can also be done by developing a low-interaction PLC honeypot that is capable of accurately emulating PLC memory and the desired communication protocol, such as S7Comm. This approach is important to consider as many bespoke vulnerabilities are inherent to the operation of the system and can therefore not be easily mitigated or would require significant investment and changes within the operational environment which is not always feasible. The goal of the usage of honeypots is to provide a level of risk reduction and impact mitigation until the issue itself can be fixed.

Although this Chapter focused mainly on one particular vulnerability we have identified within PLCs, many more of the vulnerabilities that cannot be easily mitigated are out there. The role of honeypots might not initially have been clear, but we hope that it is apparent how the nature of honeypots as a form of active and deceptive cyber defence can be beneficial when dealing with these vulnerabilities. This also links back to the goal of honeypots and where they are deployed according to HoneyPlant. Honeypots with the aim of detecting active exploitation of vulnerabilities within the network can be very beneficial when deployed inside the organisational environment. However, they must be deployed and configured with this purpose in mind.

Chapter 8

Passively Identifying Honeypots

In the previous Chapter we saw that ICSs are specialised devices and therefore have bespoke vulnerabilities. For these vulnerabilities we deployed bespoke honeypots, but just like other honeypots these can also be identified by the threat actor. In Chapter 2 we discussed threat actors in the area of critical infrastructure and nation-states in particular. As nation-state actors are generally more knowledgeable and take increased precautions to avoid honeypots and other security measures that are easy to identify are not generally effective against them. Within Chapter 5 we have discussed how honeypots can be fingerprinted and investigated how many Conpot honeypots can easily be found through Internet-scanners. In this Chapter we take this work a step further and investigate more in-depth how we can identify honeypots based on Internet-scanner data. Because this data is not gathered from the system itself this form of fingerprinting is passive and presents with many opportunities for both adversaries and researchers. The work in this Chapter also builds upon Chapter 6 as any honeypots deployed according to the framework should be evaluated for honeypot characteristics to ensure maximal effectiveness.

8.1 Introduction

While attackers previously had to obtain physical access, to compromise critical infrastructures they can now execute attacks remotely. Furthermore, IP scanners and search engines, such as Shodan, are used by adversaries to provide them with a list of Internet-accessible ICSs.

To defend against and detect attacks, ICS operators often deploy Honeypots that resemble ICS devices and can misdirect attackers toward them instead of actual ICS devices. As noted before, honeypots are designed to mimic several ICS devices by simulating the corresponding protocols, including S7Comm, BACnet and CIP. We already established that there are many

benefits to deploying ICS honeypots, such as increased alignment with regulation and sharing threat intelligence across the sector Maesschalck et al. (2022). However, these honeypots can also be identified by external actors, including researchers and adversaries.

To understand and assess the threat of cyber attacks against ICSs, researchers and security engineers have been measuring the ICS ecosystem to understand and map potential vulnerabilities, mainly relying on IP scanners such as Shodan (Alsmadi et al., 2022; Bodenheim et al., 2014; Chen et al., 2020; Tundis et al., 2021; Wang et al., 2017). Such measurements can inform the work of regulators and governmental agencies who are responsible for authoring policies and best practices in securing ICS networks (Centre, 2020; European Network and Information Security Agency (ENISA), 2011; Stouffer et al., 2011). However, the presence of honeypots can introduce artefacts in the measurements and lead to misleading conclusions about the state of ICS security, as deliberately vulnerable honeypots may be perceived as actual ICS devices. Most studies attempt to filter honeypots from the list of scanned IPs by utilising Honeyscore (Shodan, n.d.a), a metric provided by Shodan to indicate the probability that an IP corresponds to a honeypot. Nonetheless, Honeyscore has been shown to miss a lot of honeypots (López-Morales et al., 2020; Zhou, 2019), and its development has been discontinued[1].

In this Chapter, we evaluate the trustworthiness of Shodan in regard to the ICS classification. We aim to help the research community and policymakers obtain a more accurate view of ICS exposed on the Internet and the actual attack surface that adversaries can exploit. Highlight potentially misconfigured honeypots which skilled attackers can easily detect and avoid. And provide researchers with more accurate passive metrics to evaluate more advanced honeypot configurations and deployments since Honeyscore, which is the current state of practice, fails to uncover a large fraction of possible honeypots. To achieve this, we have developed a short algorithm that utilises Internet scanner data which we ran on the dataset obtained from Shodan. We then evaluate the accuracy of the algorithm with active scanning based on previous work (Zamiri-Gourabi et al., 2019). Additionally, we use the outcome of our classification algorithm to provide an evaluation of Shodan Honeyscore, which is developed specifically for ICS honeypots.

8.2 Fingerprinting ICS Honeypots

When looking for honeypots, it is essential to focus on the type of system investigated. Because honeypots are designed to portray themselves as the system attackers are looking for. Therefore, we have to ask ourselves how we would approach fingerprinting a honeypot.

[1]According to an update by Shodan's official Twitter account (Official Shodan Twitter account, 2020)

In this chapter we are focusing on PLCs, which are a subset of ICSs and used to control physical devices (Erickson, 1996).

We must first consider which type of honeypots we are looking for. For low-interaction honeypots, it is much more straightforward to consider how we can identify these. First of all, several low-interaction honeypots are available online. For ICS, Conpot is one widely used low-interaction honeypot (Maesschalck et al., 2022). However, many honeypots deployed based on these pre-made solutions tend to include default signatures which can be used to determine that a system is a honeypot (Maesschalck et al., 2021; Zamiri-Gourabi et al., 2019). Because of this, it is crucial to consider signatures of honeypots that are available for the type of system investigated, in terms of Conpot that includes signatures such as 'Technodrome'. Other ways of fingerprinting low-interaction honeypots include active fingerprinting such as interacting with the system and evaluating the services running on the system (Gallenstein, 2017). These honeypots also run an operating system which can be used as an indication. For example, a low-interaction honeypot can be run on a Linux distribution which can be identified. We have seen that NMAP has an issue identifying which operating system is running on a real PLC (Maesschalck et al., 2021). For high-interaction honeypots fingerprinting would include evaluating the services running on the system.

It is important to investigate where honeypots are deployed and if this aligns with where an actual system would be deployed. For example, a PLC would not generally be found hosted in the cloud because of their time-critical and safety nature (BR Automation, n.d.). If a PLC is found running on one of the networks of a cloud provider, such as AWS, it might be a honeypot (Maesschalck et al., 2022). Further, the environment the system is deployed within is also important. Finding a single system that is not connected to other systems or discovering a network of multiple instances of the same system could also indicate a honeypot (Antonioli et al., 2016). Additionally, the protocols (NXLog, n.d.) deployed on the system are essential to consider. A system running multiple industrial protocols (e.g. BACnet, S7, MODBUS) would also be not in-line with the expected systems. For example, a PLC manufactured by Siemens would generally run the proprietary S7 protocol as this opens up more functionality, and an Allan-Bradley PLC would run MODBUS. An industrial system would also rarely be seen running consumer-grade operating systems or as they run customised operating systems for real-time control and robustness Sehr et al. (2020). There are other ways of fingerprinting honeypots that could be used. However, the above are the main ones Therefore, we have to ask ourselves what we focus on within this Chapter.

8.2.1 Shodan Honeyscore

Classification or tagging of devices within Shodan happens when a device gets scanned over a specific protocol. Regarding honeypots, Shodan uses the Honeyscore system, which assigns a value between 0.0 and 1.0, to determine if a system is likely to be a honeypot (Shodan, n.d.a). Researchers also use Honeyscore as a prime indicator to evaluate their honeypot deployments and by researchers to identify honeypots (Bistarelli et al., 2021; López-Morales et al., 2020; Wang et al., 2017; Zhou, 2019). However, classification could be better as low-interaction honeypots are sometimes not flagged as a honeypot (Maesschalck et al., 2021) and sometimes they are also classified as real devices.

Other Internet scanners that compete with Shodan include Censys, Zoomeye and Fofa.

8.3 ICS Devices on Shodan

To obtain information on ICSs available on the Internet, we are leveraging Shodan and their ICS classification. Ulisiling the API, we can query Shodan for all the hosts it identified as ICSs on the Internet by utilising the 'ICS' tag filter. From the data obtained by Shodan's interaction with the host, we store the IP address, open ports, operating systems identified on each port, hostname, domain, organisation, autonomous system number (ASN) and the Shodan Honeyscore.

Our initial search query returned 110,863 results which we then scanned individually. Due to the changes in devices on the Internet and IP addresses, we obtained information on 90,682 different systems from our initial dataset (Table 8.1). We had to remove 20,181 systems from our initial query as these systems did not exist anymore or non-ICS systems had taken the IP address. We only used IPs that remained stable over the three weeks. The fact that these IPs are reallocated to other devices might be an indication that these systems are not actual industrial devices. For this reason, and to have a trustworthy dataset. Leveraging the information provided by Shodan (Shodan, n.d.b) on their classification, ICSs include SCADA, PLC and DCS running protocols such as Modbus, S7Comm and DNP3. Previous work (Maesschalck et al., 2021) has also concluded CIP is one important protocol used by Shodan for ICS discovery.

Regarding the open ports, 3,013 systems have port 102, which is related to Siemens S7Comm protocol. One of the most popular ICS ports in the dataset is 502, which is the port used by Modbus and is open on 21,893 systems. The discrepancy with S7Comm can be explained because, unlike S7, Modbus is not a proprietary protocol and is popular for building management systems and used in some smart home systems. Discovering ICSs that are used in critical infrastructure sectors is less likely, given the tight regulation of these environments

ICS Tag	110863	Dataset	90682
Port 102	3013	Port 502	21893
Port 44818	5434	Port 2222	1284
Port 47808	12355		
Cable/DSL/ISP	38928	NSP	19630
Educational/Research	1172	Content	1095
Enterprise	155	Non-Profit	90
Network Services	38	Route Services	4
No ASN value	29570		

Table 8.1 ICS Devices on Shodan

in many parts of the world. The ports related to the Common Industrial Protocol (CIP) using Ethernet/IP are also popular within our data, with port 44,818 open on 5,434 systems and port 2,222 open on 1,284 systems. BACnet, another popular OT protocol, runs on 12,355 systems within the dataset. This protocol is mainly used for building management systems.

Examining the autonomous system (AS) linked to the IP addresses we have investigated through the usage of the PeerDB dataset, we can see 8 AS types. The most popular classification of these AS is Cable/DSL/ISP, linked to 38,928 systems, followed by NSP with 19,630 systems. The other classifications are less popular, with Educational/Research being linked to 1,172 systems, Content to 1,095, Enterprise to 155, Non-Profit to 90, Network Services to 38 and Route Services to 4. For 29,570 systems, we could not find an AS classification. At first sight, we can identify several of these networks as unlikely to deploy actual industrial control systems, e.g. it would be improbable to see these systems on a content provider or Route Services network. A PLC deployed in the cloud would be a red flag to adversaries (Maesschalck et al., 2022).

We can see that some of these IP addresses are behind NAT as we can identify operating systems such as ASUSWRT, MikroTik RouterOS and Synology Router Manager running on several systems. Debian and Windows (including Windows Server) are also seen in the dataset. Some of these systems might not be industrial networks, and the systems running on these networks could be (smart) home systems.

We have identified that Shodan consistently only uses 0, 0.3 and 1 within their Honeyscore, with only 3 (0.00004%) out of 84319 systems gaining a score of 0.5. We have yet to see any other scores. Out of the systems we have obtained from Shodan, 66,762 (79.2%) have a score of 0, 17,522 (20.1%)) have a score of 0.3, and 31 (0.004%) have a score of 1. 6,364 systems did not have a Honeyscore, which bring us to the total of 90,682 systems in our dataset.

8.4 Passive Classification Algorithm

Opposed to machine learning (ML), we provide a novel passive scanning algorithm to classify ICS honeypots based on a set of predefined heuristics which are shown in Figure 8.1. We have opted not to use ML as at the time of research there were no available datasets containing the information required for this experiment. The usage of ML will be important for future work, which this research can hopefully present a basis for.

The proposed algorithm aims to indicate how likely a system identified as an ICS by Shodan is an actual ICS and not a honeypot/non-ICS system. Our algorithm is designed to provide this classification by not interacting with the system itself but by relying on the characteristics of those systems, such as banners and protocols. In this evaluation, we obtained this data through Shodan, but the data can be obtained through other scanners or custom scans. To identify honeypots, we have assigned a score to multiple characteristics that do not match the deployment of a real ICS device. These range from the operating systems used to the device's network. As a result, our classification algorithm can identify a non-honeypot system as a honeypot when it is not a real ICS device. However, we deem this acceptable as these would also be systems to avoid interacting with. In addition, the use of NAT can also interfere with the possibility of providing an accurate classification if there are many hosts connected to the same IP.

The data used within our algorithm stems from the same data we obtained from Shodan earlier. We further use PeeringDB data to classify the ASN classification type and use this as an indicator within our algorithm. To obtain as much data as possible from Shodan to use within our algorithm, we also search for common signatures of Conpot honeypots within the data returned by the API. As these honeypots are widely deployed and usually in their default configuration Maesschalck et al. (2021), we deem them important to use within the classification.

8.4.1 Methodology

Our passive scanning algorithm focuses on several data sources to enable it to analyse a system. The three main ones are the number of ports running on a system, the operating systems identified, and the ASN connected to the system's IP address (Figure 8.1). Based on these three data points, the algorithm determines the likelihood of whether a system is a honeypot or not. It classifies a system as either a honeypot, likely honeypot, potential honeypot, or unlikely honeypot. Given that this is a passive algorithm, the classifications provide an indication, and there is potential for crossover between bordering classifications such as unlikely and potential honeypot. First we identify the default Conpot signatures

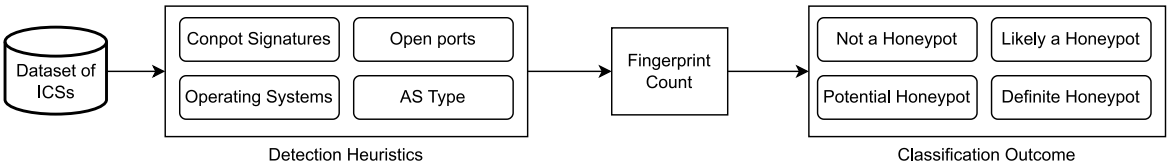


Fig. 8.1 Passive Scanning Honeypot Classification Methodology

(Zamiri-Gourabi et al., 2019) such as 'Technodrome' and 'Mouser Factory', and for each of these signatures we provide a fingerprint score ranging from 2 to 8. We also increment the fingerprint counter if the amount of open ports is larger than 3 by 1 or by ports/2 if it or larger than 4. We have chosen these values as ICS deployments are often deployed on field sites, which contain a limited amount of systems that are reachable from outside the internal network (Gardiner et al., 2019; Gonzalez et al., 2019). Afterwards, the algorithm checks the operating system running on the system. If one of the operating systems is classified as not None or a Linux distribution, the fingerprint counter gets incremented. This is because many field instruments usually run non-commercial operating systems (Weiss, 2014), although some devices such as HMIs can run Windows variants (Cárdenas et al., 2008; Matsuda et al., 2019). Finally, depending on the ASN identified, the system can receive an increment in the fingerprint score. For example, an Educational/Research ASN would increase the fingerprint score with 3 as real industrial systems would generally not be found within educational settings (Maesschalck et al., 2022). Based on this information, the algorithm classifies the system as either a honeypot, likely honeypot, potential honeypot, or not a honeypot.

8.4.2 Results

Running our passive scanning algorithm on the more than 90,000 systems within our obtained dataset, the algorithm classified most systems as not being a honeypot or a potential one. Out of all systems, 56,759 were classed as not being a honeypot and 28,373 potentially being one. Combined, they form 94.5% of our dataset. The classification with the lowest amount of systems is the definite honeypot classification, which consists of 1,409 systems. The remaining 4,141 systems are likely honeypots, as our algorithm missed no systems.

Amongst the systems that were classed as definite honeypots, we see a range of values for the fingerprint count, a value used by the algorithm to determine the amount and weight of honeypot indicators a system shows. Their values range from 10.5 to 455.5, which shows some systems are clearly honeypots or not actual ICS systems. There are 68 systems with a value of 10.5, which is the lowest value possible within the classification; these systems are, therefore, close to being classified as likely being a honeypot. Because we focus on industrial systems within an industrial network, certain characteristics that are used are related to the

expected environment in a system like this would be deployed. This means that systems that show a high amount of honeypot characteristics for being an ICS honeypot could be real systems, but likely not actual industrial systems. In addition, for example, systems deployed on networks linked with educational institutions might be ICSs. However, they would not constitute the ICSs deployed within operational technology environments within the industry we are looking for. Therefore, it is highly likely that some of these systems are not honeypots. Of the systems identified as honeypots, 93 show signatures of being a Conpot honeypot, and 990 received a Honeyscore of 0.3 or higher. We see 1,572 systems with a Honeyscore of 0.3 or higher for the likely honeypot classification. Looking at the fingerprint count, 130 systems are at the upper bound of the classification, and 263 are at the lower bound. Therefore, these systems are very close to the definite and potential honeypot classifications.

Out of the systems that are classified as potential honeypots, we see 6 systems with a Honeyscore of 0.5 or higher, 5,671 with a Honeyscore of 0.3 and 20,982 with a score of 0. For the lowest possible fingerprint count, we have 11,932 systems; these systems show some signs of being a honeypot. Again, we must caveat that these systems might not be industrial. Looking at the upper bound, we have 1,372 systems, meaning these systems show a moderate amount of honeypot characteristics and are close to being classified as likely a honeypot. Finally, for the non-honeypot systems, we see 21,163 systems with a fingerprint count of 0 which means they do not show honeypot characteristics according to our algorithm. Of these systems with the lowest possible count, 3,907 have a Honeyscore of 0.3, no systems have a Honeyscore higher than 0.3, and 15,517 have a score of 0. When looking at the upper bound of the fingerprinting count of this classification, 3,509 systems have the highest count. Of these systems, 825 have a Honeyscore of 0.3 and 2,486 have a score of 0.

8.4.3 Evaluation with Active Scanning

To evaluate our passive scanning algorithm, we have performed an active scan of a subset of systems identified by Shodan. As active scanning is the most effective way to identify if a remote system is a honeypot, although 100% certainty cannot be reached on systems you do not own. For this, we randomly selected 327 systems and evaluated them. When we map the classification from the active scan to the classification given by our passive scanning algorithm, we can better determine the accuracy. It is, however, essential to note that the active scanning algorithm might not be 100% accurate, and there might be misclassifications.

For the active scanning, we have based ourselves on previous work by (Zamiri-Gourabi et al., 2019) and (Sun et al., 2020) without the machine learning, adapted to our specific needs. This scan involved scanning ICS-specific ports by obtaining information about the system to verify the system is a real ICSs. These probes are designed to be as non-intrusive as possible

by testing them on our operational ICS testbed. We use S7Comm packets using Snap7 using the get_cpu_info(), connect(), get_connected(), and get_plc_datetime() commands. These query the PLC for basic system information, which we only used during run-time to verify the response and did not store. Honeypots that do not incorporate an in-depth implementation of S7Comm will respond differently than real PLCs. For example, a Conpot honeypot would have issues responding to these requests and throw errors (Maesschalck et al., 2021).

First, we compare the outcome of the active scanning with the Shodan Honeyscore. Out of all the systems classified, there were no systems we found to be definite honeypots. However, one system received a Honeyscore of 1, which active scanning deemed not a honeypot. Further investigation into this system leads us to believe this is a real ICSs, as it passed all our active scanning tests. This appears to be a discrepancy on Shodan's behalf. Out of the systems that received a Honeyscore of 0.3, we see systems that our active scanning has identified as potential, likely and not a honeypot. This further blurs the meaning of the 0.3 classification. Like before, 5 systems that passed all our active scanning tests received this 0.3 classification. 10 systems that only passed 1 of our tests and we classified as likely being a honeypot received a Honeyscore of 0.3. This means that Shodan failed to give a Honeyscore of more than 0 to 76% of the systems we identified as potential or likely honeypots. We see a similar spread of systems we classify as not, potential and likely being a honeypot receiving a score of 0. This included 152 systems that did not pass any of our tests and 30 that only passed one test. These results question the reliability of the Honeyscore.

Looking at a direct comparison between the scanning techniques, we see that our passive scanning classifies 28% of the systems in line with the active scanning. This is not surprising, as active scanning should always receive significantly better results, given the very intrusive nature of the scan. However, when we allow for crossover between the classifications (e.g. allowing a definite honeypot classification by the active scan result in both a definite and likely honeypot classification by the passive algorithm) to recognise the potential crossover between classifications and to view the passive scanning as a quick and unintrusive first opinion of the system, we achieve an accuracy of 71.96%. An overview of the outcome of the passive scanning vs the active scanning can be seen in Table 8.2. Suppose we focus on the second value, where we allow for crossover between the classifications. In that case, the passive algorithm obtained a relatively good accuracy compared to active scanning. Active scanning is more intrusive and passive scanning could allow for an initial shift between systems that are more likely to be a honeypot. This is a good initial score for a passive scanning algorithm that shows passive scanning is used to indicate the likelihood that a system is a honeypot. If we look at the accuracy, precision and recall of both the passive scanning (based on the crossover) as can be seen in Table 8.3) we can see a good accuracy

		Active Scanning	
		True	False
Passive Scanning	True	13	2
	False	28	64

Table 8.2 Confusion Matrix of Active Scanning and Passive Scanning

Accuracy	71.96%	Precision	86.67%
Recall	31.7%	F1-score	46.43%

Table 8.3 Performance of Passive Scanning

and precision score but a significantly lower recall score. This lower recall score also leads to a lower F1 score. Given this is an initial evaluation of passive scanning for ICS honeypot discovery we deem that more work can be done to increase the recall score and the F1 score at the same time. However, due to the good ratio of true positives, if our passive scanning indicates a system is a honeypot it is highly likely it is. This is very useful as the technique will not trigger alerts on the system, unlike active scanning. This brings many benefits when evaluating a system, as you can obtain a relatively high percentage certainty a system is a honeypot when identified by our passive algorithm.

8.4.4 Comparison with Shodan Honeyscore

Comparing our classification with the Shodan Honeyscore is difficult given the low volume of Honeyscores aside from 0 and 0.3 that were included in our dataset. But important because, as stated earlier, Shodan Honeyscore is often used to evaluate honeypot deployments and is used within the measurement community. However, comparing them allows us to compare our algorithm with the current state-of-the-art from Shodan. Out of the 31 systems Shodan identified as honeypots (Honeyscore of 1), we have identified 25 (81%) of them as definite honeypots, 2 as likely honeypots and 4 as potential honeypots. If we allow for a grouping of definite and likely, we achieve a similarity of 84%. Out of the 3 systems that received a Honeyscore of 0.5, our algorithm identified 2 of them as potential honeypots and 1 as not a honeypot. Given the low amount of systems receiving a Honeyscore of 0.5, we cannot make an adequate comparison.

Looking at the 17,522 systems with a Honeyscore of 0.3, our algorithm identified 965 systems as definite honeypots, with 44 showing signatures of being a Conpot honeypot. 1570 were classified as likely being a honeypot, and 5,671 were classified as a potential honeypot. The remaining 9,316 systems fell under the not a honeypot classification. If

we view this Honeyscore as Shodan believing the system shows honeypot signatures and, therefore, would fall under our potential or likely classification, we achieve a similarity of 41%. If we include the systems we identified as a definite honeypot, this increases to 47%. Out of the 66,762 systems that received a Honeyscore of 0, our algorithm identified 43,024 as not being a honeypot, which equates to 64%. We identified 20,982 systems as potentially being a honeypot and 2,414 likely being one. For definite honeypots, we identified 342. Out of all systems, Shodan gave a Honeyscore of 0, meaning they believe it is not a honeypot and 19 show signatures of being a Conpot honeypot. Shodan failed to assign a Honeyscore to 6,364 systems, but our algorithm did provide them with a classification. We identified 77 as being definite honeypots, 155 as likely being a honeypot, 1,714 as potential honeypots and 4,418 as not a honeypot.

When we compare Shodan's Honeyscore to our algorithm, we can see that many systems fall under a similar classification. However, Shodan seems to miss several systems that show clear signs of being a Conpot honeypot. Because Shodan classified 99% of the systems as 0 or 0.3, we feel that the Honeyscore is not an exceptionally reliable classification source. Several of the systems Shodan tagged as being 'ICS' are not industrial systems, but rather smart home systems or other non-industrial systems discrepancies between our algorithm designed for ICS are to be expected. Overall, the similarity obtained is positive for our algorithm. In addition, our algorithm has a bigger spread along the four classifications and identifies several systems that show signatures Shodan did not pick up on. Given that we incorporate these signatures within our algorithm, we manage to pick up on more Conpot honeypots than Shodan. The algorithm can also become more effective by including more signatures of widely used ICS honeypots.

8.5 Discussion

Looking at the results of our initial scan for ICSs on Shodan, we could see a wide range of systems within the data. Although Shodan tags all systems as ICS systems, many show signs of not being industrial systems. Such as running Ubuntu, Synology software and ASUS router software. This can be explained due to how cheap industrial protocols are to implement and how protocols like WirelessHART have been explored for usage within other devices and systems such as smart cars (Bi et al., 2015) and MODBUS, which has been used in home automation (Hassanpour et al., 2017). Therefore, some of the systems identified by Shodan as ICSs are not actual industrial systems. Papers scanning for ICSs on the Internet have also relied on industrial protocols Sun et al. (2020). However, as seen in a previous study (Feng et al., 2016), proper scrutiny is not always done if those devices are actual industrial devices.

In addition to this, service banner manipulation can be used to mislead scanners Bodenheim et al. (2014).

When we look at our passive scanning algorithm, we can see a positive evaluation with active scanning. Achieving an accuracy of 72% and precision of 87% with active scanning is a significant result, indicating that passive scanning for ICS honeypots is an opportunity for further investigation. However, we did achieve a low recall value. Although active scanning is more accurate than passive scanning, our passive scanning algorithm can be used as a triage point to reduce the number of systems needed to use active scanning on. With honeypots, any interaction is beneficial to the defender; therefore, being able to identify honeypots in a dataset through passive scanning is hugely helpful for an attacker. In return, when deploying a honeypot, it is crucial to reduce as many characteristics that could make an adversary believe the system is a honeypot. Hence, scanning deployed honeypots passively can help evaluate the deployment. Active scanning can also do this, but when an adversary gets to the stage they perform an active scan, the honeypot collects data. Therefore, passive scanning allows for the evaluation of honeypots in a similar way adversaries would and is extremely helpful for researchers to exclude systems that are likely honeypots.

Comparing our algorithm with the Shodan Honeyscore presents us with evident shortcomings within the system used by Shodan. First, several signatures that are part of default Conpot honeypot deployments are not adequately used. Given that this is a very popular ICS honeypot and signatures can be found in hundreds of devices on Shodan and more than 4,000 on ZoonEye (Maesschalck et al., 2021), this should be incorporated within honeypot identification algorithms. Further, more than 6,000 or 7% of systems did not receive a Honeyscore from Shodan for unknown reasons. In addition, given the lack of variety and explanation of what Honeyscore means, it is a relatively unreliable metric. Based on our algorithm, several systems with a score of 0.3 are clearly honeypots, and some are not, leading us to question the metrics behind the scoring. As nearly all systems either received a score of 0 or 0.3, there is uncertainty about the accuracy of the Honeyscore itself. Our algorithm had no issue providing any of the systems with classification, and for the systems with a Honeyscore of 0, the algorithm identified 70.5% of the systems as not a honeypot or a potential honeypot. This does show that we are generally in line with that classification by Shodan. However, it is surprising that the Shodan Honeyscore cannot identify ICS honeypots, mainly because it is designed for them. Even when Shodan classifies a system as a honeypot, the Honeyscore does not always reflect this (Skhomenko, Andrey, 2020). We can also see this when we compared the Honeyscore with our active scanning, clearly showing that systems that failed all our scans received a Honeyscore of 0 and systems that passed all our scans received a Honeyscore of 1.

8.6 Conclusion

This Chapter set out to investigate Shodan and its capability to identify ICS honeypots. For this, we have developed an algorithm that leverages several characteristics that indicate non-industrial deployments, classifying the systems into four categories ranging from not a honeypot to definitely a honeypot. Based on our initial results, passive scanning is a potential mechanism to provide researchers with a way to identify systems with a higher likelihood of being honeypots and can present organisations with an indication that their honeypots are well deployed. As we have obtained a low recall score more work would need to be done to improve this to make passive scanning more effective, but given the strong accuracy and precision scores we deem this shows the possibilities of passive scanning being effective for this purpose. Some active scanning methods have also shown a lower recall compared to the precision scores (Sun et al., 2020). Especially as not triggering alerts on the system is a significant benefit. We further identified that the Shodan Honeyscore in unsuccessful in detecting honeypots and fails to provide a clear definition of their values. These values were also very limited within our dataset, with only 4 distinct values (0, 0.3, 0.5 and 1), of which 99% of systems fall under 0 or 0.3. Therefore, currently, Honeyscore should not be used to evaluate honeypots.

Chapter 9

Obfuscating Systems Through Programmable Honeypot Mimicking

In the previous Chapter we further explored honeypot fingerprinting and found that passive fingerprinting of honeypots is possible. When we combine this conclusion with the outcome found in Chapter 5 we have identified a clear problem. To properly deploy honeypots and obtain the most value out of them we need to make sure they do not have any honeypot characteristics. In Chapter 4 we have seen many honeypots we surveyed have clear indications of being ICS honeypots, such as being deployed in the cloud and maintaining the default Conpot signatures which we have shown can be used in passive fingerprinting. To adequately address all the honeypot characteristics a lot of resources need to be spent. Therefore, we raise the question if we can circumvent that. Within this Chapter we introduce obfuscation as a technique to turn honeypot characteristics into a benefit and use them to protect our system. Doing this we do not need to make out honeypot present itself as a real system, saving resources and still achieving the objective of increased security. This concept can also be used for honeypots deployed according to the HoneyPlant framework introduced in Chapter 6 and also leverages the goal of certain adversaries (e.g. nation states) to remain anonymous in the network and not take any risks to be discovered.

9.1 Introduction

The notion of security by obfuscation within an OT environment, by mimicking honeypot characteristics on operational PLCs is similar to how attackers try to obfuscate their malicious code during a cyber attack (O'Kane et al., 2011). We have previously established that honeypots are a form of deceptive cyber defence and the notion of obfuscation is also part

of this. Obfuscation has been shown to be effective and used as a security measure on a regular basis (Xu et al., 2020). There are often stories about items hidden in plain sight, which we overlook until looking at in more detail (Zerubavel, 2015). By carefully attracting attention to one object, other objects can successfully be hidden away. Other crimes that are not perpetrated in cyberspace, such as human trafficking, have also been hidden and obfuscated (Hepburn and Simon, 2013). Within cyberspace, obfuscation has become popular as well. Services such as Tor or using a VPN are all successful in helping users achieve their goals, ranging from buying drugs online to watching a geoblocked movie on Netflix (Khan et al., 2018; Loshin, 2013). We aim to use obfuscation to protect PLCs better and hide them in plain sight. Our system would focus on hiding the PLC by pretending it is a honeypot, which is similar to what the other uses we discussed and does not change the system itself. This means that we do not want to interfere with any operations of the PLC but rather surround the PLC with honeypot-like characteristics that would make an adversary believe the system is a honeypot. Our proposed system is not designed to make other security systems obsolete but rather to add an additional security measure to the system. It would not stop direct attacks on the PLC if executed by an adversary in the network but aims to convince the adversary not to attack the system.

Due to the critical nature of the devices we are focusing on (Green et al., 2017), the limited resources (Dodson et al., 2020), and the impact on safety they can have, we aim to achieve this with no interruption to the operation. To achieve this in our practical evaluation, we leverage software-defined networking (SDN) to allow us to dynamically route traffic to other devices, isolate devices and virtually move systems within the network. SDN also allows us to monitor all traffic in the network and obtain a broader look at our network activity to identify suspicious traffic such as botnet traffic (Ja'fari et al., 2021). This also aligns with the concept of moving target defence, where the attack surface can change based on current activity (Luo et al., 2019).

9.2 Obfuscating ICS

Within this Chapter, we focus on a component of ICSs, namely PLCs which reside on level 1 of the Purdue Model (as depicted in Figure 2.1). These controllers control other devices within the industrial environment, such as a manufacturing process and output signals that change the state of physical devices such as valves. What we aim to do with these systems is implement a different approach to the use of honeypots within the environment. Rather than deploying honeypots and making them look like a real system, we are making PLCs appear to look like honeypots. This is what we define as obfuscation within this Chapter, obscuring

the PLC by making it pretend to be a honeypot. It can be challenging to deploy a honeypot without any sort of signatures, such as no active connections going to it and no actual data being on the system. Instead of making everything look like a real system, why don't we go the other way and make real things look like honeypots?

The usage of cyber deception techniques have increasingly found more interest and are used for multiple purposes (Bushby, 2019). This can be seen in techniques such as moving target defence (MTD) (Babadi and Doustmohammadi, 2022; Lei et al., 2018) and studies into the effectiveness and timing of deception have also been done (Aggarwal et al., 2016). The goal is to provide plausible but misleading information (Wang and Lu, 2018), which is exactly what a honeypot aims to do as well. However, this can also be used in a different way. By providing implausible information attackers would be less intrigued by a system. This is where the notion of fake honeypots originates.

Fake honeypots have already been described in research (Rowe et al., 2006) and present an interesting approach to both secure the system and circumvent the issues in deploying realistic honeypots. However these have, to our knowledge, not been explored in more depth. These systems do fulfil both goals of defence and threat intelligence. This dual-purpose system has a number of essential characteristics of themselves when looking into deploying. We have to ask what is needed to effectively make an adversary believe the system is a honeypot and also make sure the 'obfuscator' system does not hinder the system itself. Especially within industrial control system environments, this is an important aspect that has to be taken into account. Availability is one of the primary focuses of these types of systems, as they can operate critical processes within a facility such as a nuclear power station.

9.2.1 Honeypot Characteristics

Contrary to asking what characteristics real systems have, we have to ask what a honeypot looks like. In a previous survey of ICS honeypots (Maesschalck et al., 2022), many characteristics were found that could lead to a worse capability to fool adversaries into attacking the deployed honeypot. These ranged from the deployment of a PLC on Amazon's AWS to deploying the basic configuration of a honeypot tool such as Conpot. Low-interaction honeypot tools such as Conpot have many common signatures themselves when they are not properly deployed, which can lead to a difference in traffic to the system (Maesschalck et al., 2021). Another important indication a system might be a honeypot is when there is no traffic or suspicious traffic going to the system. For example a series of messages that keeps repeating itself. Many of these characteristics can be found in the initial state of the Cyber Kill Chain (Yadav and Rao, 2015), reconnaissance when an attacker does asset discovery.

When evaluating the characteristics of honeypots, it has to be noted that there are many differences between deployments. Broadly, these can be linked to their level of interaction (low-, medium- and high-interaction) (Mokube and Adams, 2007), which are an indication of how much the system responds to the actions an adversary makes. The obfuscator honeypot itself could be viewed as a hybrid-interaction honeypot with the services running on the obfuscator being low- to medium-interaction and the PLC behind not being an actual honeypot but providing similar functionality to a high-interaction PLC honeypot.

9.2.2 Ensuring Availability

As stated previously, availability is one of the most important characteristics of industrial control systems. Therefore, this has to be at the forefront of implementation when considering the use of an obfuscator. The main concern when using an obfuscator has to be what would happen when the system went offline. This can happen for many reasons, such as the OS running into an error due to an adversary that gained access to the system or a general issue with the device hardware. Therefore, when implementing an obfuscator, we aim to ensure availability to the PLC in any case. For this, we have leveraged software-defined networking (SDN) to allow us to deploy the obfuscator physically detached from the PLC, but from a network perspective, they will still pose as the same device.

9.2.3 Increasing Security

This system's main goal must be to increase the security of the industrial control systems deployed within the network. To achieve this, there is no one solution. We expect the obfuscator to be deployed alongside other security measures and for it to be seen as an extension to those systems. The use of firewalls, secure programming practices and other good security practices has to remain. Especially within critical environments, there are a wealth of standards and guidelines that need to be followed, and honeypots can fit into (Maesschalck et al., 2022). Due to the obfuscator behaving similarly to a honeypot, it can also fit within these, although the aim is to discourage adversaries from interacting with the system aside from basic reconnaissance. By deploying the honeypot around the real PLC, we aim to divert the attacker's attention from the industrial protocol to the honeypot services. If an adversary were to interact with the obfuscator, there should be a link to other security systems such as an IDS and alert the SOC.

9.3 Obfuscator Architecture

As we are working in ICS, we have to keep the Purdue Model (Figure 2.1) in mind. This model describes how the OT environment is structured and how the interaction between devices situated on different layers happens. Within this Chapter, we assume we have a setup that conforms to the model and only focuses on the OT environment, specifically level 2 and level 1. For this proof-of-concept the main focus will be on a PLC, which resides on level 1 of the Purdue Model. This allows us to focus more on the concept, implementation and working of the obfuscator. Level 2 encompasses HMI and SCADA systems, which can also be deployed as a fake honeypot or obfuscator in a similar manner.

Within this section, we introduce the architecture of our obfuscator, followed by how the obfuscator would be viewed on the network. Finally, we describe how the obfuscator can be implemented within an SDN environment.

9.3.1 Obfuscator

The goal of the obfuscator is to have the system resemble itself as a honeypot effectively; in other words, it should make the system look less like an actual PLC. With this, we want to ensure that the system looks more like a honeypot during the asset discovery or reconnaissance phase. We can do this by looking at some existing ICS honeypots, such as Conpot and mimicking their configuration. This means we can have our obfuscator run services like HTTP, SSH and FTP. However, there is no limit to the services the obfuscator can run. In addition to this, there has to be a system connected to these services that interacts with these services in a suspicious manner. For example, a repeated series of the same messages will have an adversary question if the traffic is real or simulated. For every PLC, there also needs to be a forwarder configured so that all communications aside from the industrial control port traffic get sent to the obfuscator instead of the PLC. Industrial traffic can happen at several ports, such as 502 for MODBUS or 102 for Siemens S7comm. Therefore, the obfuscator is twofold. One part is the honeypot, and the other part is the forwarder that enables the honeypot and the obfuscated system to present as one system.

The obfuscator services themselves can be deployed very flexibly. They can range from basic implementations deployed by importing libraries in a small script, such as seen on a low-interaction honeypot, to actual deployments of the services, such as seen with a high-interaction honeypot. However, as these systems are deployed to make attackers believe they are on a honeypot, there should be a limit to how extensive these are implemented. The obfuscator should be a clearly worse deployed honeypot than the actual honeypots deployed

within the environment. Additionally, all traffic captured by the obfuscator should be logged and sent to a device that is connected to the SIEM and SOC.

In terms of the implementation of the forwarder, we would recommend this to be done on the network infrastructure itself, such as through SDN, rather than on the obfuscator honeypot itself. This is mainly to make sure that if the obfuscator goes down, the PLC itself still receives all the industrial port traffic. This is important as we do not want to interfere with the device's actual operation. Additionally, the obfuscator's goal is not to prohibit any attacks on the PLC but rather to make adversaries less likely to carry them out. Therefore, the system does not prohibit an attacker from performing any action on the OT protocols that are running on the PLC and have to be accessible; this system needs to be deployed alongside other security measures. When deploying the forwarder, it is crucial that this goes both ways, and the traffic going back to the client/attacker is structured as if it comes from the actual PLC. The obfuscator honeypot should not interfere with the operation of the PLC and preferably be deployed separately to make sure the PLC remains operational when the honeypot is unresponsive.

9.3.2 Obfuscator from a Network Perspective

From a network perspective, the obfuscator has to be indistinguishable from the actual PLC. There should be no difference between the response of the obfuscator and the PLC. Therefore, from an attacker's perspective, it has to look like they are interacting with the PLC even though the obfuscator honeypot responses (Figure 9.1). This links in with asset discovery and reconnaissance, and an attacker at this stage should believe the system is more likely to be a honeypot and not see that services are running on a separate system. Any data obtained by the honeypot should feed back into the security systems within the network.

The network is a standard OT network deployed according to the Purdue model we discussed previously. In our example network overview, we have several HMIs that are each configured to communicate with one of the PLCs. Both devices represent level 1 and level 2 of the Purdue model. We have included a historian and remote access server to represent level 3 of the model. Level 4 and level 5 are incorporated into the Enterprise Network. When we add some level 0 devices such as a pump, sensor, robot and actuator, we complete the Purdue model. This is an essential high-level representation of an OT environment within an organisation.

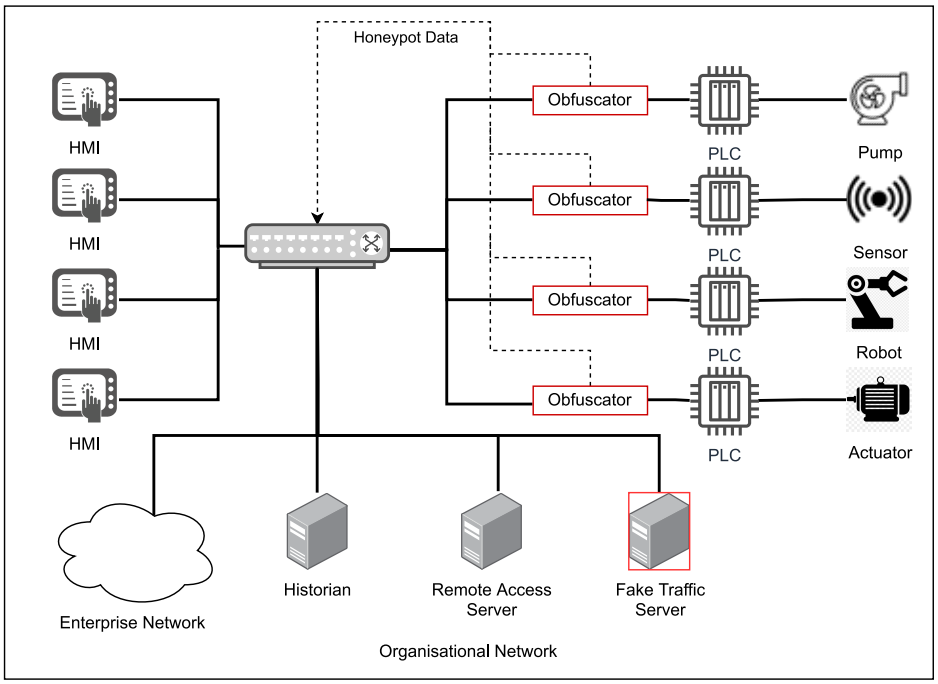


Fig. 9.1 Obfuscator OT Network Perspective

9.3.3 SDN Enabled Obfuscator

Software-defined networking has changed networking into a dynamic area. Instead of needing to change cables and configurations physically, the network can be reconfigured virtually. Although SDN has not yet propagated throughout the industrial internet yet, there are many opportunities for the OT environment to benefit from its deployment. Benefits such as the ability to amalgamate a copy of all traffic on the network and route it to an intrusion detection system can significantly improve the level of oversight and security within the OT network (di Lallo et al., 2017). Additionally, SDN can be helpful in automatically evaluating security policies on the network, whereas doing that in a non-SDN environment can be both time-consuming and impact the critical systems themselves (Ravindrababu and Alves-Foss, 2020). However, this does not mean there are no downsides to the implementation. Introducing a single point of failure, the SDN controller, can provide adversaries with full control of the network when taken control, particularly with the weak security of industrial protocols (Gardiner et al., 2021).

Leveraging SDN within this environment enables many more functionalities than when the obfuscator is deployed within a traditional network and also allows the forwarding of traffic across the network. This means that the issue related to the availability of the device we have discussed earlier in the Chapter as the forwarding of traffic to and from the obfuscator can be handled on the network infrastructure/SDN controller itself.

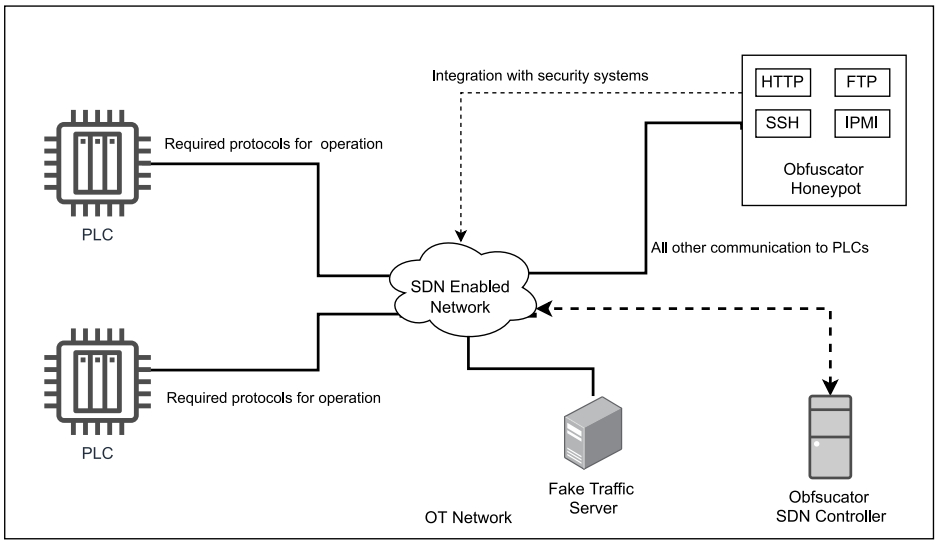


Fig. 9.2 Example of SDN Enabled Obfuscator Architecture

Implementing SDN within the underlying architecture does not change the adversary's perspective but allows for more flexibility within the existing network architecture in terms of resilience and traffic engineering, such as forwarding traffic streams to an IDS (di Lallo et al., 2017) or as a flexible security feature (Tsuchiya et al., 2018). An overview of an SDN-enabled architecture for the deployment of the obfuscator can be seen in Figure 9.2. Within the example architecture, OpenFlow is leveraged in combination with Ryu to provide the capabilities needed within the environment. Instead of these, SDN can also be implemented with solutions such as P4, Cisco ACI, and ONOS. The controller determines the traffic flow to the PLCs and to the obfuscator honeypot. Both PLCs are obfuscated by the honeypot and therefore receive traffic directed to both PLCs, but the SDN controller shows the traffic originating from the respective PLC instead of the honeypot which can assist with network monitoring. On the network itself, this would not be noticeable. Similar to the non-SDN environment, the fake traffic server is on the network providing traffic to the obfuscator honeypot to allow passive network scans to pick up the services.

9.4 Obfuscator Model

We have constructed a Hidden Markov chain model to obtain a theoretical evaluation of the obfuscator. Within this model, we assume the obfuscation successfully makes the obfuscator honeypot and the real PLC present as one system. In a real environment, this would depend on the implementation of the obfuscator forwarder. We evaluate our implementation of the forwarder later in the Chapter. This model represents one instance of the obfuscator

and allows for flexibility in deploying multiple services. Due to the nature of the system, the model represents different states of which the next states depend on the states already visited. A service $(S_1 \rightarrow S_n)$ that looks more similar to one that might be deployed on a (poorly) deployed honeypot would result in the adversary estimating a higher probability (belief) that the system is a honeypot (Holz and Raynal, 2005; Maesschalck et al., 2022; Naik et al., 2018) which is the opposite of the goal of a honeypot (Thom et al., 2021). If the adversary engages in multiple honeypot-like services, the total belief weight increases more significantly than the total belief weight the system is a real system. The exact belief weight an adversary believes a system is a honeypot (h) or not (r) depends significantly on the type and configuration of the service deployed. Both belief weights have to be a value between 0 and 1, with the sum of both equating to 1. The probability of the adversary moving across services is determined to be random as the attacker can move freely within the system. However, suppose the belief weight the system is a honeypot and the belief that the system is a real system is similar or low. In that case, an adversary is more likely to move to another service or return to the service to continue investigating, as can be seen with different honeypot deployments (Maesschalck et al., 2021). When this confidence is high, an adversary could leave the system if they determine it to be a honeypot, as they do not want to risk being exposed (Alshamrani et al., 2019; Holz and Raynal, 2005; Spitzner, 2003), and might be likely to investigate the actual OT protocol more if they determine it is a real system. However, they would not be likely to move to another service if they believe the system is a honeypot and might not be a valuable target or if the risk of exposure is too great (Chen et al., 2014; Naik et al., 2018). The sum of the belief weights has to remain between 0 and the number of services the attacker has visited as we assume the attacker has no belief before interacting with the system.

In order to calculate the probability of an adversary deciding whether the system is a honeypot or a real system, we need to use a formula that can encompass the adversary visiting multiple services (X). For each service visited, the attacker obtains a belief that the system is a honeypot (h) and a belief that the system is not a honeypot (r). For each extra service the attacker visits, the beliefs get added to the current belief the attacker has. The value of this at the end is the total belief that the attacker thinks the system is a real system (R) or a honeypot (H). To calculate if the attacker believes the system is more likely to be a honeypot (E[H]), we use the total belief weight it is a honeypot (H) and divide this by the number of services the attacker visited (n(X)). Dividing H by the number of services visited normalises the value to be within a range of 0 to 1. This gives us the probability that the attacker believes the system is a honeypot. Subtracting this value (E[H]) from 1 gives us the probability that an attacker believes the system is a real system (E[RS]). This is also obtainable by replacing

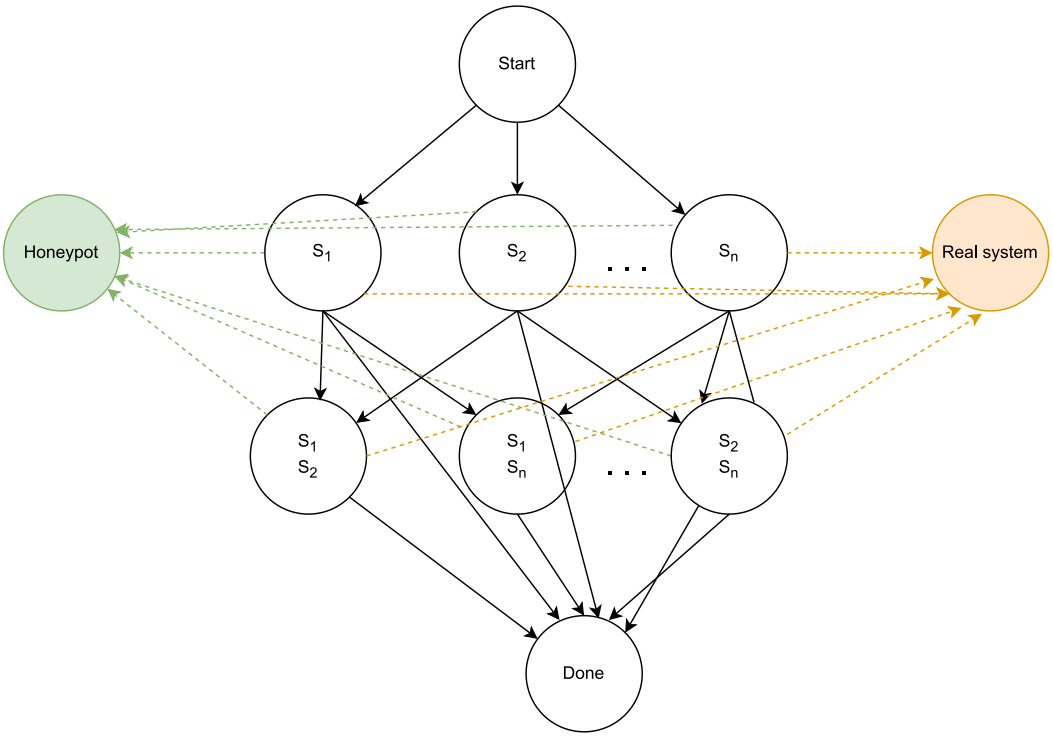


Fig. 9.3 Generic Model of the Obfsucator System

Service	HP Belief Weight	RS Belief Weight
S_1	h_1	r_1
S_2	h_2	r_2
...	...	...
S_n	h_n	r_n

Table 9.1 Generic matrix of Honeypot and Real System beliefs

H with R in the formula used to calculate E[H]. If this value is greater than 0, the attacker believes the system is a honeypot rather than a real system. This has to be calculated when the attacker reaches the 'done' state of the model based on all the services they have visited. The following formulas can therefore be derived:

$$H = \sum_{n \in X} h(S_n)$$

$$R = \sum_{n \in X} r(S_n)$$

$$E[H] = \frac{H}{n(X)}$$

Service	HP Belief Weight	RS Belief Weight
HTTP	0.95	0.05
FTP	0.91	0.09
MODBUS	0.08	0.92

Table 9.2 Example Matrix of Honeypot and Real System Beliefs

$$E[RS] = \frac{R}{n(X)}$$

or

$$E[H] = \frac{\sum_{n \in X} h(S_n)}{n(X)}$$

$$E[RS] = \frac{\sum_{n \in X} r(S_n)}{n(X)}$$

We have adapted the generic model of the obfuscator to represent and evaluate one possible configuration (Figure 9.4). This example represents a deployment of the obfuscator where three services are accessible: an emulated HTTP and FTP service and the real Modbus service running on the actual PLC. We assume the attacker has free choice to start with either service and at the end state of the model, the attacker has assumed the system is either a honeypot or a real system. Beliefs within this model (Table 9.2). We have used a Conpot honeypot (running HTTP and FTP) and an Allen-Bradley PLC (running MODBUS) as an example system. These numbers are specific to our deployment and configuration of these services. How these services are deployed, e.g. a dummy FTP service with no interaction or an HTTP service with a different website, will impact the values. Therefore, these should be taken as an example for our system and might not necessarily be the same in another deployment. Mapping these beliefs to the previously described functions shows that an attacker is significantly more likely to believe the system is a honeypot rather than a real system. We have mapped these for three scenarios which the attacker could follow.

- Scenario 1: The attacker initially investigates the FTP service; from this, they move onto the Modbus service, which then leads to the HTTP service. The honeypot belief weight the attacker has is 1.94 (honeypot belief of FTP + Modbus + HTTP) compared to 1.06 (real system belief of FTP + Modbus + HTTP) it is a real system when they reach the final state. When we (calculate $E[H]$), the result is 0.647 and the expectation that the system is a real system (E[RS]) is 0.353. Therefore the attacker determines there is a greater probability that the system is a honeypot than a real system.

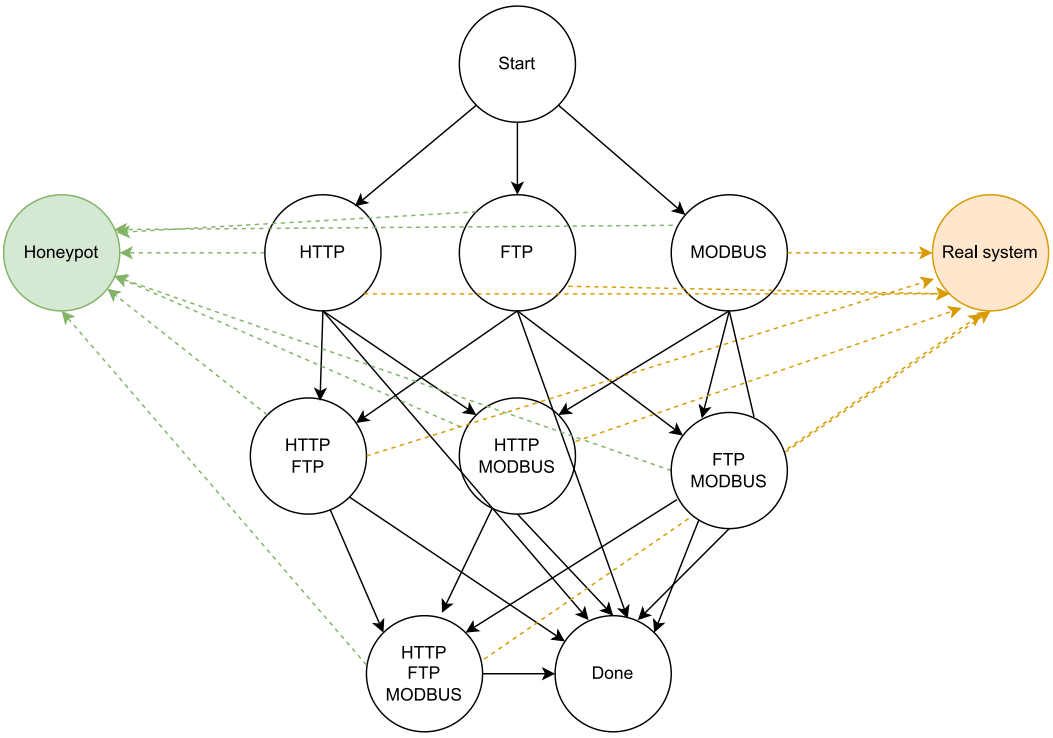


Fig. 9.4 Example Model of the Obfuscator System

- Scenario 2: The attacker initially investigates the HTTP service; from this, they leave the system. The probability the attacker thinks the system is a honeypot is 0.95 (honeypot belief of HTTP) compared to 0.05 (real system belief of HTTP) it is a real system when they reach the final state. When we (calculate $E[H]$), the result is 0.95 and the expectation that the system is a real system (E[RS]) is 0.05. Therefore the attacker determines there is a greater probability that the system is a honeypot than a real system.

- Scenario 3: The attacker initially investigates the Modbus service; from this, they move on to the HTTP service. Afterwards, they leave the system. The probability the attacker thinks the system is a honeypot is 1.03 (honeypot belief of Modbus + HTTP) compared to 0.97 (real system belief of Modbus + HTTP) it is a real system when they reach the final state. When we (calculate $E[H]$), the result is 0.515 and the expectation that the system is a real system (E[RS]) is 0.485. Therefore the attacker determines there is a greater probability that the system is a honeypot than a real system.

- Scenario 4: The attacker initially investigates the Modbus service; afterwards, they leave the system. The probability the attacker thinks the system is a honeypot is 0.08 (honeypot belief of Modbus) compared to 0.92 (real system belief of Modbus) it is a

real system when they reach the final state. When we (calculate $E[H]$), the result is 0.08 and the expectation that the system is a real system (E[RS]) is 0.92. Therefore the attacker determines there is a greater probability that the system is a real system than a honeypot.

From this, we can see that because the Modbus protocol is not emulated and is part of the real system, it reduces the chance an adversary would view the system as a honeypot compared to the other services. However, if we consider a more appropriate path for an attacker investigating multiple parts of the system, the chance an attacker determines the system is a honeypot is significantly higher. It would be improbable that an attacker would only focus on the OT protocol and disregard investigating the other services running on the system. In addition to this, once an attacker has interacted with services on the system, aside from the real service, they should have been noticed by the SOC as the honeypot aspect of the system should be linked in with other security systems.

9.5 Evaluation

9.5.1 Setup

The evaluation setup focuses on the obfuscator rather than the network around it. For this, we deploy one PLC with one Conpot obfuscator honeypot and a Ryu-based (Tomonori, 2013) obfuscator controller within our test network (Figure 9.7), which will be running within our ICS lab deployed according to the testbed proposed by Green et al. (2020) and uses IPv4 as IPv6 has only recently been supported by Siemens PLCs (Siemens, 2019b). The PLC deployed is a Siemens Simatic ET 200S and is connected to the obfuscator and attacker system through a Pica8 P-3297 switch deployed in OVS mode. The obfuscator is configured to redirect traffic to a Conpot honeypot (Ubuntu 20.04 LTS, i5-8265U, 8GB) which is set up in a default configuration emulating a Siemens Simatic ET 200S PLC. As this template also includes an S7Comm emulation, we have disabled this as the Siemens PLC provides this service. With this setup, the obfuscator presents itself as a Siemens Simatic ET 200S PLC, as the real device it is connected to, and it has all the default Conpot signatures. The obfuscator controller, running on Raspberry Pi4, is designed to obfuscate traffic by redirecting traffic to the Conpot honeypot if any protocol other than S7Comm or SNMP is used. The honeypot itself is obfuscated as all packets directed to its IP address that are not redirected from the obfuscated PLC IP are dropped. We have also deployed a system providing fake traffic to the honeypot on a regular basis. This system connects to the Modbus, HTTP, Bacnet, IPMI,

FTP and TFTP services by sending a basic request to those services sequentially on a rolling basis.

To establish a baseline for our obfuscation, we also run the tests on the same PLC. We do this by bypassing the SDN environment by directly connecting the PLC to the system we use as the attacker.

Conpot is running the following built-in protocols which are part of its Siemens S7-200 emulator:

- Modbus on port 502 TCP
- HTTP on port 80 TCP
- Bacnet on port 47808 UDP
- IPMI on port 623 UDP
- FTP on port 21 TCP
- TFTP on port 69 UDP

The obfuscator itself uses the Ryu framework for OpenFlow controllers, where it acts as a firewall, layer 2 learning switch, and an obfuscator. The functionality of this controller is split across five flow tables, as shown in figure 9.6, where all static logic is populated based on a provided configuration. When a packet reaches the SDN switch it checks if there is a dynamic flow rule written to handle the packet, if not the controller installs a new dynamic rule. This only happens when a packet matches one of the firewall rules and does not match a bypass rule which dictates if a packet can bypass the obfuscator and go to the PLC. The configuration 9.5 primarily consists of a set of hidden nodes representing any PLCs within the network; and a set of obfuscation rules, in which a set of header values are provided along with a honeypot node that should handle any traffic of that type. From this, the controller ensures that all incoming traffic to the network destined for a node used as a honeypot is dropped by the firewall table. The controller then installs flow entries in the static obfuscation table to identify any traffic that should be obfuscated, associating any packet in messages with a unique cookie per obfuscation rule. Then any future packet that matches with a static obfuscation rule will trigger the controller to install the appropriate bi-directional dynamic rule. The result of this is that any packet addressed to a protected node of a traffic type specified in an obfuscation rule will have its destination MAC and IPv4 addresses rewritten with that of a honeypot, and any returned packets will have their source MAC and IPv4 rewritten to that of the original packet's destination.

```
#bypass the obfuscator
[[86fe486e7302002f.bypass]] #port 102
eth_type = 2048
ip_proto = 6
tcp_dst = 102

[[86fe486e7302002f.bypass]] #port 161
eth_type = 2048
ip_proto = 17
udp_dst = 161

#Redirect any IP traffic heading to any of the protected nodes redirect to the honeypot
#Higher priority comes first (e.g. 2 then 1)
[[86fe486e7302002f.rule]] #All IP traffic from PLC to honeypot
id = 1
name = "redirect ip"
priority = 2
[86fe486e7302002f.rule.match]
eth_type = 2048
[86fe486e7302002f.rule.honeypot]
mac_address = "98:fa:9b:53:c5:17"
ipv4_address = "172.21.1.190"

[[86fe486e7302002f.rule]] #HTTP to honeypot
id = 3
name = "redirect HTTP"
priority = 4
[86fe486e7302002f.rule.match]
eth_type = 2048
ip_proto = 6
tcp_dst = 80
[86fe486e7302002f.rule.honeypot]
mac_address = "98:fa:9b:53:c5:17"
ipv4_address = "172.21.1.190"
```

Fig. 9.5 Example of Obfuscator Flow Rules

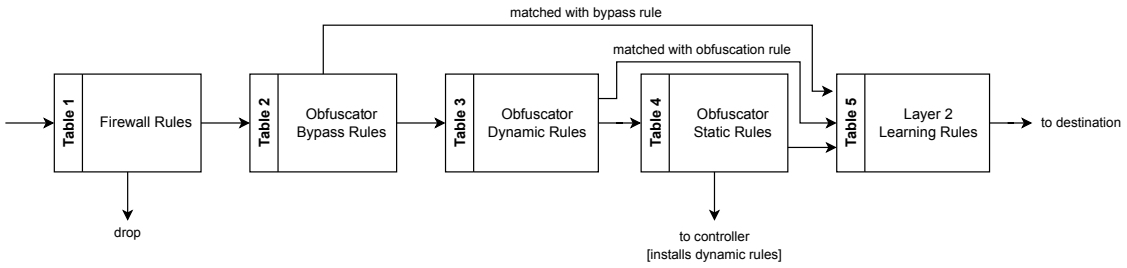


Fig. 9.6 Flow Tables used by the OpenFlow Obfuscator

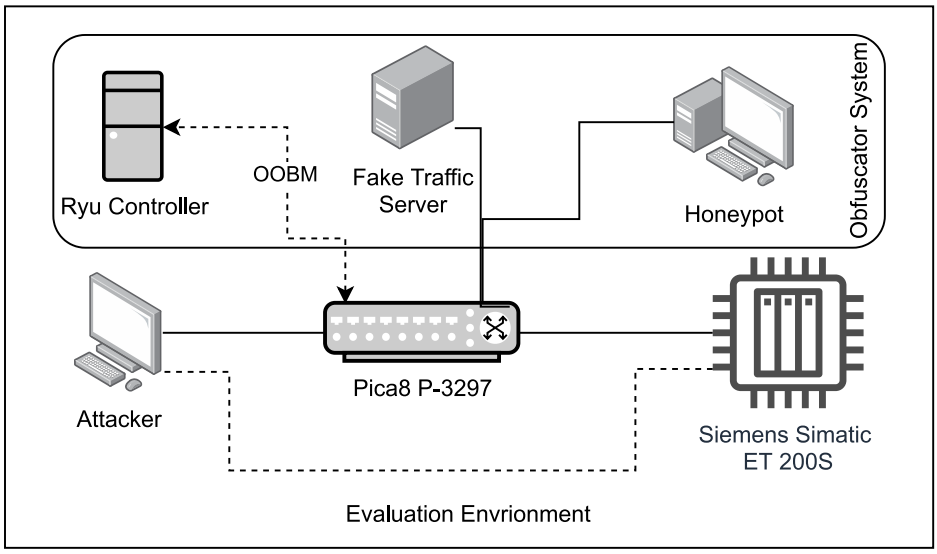


Fig. 9.7 Diagram of Evaluation Scenario

9.5.2 Methodology

To evaluate the success of the obfuscator in obfuscating the PLC to present itself as a honeypot, we set up the test environment as described earlier. Within this environment, we have run a number of penetration testing and network evaluation tools to evaluate how the system presents itself to adversaries running these tools. An overview of the tools we have used within the environment can be found in Table 9.3. Many of these tools, such as Nmap and Wireshark, are used for asset discovery, and Wireshark, as an example, can also be used for passive discovery of devices on the network. We understand that Nmap is not commonly used by adversaries within these environments due to the noise it generates on the network, however, within our evaluation it can show us how our system would behave when scanned. The tools we use in this evaluation generally fit with the MITRE ATT&CK for ICS Matrix (Alexander et al., 2020; MITRE, 2020) discovery tactic and can be used for network sniffing, network connection enumeration and other techniques listed under this tactic. We have done this as asset discovery is what our obfuscator and evaluation focuses on this area of the cyber kill chain.

We have run each of these tools on both the obfuscated PLC and the non-obfuscated PLC IP address. This has been done to understand the impact the obfuscator has on network performance, how well the obfuscator is able to hide from being a separate system and most importantly, the capability to obfuscate the PLC as a honeypot from an attacker's perspective.

Penetration testing	Networking
Nmap	Wireshark
Nessus	Snap7
PLCScan	Ping
S7Scan	httping
	hping3

Table 9.3 Penetration Testing and Networking Tools used for Evaluation

Obfuscation

To evaluate the capabilities of the obfuscator to obfuscate, we have deployed several tools on both the obfuscated and non-obfuscated PLC. In terms of penetration testing tools, we have deployed all tools listed within Table 9.3. Nmap has been used to evaluate how the obfuscator responds to network scanners commonly used within penetration tests. This indicates if the scanners reveal any indication that the S7 protocol does not run on the same system as the services run by Conpot. Nessus, PLCScan and S7Scan focus on vulnerability scanning. Running these tools indicates which vulnerabilities can be found through the obfuscation, which cannot be found on the PLC. In addition to these penetration testing tools, we ran Wireshark, Ping, HTTPing and Snap7. The Snap7 script used runs the following commands: plc.get_connected, plc.get_cpu_info, plc.get_plc_datetime, plc.ab_read(1,5), and plc.upload(1). These commands try to connect to the PLC and perform several tasks, such as enquiring what CPU is running and downloading data from the device (upload).

Wireshark provides an overview of activity on the network and if network traffic to the obfuscated IP differs from the traffic sent to the non-obfuscated PLC. To further evaluate the network-level obfuscation, we deploy Ping to evaluate if there is any difference in delay between the host and the services running on Conpot and the host and S7Comm on the PLC. Finally, Snap7 is used to evaluate the response to S7comm requests through the obfuscated and the non-obfuscated PLC.

Performance

To evaluate the performance of the obfuscator, we have deployed three networking tools within our evaluation environment. We used Ping and HTTPing to evaluate the delay between the host, the obfuscated services and the S7comm service running on the PLC through the obfuscator. In addition to Ping, we have used a script to evaluate the amount of traffic our Ryu obfuscator controller and the non-obfuscated PLC can handle. When using the obfuscator the first packet that reaches the switch is evaluated to install and learn the rules, which means

subsequent packets are handled quicker. In the non-obfuscated environment the packets go straight to the PLC. The script leverages both HPing3 to flood the environment with data and Ping to evaluate the performance of the network at the same time, starting at 2 packets per second (0.13 KBps) and increasing to 1,000,000 packets per second (64 MBps).

9.5.3 Results

Without Obfuscation

When running Nmap scans on the non-obfuscated PLC, Nmap detects only port 161/udp that is running SNMP and port 102/tcp that is running iso-tsap, which is the S7comm service. Running the Nmap built-in S7-info script, Nmap returns port 102/tcp as open and gives the correct information about the PLC (e.g. System name, MAC address and Serial Number). Nmap fails to detect the OS running on the machine (on all scans) and guesses it to be devices such as BorderGuard firewall, Xerox printer and Motorola wireless bridge. PLCScan and S7Scan both return the expected information of the PLC. Nessus returns one high-risk SNMP vulnerability. All Snap7 commands return the expected values and do not return any errors. Wireshark does not show any odd behaviour, and all packets show to be sent from and to the correct machines. Ping returns an average RTT of 5ms, a maximum RTT of 9ms, and a minimum RTT of 3ms for 50 pings. The PLC starts to drop packets (>10%) when sending 10,200 packets per second (0.65 MBps) (9.8) and starts to significantly drop packets (24%) at 10,600 packets (0.68 MBps). The highest packet loss (100%) is first reached at 55,500 packets per second (3.55 MBps). The maximum RTT (11483 ms) observed during the test is at around 52,500 packets per second (3.36 MBps), the average RTT maximum throughout the test is 689 ms, and the average of the average RTT is 119 ms.

With Obfuscation

Running Nmap on the obfuscated PLC returns a wealth of information. Scanning TCP ports, Nmap returns FTP running on port 21, HTTP on port 80, iso-tsap (S7comm) on port 102 and mbap on port 502 (MODBUS). The s7-info script returns port 102/tcp as open and shows the correct information about the PLC. The five highest guesses Nmap made for the OS are Linux 2.6.32 - 3.10 (93%), Synology DiskStation Manager 5.2-5644 (92%), Linux 3.1 (91%), and Linux 3.2 (91%). Scanning UDP ports on Nmap only shows port 161/snmp being open, with the other services running (TFTP, IPMI (reported as asf-rmcp), and Bacnet) being reported as openedlfiltered. Aggressive OS guesses are widely different and include Citrix Access Gateway VPN gateway (95%), Linksys WRT610Nv3 WAP (95%), and 3Com

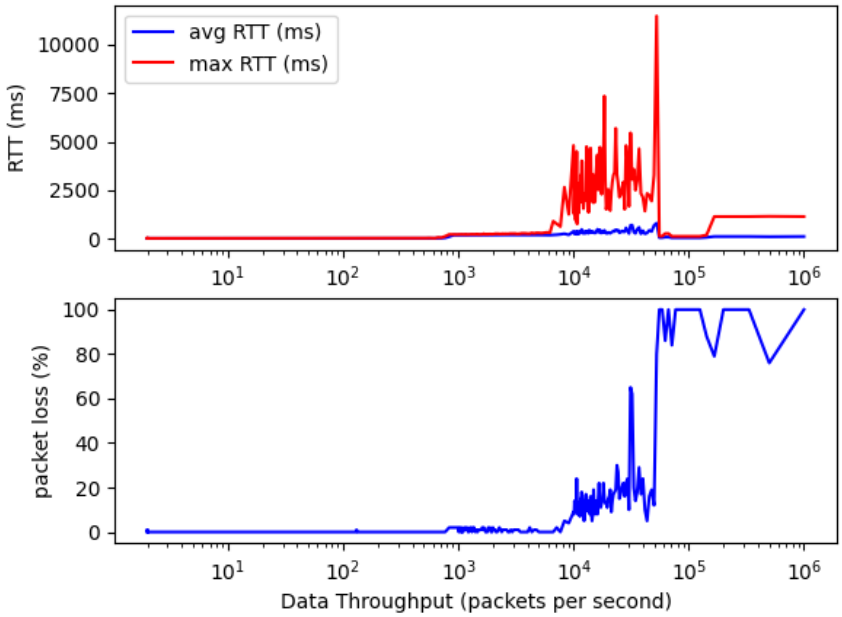


Fig. 9.8 PLC Performance Evaluation Graph

OfficeConnect 3CRWER100-75 wireless broadband router (94%). Both TCP and UDP scans return the MAC address as belonging to Siemens AG.

PLCScan and S7Scan show the expected information about the PLC, which is to be expected as these are only scanning the S7 protocol. Nessus reported no vulnerabilities on the device. When we run our Snap7 script resulted in the expected return values from the PLC. Wireshark does not show any communication with the honeypot as the source, and the Ryu obfuscator successfully obfuscated the packets as if they were sent from the obfuscated PLC.

Ping returns an average RTT of 3ms, a maximum RTT of 4ms and a minimum RTT of 2ms after 50 pings (which are redirected to the honeypot). HTTPing returns a higher average RTT of 235ms, with a maximum response time of 358ms. Looking at the packet loss in our evaluation setup 9.9 we reach a maximum of 72% packet loss at around 330,000 packets per second during our test (up to 1,000,000 packets per second), which results in 21.3 MBps. Any significant packet loss (25%) starts at 125,000 packets per second (8 MBps). The first packet loss we see higher than 10% is at around 52,000 packets (12%) or 3.37 MBps. The maximum RTT (4,718 ms) observed during the test was at 19,607 packets per second (1.25 MBps), the average RTT maximum throughout the test was 276 ms, and the average of the average RTTs was 30 ms.

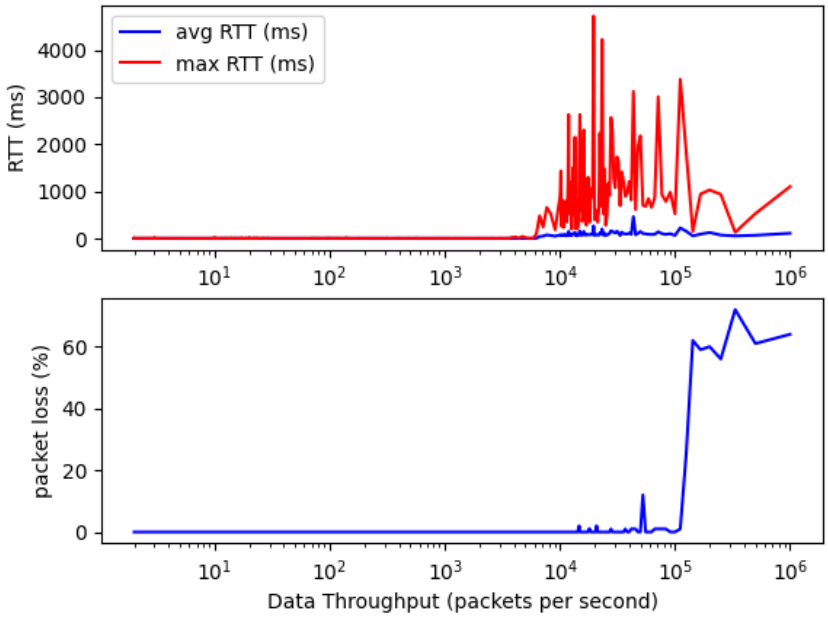


Fig. 9.9 Obfuscator System Performance Evaluation Graph

9.5.4 Discussion

Our previous work (Maesschalck et al., 2021) where we evaluated the performance of several honeypot deployments on the Internet has shown that the deployed Internet-facing ICS honeypots received a very low level of traffic on protocols such as S7Comm and Modbus. And see most traffic on IT protocols such as HTTP and FTP, protocols such as BACnet that are also used outside of industrial environments such as smart homes (Vakakis et al., 2019), and the Common Industrial Protocol (Schiffer et al., 2006) which is used to integrate with enterprise-level Ethernet networks. Furthermore, the study saw a lot of Internet scanners or sources that do not provide much information on the adversary targeting the system. During the first and second period, 24% and 21%, respectively, of IP addresses seen belonged to Censys, and 26% and 20%, respectively, belonged to Digital Ocean. Therefore this evaluation focuses on how the system would be perceived by the adversary. Via an Internet-based test we could not verify the thought process of the attacker and if they perceive the system as a real system or a honeypot.

We can see a positive outcome when looking at and comparing the results of the non-obfuscated and obfuscated PLC. Based on the data, there is no indication that our obfuscator impairs any of the functions of the PLC. This is very important in an ICS environment as availability is of utmost importance. The average RTT between the devices is nearly identical

with only a 2ms difference, in favour of the obfuscated PLC, which we can consider a similar result. Further, the stress test shows that our SDN testing environment does not impact the PLC performance as our evaluation setup first sees any noteworthy packet loss (25%) at 125,000 packets per second (8 MBps), which is significantly higher than the first considerable packet loss (24%) we have observed on the PLC at around 10, 600 packets per second (0.68 MBps). We also saw a nearly identical average RTT during the ping test, showing the SDN environment does not add an extra delay but performs similar or even better. During the stress test the average and maximum RTT were also lower than the RTT when connected to the PLC.

Looking at the obfuscation capabilities, we can observe that both systems present as one from a network perspective, and all communication going to the attacker originates from one IP address (from the attacker's perspective). However, in our tests, it is important to note that the MAC address does report back as belonging to a Siemens AG device. This could be mitigated by using the IP address of the honeypot as the target IP address and obfuscating the IP of the PLC. Further, we can see that all S7Comm-related scanners and communication reports back in the exact same way on both the obfuscated and non-obfuscated PLC. Therefore, the obfuscation does not impact the PLC's expected performance.

The OS guesses from Nmap show differences when scanning the TCP or UDP protocols. Scanning the TCP ports on the obfuscated PLC reports a high probability of the system being a Linux-based system which is what the honeypot is running. When scanning all UDP ports, only SNMP returns as being open, and the OS guesses are similar to the guesses on the non-obfuscated PLC. One odd result can be seen on the Nessus scan. For the obfuscated PLC, Nessus does not report any vulnerability. In contrast, the non-obfuscated PLC returns one high vulnerability on the SNMP service, which bypasses the honeypot on the obfuscated PLC.

Based on these results, we can conclude that the goal of the obfuscator has been achieved as both systems show as one from a network perspective. As none of the tools we ran on the system give any indication that there are actually two different systems responding to the attacker, the average attacker would not notice this. However, a highly-sophisticated attacker could potentially have specific tools they might have developed themselves that could give an indication that these are two separate systems. We deem this a possibility with every security measure, as there can always be ways highly-knowledgeable attackers can circumvent a security measure. Furthermore, from running the tools, the attacker would see multiple services running on the device and spot the Conpot signatures, which can be found on the website. Because we are not using the S7Comm service of Conpot, not all signatures can be seen on the system. Looking back, if the attacker ran Nmap on the PLC, it is highly

likely they would have interacted with the honeypot services, which would result in the SOC being notified of unusual activity, which is beneficial to the security of the PLC.

9.6 Deployment Considerations

When implementing an obfuscator within an industrial network, attention has to be paid to the requirements of the network. As stated previously, it is vital to keep availability, reliability and safety in mind. To satisfy this, a risk assessment can be done on the implementation of the obfuscator and the impact it can have on the network and system it is connected with, and on the implementation of SDN if this will be implemented as well.

SDN can also be used for other purposes, such as rerouting possible adversaries that are hitting the obfuscator to fake honeypots, PLC or a high-interactive honeypot environment. This can be done by implementing ML algorithms or other practices that classify interactions with the obfuscator. Systems for intrusion detection in ICS environments, such as DEIDS (Gu et al., 2022) and Hamids (Ghaeini and Tippenhauer, 2016), have already been developed that could be implemented in this system. This further has to be in line with other logging and monitoring systems deployed and the operational process in the SOC. Implementation of this has to be evaluated when the system is implemented.

Instead of deploying an actual system to function as an obfuscator honeypot, the services can also be virtualised by deploying bespoke Docker containers that implement a specific service or set of services. This can be extended by dynamically spinning up a Docker instance for each service an actor within the network tries to connect to. Doing this both provides an individual environment for each adversary within the network and allows for flexibility in the deployment of the obfuscator. When implementing the forwarder part of the obfuscator, it is essential to evaluate the realism of the forwarding. It is important that an attacker cannot distinguish between a reply from the obfuscator and a reply from the actual PLC. Certain characteristics of the network stack might have fingerprintable information regarding the source system. This further ties in with ensuring the obfuscator itself is not directly accessible by an attacker. Otherwise, they might be able to identify the system as an obfuscator.

Finally, for evaluation purposes, we have deployed a limited Ryu version of the obfuscator controller in a bespoke environment, combined with a Conpot honeypot. This has been done to allow us to evaluate the concept of obfuscating PLCs in a practical setting. Obfuscating a PLC within a real-world environment would require developing code specific to this environment and analysing the security of the obfuscator code before deployment.

9.7 Conclusion

This Chapter has introduced and evaluated the concept of obfuscation within industrial control system environments. We have shown that obfuscation is a valid approach to reduce attackers' appetite to interact with certain systems and that obfuscation can successfully be deployed within an ICS environment. Our evaluation happened within a physical environment using a Pica8 Openflow SDN switch, Ryu-based Openflow controller and a Siemens S7 ET 200S PLC. A Hidden Markov model further strengthened this evaluation with a theoretical approach.

During the practical evaluation, we ran several penetration testing tools (such as Nmap) and networking tools (such as Wireshark). Our tests have shown that our obfuscator successfully presents both the deployed honeypot and the Siemens PLC as the same system in a real environment. The tools ran on the system show no differences between them, and a Nmap scan on the PLC covers both the PLC and the honeypot. When we investigated the network traffic on Wireshark, we saw no connection originating from the honeypot going to the attacker; all packets were sent from the PLC with the correct MAC address. In addition to this, we have also shown that the Obfuscator system does not interfere with operations on the PLC as all S7comm communications still report back as expected. Therefore, the availability of the PLC has not been affected. The physical environment we evaluated the system in also performs better than the PLC itself. This means any bottleneck would originate from the PLC rather than the SDN switch, controller or honeypot. This can change depending on the devices deployed within the obfuscator system.

We believe that the obfuscator presented is feasible as a security mechanism within a network to both discourage attackers from interacting with a real PLC and increase the chances of capturing attackers on the network. Even if an attacker decides to interact with the system because it is a honeypot, the interaction should warn the SOC, which then can apply the proper mitigation. Consequently, the system can also be connected with other security mechanisms as the data captured by the honeypot can be used in systems such as an IDS.

Although our evaluation focused on a Conpot honeypot combined with a Ryu Openflow controller, the system can be deployed in many configurations. Whereas we have leveraged SDN as a tool to enable traffic redirection, other systems are equally valid. This also links back to the SDN technology used, as this system can be deployed within P4, other SDN environments or intent-based networking as well (Riftadi and Kuipers, 2019; Tsuzaki and Okabe, 2017; Xia et al., 2014).

Chapter 10

Conclusion

At the start of this thesis we set out to find out how honeypots can be used within industrial control system and critical infrastructure environment. Throughout the thesis we had to delve into the notion of cyberspace and cyberspace operations, regulations and many technical aspects of honeypots and ICSs. By doing this we accomplished the goal set out at the start, which we defined as four aims and objectives. In this conclusion we will cover all four and provide a synopsis on what we did to answer them and the overall results we have obtained.

10.1 Analyse the Role of ICS Honeypots in Critical Infrastructure

The first contribution of this thesis is ana analysis of the role of honeypots within industrial control system security and critical infrastructure regulations. To do this, we have explored some of the major relevant regulations, such as the U.S. NIST Framework for Improving Critical Infrastructure Cybersecurity, the U.K. Cyber Assessment Framework and multiple other national regulations. Some of those, such as the Spanish National Cybersecurity Institute regulations, directly refer to honeypots. However, this is not widespread. Many European regulations are tied to the EU NIS Directive; therefore, we have selected the U.K. framework as our primary focus, given that it is one of the most comprehensive implementations. We also used the NIST framework, given its widespread usage.

When mapping the benefits honeypots provide to the objectives of the CAF, we identified 3 of the 4 objectives that honeypots can support. Further, honeypots can be used in 6 out of the 16 principles that are listed within the framework across all 4 objectives. Showing that honeypots can be of significant value if an organisation wants to strengthen its alignment and adoption of the CAF (and the EU NIS Directive). Regarding the NIST framework, we

have been able to map honeypots to the protect, detect and respond objectives out of the 5 objectives. Again showing that honeypots can be helpful to better adhere to regulations and can be useful for the cyber security of critical infrastructure.

10.2 Develop a Novel Honeypot Deployment Framework

The second contribution of this thesis is the development of a novel honeypot deployment framework to improve the detection of bespoke ICS vulnerabilities. After verifying that honeypots can be mapped to the necessary regulatory frameworks, it is essential to deploy these correctly to achieve the best return on investment. Therefore we developed a framework that focuses on three places where honeypots can be developed. Out of those three, two reside within the organisation and a third links to external organisations that the organisation has no control over. This link is crucial as it helps protect the organisation from threats targeting the industry and are seen on the Internet but might not have reached the organisation's network yet. The framework's goal was to allow the deployment of honeypots safely whilst ensuring they achieve the best value. This is why there is a distinction between the operational network, where low-interaction honeypots should reside and isolated networks, where there are more possibilities. Compared to other frameworks we have identified, which focus more on the honeypot itself, our framework presents a clear overview of what an organisation should deploy in which part of the network. This is important as honeypots also introduce additional risks to the environment, which our framework aims to reduce. We have also provided some in-depth use cases of each type of honeypot that is part of the framework. This is done to allow organisations to choose which part of the framework would be most beneficial to them, as not everyone has the resources to deploy all parts of it.

To achieve the best results, we have also provided an evaluation of honeypot deployments, focusing on a low-interaction Conpot deployment. We have done this to provide evidence that a honeypot requires proper planning and configuration to be as effective as possible. Within our experiments, we saw a range of interactions within each differently deployed, Conpot honeypot signifying the difference in traffic and interest each one generated. Attaining a balance between the amount and quality of data obtained through the honeypot is important, as more data requires more time to investigate. For example, we saw a significant amount of Internet scanner traffic which is less beneficial in identifying threat actors. Based on this, we have provided several improvements that can be made to the honeypot deployment and configuration to enhance the likelihood of honeypot interactions by threat actors. In addition to this, we have identified a PLC-specific memory vulnerability for which we developed a scanner tool. The risk of this vulnerability can be partially mitigated by the deployment of

honeypots, which we have explored by deploying a PLC honeypot that continuously hashes its memory to identify any changes. This is an example of how honeypots can be used to monitor for the exploitation of particular vulnerabilities.

10.3 Measure the Fingerprintability of ICS Honeypots

The third contribution of this thesis is the measuring of the fingerprintability of ICS honeypots based on passive data analysis of Internet-wide scans. Previously we have evidenced that honeypots can be deployed to strengthen adherence to regulation and how to deploy honeypots for a variety of goals. This contribution aimed to take that a step further and investigate how wrongly deployed honeypots can be identified based on passive data and why it is essential to evaluate your own honeypot deployments. To do this, we have queried Shodan for all ICS devices they have indexed and evaluated them. We have found that several of these 'industrial' devices are not actual industrial devices but rather other systems, such as smart home systems. Secondly, on this dataset, we have run a new passive scanning algorithm for ICS honeypots that classifies each ICSs within the obtained dataset into one out of four classifications. These classifications range from not a honeypot to a definite honeypot representing the likelihood of a system being a honeypot based on initial triage. We evaluated our algorithm with active scanning and saw an accuracy of 69% when accounting for crossover between the classifications.

Additionally, we have compared our classifications with the Shodan Honeyscore, which was developed specifically for ICS honeypots. The result of this comparison showed that Honeyscore is unreliable and overly categorises systems into the 0 and 0.3 values. We have also noticed there are three main values used by Honeyscore, with 1 being the third one. A fourth one, 0.5, is also used but very rarely. Comparing the Honeyscore with the active scanning revealed systems that pass all our active tests receiving a 1 (meaning it is a honeypot), and systems that failed all our tests and showed clear Conpot signatures receiving a 0 (meaning it is not a honeypot). Overall, we have proven that passive scanning of honeypots can be done and can be used as an initial evaluation of a system. This evaluation is useful for adversaries but also for organisations deploying honeypots to appraise their deployment and identify if improvements need to be made. We have also proven that the Shodan Honeyscore is an unreliable metric to use.

10.4 Programmable ICS Obfuscation Network Overlay

The final contribution of this thesis is the design and implementation of a programmable ICS obfuscation network overlay to deflect adversaries within a critical infrastructure environment. This builds upon the three other contributions made and recognises the challenges in deploying a honeypot within any honeypot signatures. To circumvent these challenges, we have developed and evaluated a new way of using honeypots, namely, using them as obfuscation devices. We have deployed a honeypot around a real PLC, which aimed to make the PLC look more like a honeypot and, as a result, make it a less appealing target. As an initial evaluation of this concept, we created a Hidden Markov model of the system on which we investigated 4 scenarios. Out of these scenarios, 3 resulted in the adversary believing the system was a honeypot. The other failed to do this as the attacker only interacted with the actual industrial protocol. Because any security system deployed within these environments should not interfere with the operation of the PLC, after this successful theoretical evaluation, we deployed a honeypot that looked as much like a honeypot as possible and made it appear as being the same system as a PLC. To achieve this, we leveraged SDN and Ryu to allow us to change the traffic k on the network and make both the honeypot and PLC appear under one IP address. This was then evaluated within our ICS testbed by running penetration testing and networking tools and by two ICS security experts. Our tests showed that from a network perspective and according to all penetration testing tools, there is no indication that both systems are district. Further, stress testing has proven that our SDN environment was more performant than the PLC itself, and the system did not hinder the PLC operations. The external experts both agreed with our tests and did not report any indications that the honeypot and the PLC are different systems. One expert identified the system as a honeypot, proving our approach was successful. Being able to use honeypots in this manner both improves the security of the PLC and reduces the resources needed to deploy a honeypot to be indistinguishable from a real system without interfering with any operations. Making a further case for honeypot deployment within critical infrastructure OT environments.

10.5 Industry Impact

Due to the industry funding this PhD received it is important to specifically investigate the impact the contributions of this thesis have on industry. As stated previously within this thesis honeypot deployment requires a lot of thought, investment and knowledge. Therefore, a case has to be made why investing in honeypots is worth it and how to do it to obtain the most value out of them. Those were underlying principles of this thesis and have influenced

the work done. Although we might not have been able to implement any of the tools within a real operational environment due to the risks involved, our evaluation within a representative testbed and discussions with experts provided a link to these environments.

In the first contribution, we have focused on investigating the role of honeypots within an organisation. We started this by evaluating the major regulations within the area of critical infrastructure to which the organisations this PhD focuses on (critical infrastructure) has to adhere. This resulted in us establishing that honeypots are applicable to many of the objectives laid out by these regulations and can therefore be very useful for organisations to show their adherence to them. Investigating this is important to achieve a baseline why this research into ICS honeypots is applicable to industry and forms the foundation of the impact of this work.

The second contribution focused on the development of a framework to assist organisations with their honeypot deployments. This was supported by the first contribution as the U.K. Cyber Assessment Framework objectives were used in the development. These objectives were linked to the distinct locations where honeypots can be deployed within the framework and included two areas within the organisational network and one within external organisations. The aim of these is to enable an organisation to understand which types of honeypot should be deployed in those places within the network and what benefits each of those areas bring. We also shown how honeypot can be used to detect adversaries that leverage bespoke vulnerabilities, such as the novel memory vulnerability we have identified. This vulnerability by itself is an important finding for organisations, as this can put their OT environments at risk. Further, we also focused on how honeypots should be configured by presenting an experiment that shows the difference in data collected via differently deployed honeypots. This work assists organisations in deploying the right honeypot in the right place, and in the right manner to get the best value out of them, and also aids them to understand the regulatory benefits of them. We hope this empowers organisations to deploy honeypots more efficiently and with more confidence.

Within the third contribution we build upon the previous contributions to show why honeypots need to be deployed correctly. This contribution focuses on characteristics that can be used to fingerprint ICS honeypots. These characteristics are important to consider when deploying honeypots as they can be used by researchers to identify the system is a honeypot. Within this and the previous contribution we present organisations with several of these honeypot indicators as many organisations do not consider these when deploying their honeypots. These are also important to consider when organisations are scanning for ICSs on the Internet, as honeypots can skew their dataset. However, to obtain these characteristics interaction is often needed to infer if the system behaves like a honeypot. Therefore we

evaluated if honeypots can be identified through passive analysis. Our experiments show that passive scanning can be useful and our algorithm is an improvement upon the current state-of-the-art used by researchers (Shodan Honeyscore). This is important for industry and government as they can now evaluate their dataset and remove more honeypots than they previously could, resulting in a more accurate dataset.

Our fourth contribution considers the outcome of all previous contributions, focusing on how we can mitigate some of the challenges of honeypot deployments. Because deploying honeypots appropriately requires a significant amount of resources of organisations, therefore we aimed to find a way to reduce this. By fundamentally changing the usage of honeypots we managed to achieve this. Relying on the characteristics that make badly deployed honeypots, badly deployed we presented a case how honeypots can be used as deterrents within industry environments. Resulting in an increase of security whilst not requiring the investment to deploy honeypots so perfectly that they capture attackers. By negating some of the benefits we provide industry with an even stronger case why honeypots should be deployed within these critical environments.

This work could be taken by industry partners to develop further and incorporate honeypots within their areas. For example, we have had discussions with Ofgem regarding honeypots for the energy and gas sector. They can build upon this work to create guidance to deploy honeypots within the sector and can also use this to deploy honeypots themselves. We have also been working with Fujitsu, mainly regarding honeypots and threat intelligence, and they could use this work to develop novel honeypots based on the obfuscator design, use the fingerprinting to identify other honeypots and evaluate their deployments. The work we have done in the deployment of bespoke honeypots to evaluate deployments can also be used by Fujitsu to scan for the exploitation of these bespoke vulnerabilities in the wild. The fingerprinting work can also be used by Shodan, Censys and other scanners to improve their honeypot detection. The police can also use the data collected by honeypots for their investigation when criminal activity has occurred, which we hope this work could help them understand the role and technical aspects of then.

10.6 Future Work

This thesis has covered several aspects of honeypots and how they fit within critical infrastructure environments. However, only some possibilities have been explored, as they are plentiful. Further research within these fields can show new ways of enabling obfuscation to improve the obfuscator system. For example the usage of ML or AI can allow the obfuscator to be more dynamic in addition to other active defence techniques such as MTD. Future

work within this area can further delve into the usages of SDN for honeypots within ICS environments, such as redirecting adversaries towards different systems and creating more dynamic honeypot environments. This feeds into the usage of moving target defence within these environments and how honeypots can be of imperative value within this. Techniques such as MTD can be leveraged to include a dynamic part within the system as honeypots could be deployed according to the activity seen within the network. By leveraging ML together with this the obfuscator can become a truly self-learning and adaptive system to provide more ambiguity if a system is real or a honeypot. This can also make the obfuscator highly cost-effective as only the services attackers are actively looking for can be deployed. However, other areas ML and AI can be used in combination with the obfuscator is linking intrusion detection and anomaly detection to identify and redirect attackers away from the actual ICS without the risks of interupting operations. We will also conduct further evaluation of the obfuscator by deploying PLCs ranging from non-obfuscated to ones obfuscated with different honeypots to further evaluate the capabilities of the obfuscator system. This can open up new fields within deceptive cyber defence and active cyber security.

There are also several areas of research that would benefit from additional exploration from an ICS honeypot prospective. This includes an analysis of data captured by tools within the ICS honeypot environment, in order to improve their forensic value for investigators. This is particularly important for low-interactive honeypots, which have relatively limited capabilities. For this goal, it should be investigated how interactive an ICS honeypot has to be in order to capture an adequate amount of data and if low-interaction honeypots can achieve this. Further work can also look more in-depth at mappings between honeypot systems and specific levels within the Purdue model to provide complete coverage, and therefore improving the realism of the honeypot. This could also result in further improvement to Conpot and other readily available low-interaction honeypots, or the development of a new system. The potential use of AI can assist in turning low-interaction ICS honeypots more realistic by presenting adversaries with expected responses even though those services are not Incorporated on the system. On a general note, research can be done to improve the deceptiveness of honeypots, and how honeypots can be used as a preventive measure aside from their current implementations. This should definitely include the ability for honeypots to interact with phishing attacks, as this is a popular way for attackers to get into the network. As well as how honeypots can be deployed in a more defensive manner, similar to traditional security systems. Particularly interesting within this area, due to the specific nature of ICSs, would be how anomaly detection could be used with data captured from ICS honeypots and how it could feed into the security of the network.

Further work within the area of honeypot fingerprinting, primarily passive, could also be fruitful, given that we achieved positive outcomes with our experiments. A proper ML-based implementation could result in more accurate predictions if these are based on a strong fundamental dataset, which our classification method can be used for. We hope that passive scanning for honeypots can become a new field within the measurement community. Implementing honeypots within network testing and simulation for ICS environments is also worth investigating. This will allow the proper deployment of different honeypots within digital twins and training scenarios to provide benefits from a lessons learnt perspective. Finally, more work can be done in the area of using honeypots for obfuscation purposes, especially as our novel usage of honeypots proved to be very successful and provide many benefits, such as negating the need to spend resources on the realistic deployment of honeypots. These areas of future work are some of the possible areas, and there are and will be many more. The applicability of honeypots within critical infrastructure and ICSs in many ways is proven, opening up a wealth of new opportunities.

Finally, further research could include examining the applicability of honeypots within the legal environment, to give organisations both the confidence and motivation to deploy them. A thorough analysis of legal requirements could provide further evidence that such an investment benefits an organisation from both a legal and security perspective. In addition, there is a need for research that establishes clear legal guidance for the implementation of honeypots from a legal/ethical perspective. One of the main problems that arise from a honeypot deployment is the usage of those honeypots by the attacker for other goals such as botnets or dissemination of illegal material. Another problem that can arise when using honeypots is so-called entrapment, which could result in a legal procedure.

References

Ali Abbasi, Majid Hashemi, Emmanuele Zambon, and Sandro Etalle. Stealth low-level manipulation of programmable logic controllers i/o by pin control exploitation. In *International Conference on Critical Information Infrastructures Security*, pages 1–12, New York, 2016. Springer. doi:10.1007/978-3-319-71368-7_1.

Janet Abbate. *Inventing the internet*. MIT press, 2000. doi:10.2307/2652133.

Shingo Abe, Yohei Tanaka, Yukako Uchida, and Shinichi Horata. Developing Deception Network System with Traceback Honeypot in ICS Network. *SICE Journal of Control, Measurement, and System Integration*, 11(4):372–379, 2018. doi:10.9746/jcmsi.11.372.

Agence nationale de la sécurité des systèmes d'information. *Managing Cybersecurity for Industrial Control Systems*. 2014.

Palvi Aggarwal, Cleotilde Gonzalez, and Varun Dutt. Cyber-security: role of deception in cyber-attack detection. In *Advances in Human Factors in Cybersecurity: Proceedings of the AHFE 2016 International Conference on Human Factors in Cybersecurity, July 27-31, 2016, Walt Disney World®, Florida, USA*, pages 85–96. Springer, 2016. doi:10.1007/978-3-319-41932-9_8.

Atif Ahmad, Jeb Webb, Kevin C Desouza, and James Boorman. Strategically-motivated advanced persistent threat: Definition, process, tactics and a disinformation model of counterattack. *Computers & Security*, 86:402–418, 2019. doi:10.1016/j.cose.2019.07.001.

Irfan Ahmed, Sebastian Obermeier, Sneha Sudhakaran, and Vassil Roussev. Programmable Logic Controller Forensics. *IEEE Security and Privacy*, 15(6):18–24, 2017. doi:10.1109/MSP.2017.4251102.

E. Alata, V. Nicomette, M. Kaâniche, M. Dacier, and M. Herrb. Lessons learned from the deployment of a high-interaction honeypot. *Proceedings - Sixth European Dependable Computing Conference, EDCC 2006*, 22:39–44, 2006. doi:10.1109/EDCC.2006.17.

Otis Alexander, Misha Belisle, and Jacob Steele. Mitre att&ck for industrial control systems: Design and philosophy. *The MITRE Corporation: Bedford, MA, USA*, 2020.

Allen Bradley. SLC 500 Instruction Set. Technical report, Allan Bradley, 2008. URL https://literature.rockwellautomation.com/idc/groups/literature/documents/rm/1747-rm001{_}-en-p.pdf.

Adel Alshamrani, Sowmya Myneni, Ankur Chowdhary, and Dijiang Huang. A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. *IEEE Communications Surveys & Tutorials*, 21(2):1851–1877, 2019. doi:10.1109/COMST.2019.2891891.

Izzat Alsmadi, Zyad Dwekat, Ricardo Cantu, and Bilal Al-Ahmad. Vulnerability assessment of industrial systems using shodan. *Cluster Computing*, 25(3):1563–1573, 2022. doi:10.1007/s10586-021-03330-3.

Uchenna D Ani, Nneka Daniel, Francisca Oladipo, and Sunday E Adewumi. Securing industrial control system environments: the missing piece. *Journal of Cyber Security Technology*, 2(3-4):131–163, 2018. doi:10.1080/23742917.2018.1554985.

Daniele Antonioli, Anand Agrawal, and Nils Ole Tippenhauer. Towards high-interaction virtual ICS honeypots-in-a-box. *CPS-SPC 2016 - Proceedings of the 2nd ACM Workshop on Cyber-Physical Systems Security and PrivaCy, co-located with CCS 2016*, pages 13–22, 2016. doi:10.1145/2994487.2994493.

Rob Antrobus, Sylvain Frey, Benjamin Green, and Awais Rashid. Simaticscan: Towards a specialised vulnerability scanner for industrial control systems. In *4th International Symposium for ICS & SCADA Cyber Security Research 2016 4*, pages 11–18, Swindon, 2016. BCS Learning & Development Ltd. doi:10.14236/ewic/ics2016.2.

Rob Antrobus, Benjamin Green, Sylvain Frey, and Awais Rashid. The forgotten i in iiot: a vulnerability scanner for industrial internet of things. In *IET Conference Proceedings*, London, 2019. IET. doi:10.1049/cp.2019.0126.

Peter Apps. Analysis: In cyber era, militaries scramble for new skills. https://www.reuters.com/article/us-defence-cyber-idUSTRE8182HI20120209, 2012.

John Arquilla. Twenty years of cyberwar. *Journal of Military Ethics*, 12(1):80–87, 2013. doi:10.1080/15027570.2013.782632.

Michael J Assante and Robert M Lee. The industrial control system cyber kill chain. *SANS Institute InfoSec Reading Room*, 1, 2015.

AT&T. AlienVault Unified Security Management. https://cybersecurity.att.com/resource-center/videos/alienvault-unified-security-management-usm-overview, n.d.

Austrian Government. Austrian National Cyber Security Strategy. https://www.enisa.europa.eu/topics/national-cyber-security-strategies/ncss-map/AT_NCSS.pdf, 2013.

Narges Babadi and Ali Doustmohammadi. A moving target defence approach for detecting deception attacks on cyber-physical systems. *Computers and Electrical Engineering*, 100: 107931, 2022. doi:10.1016/j.compeleceng.2022.107931.

Maria Bada and Ildiko Pete. An exploration of the cybercrime ecosystem around shodan. In *2020 7th International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, pages 1–8. IEEE, 2020. doi:10.1109/IOTSMS52051.2020.9340224.

Paul Baecher, Markus Koetter, Thorsten Holz, and Maximillian Dornseif. The Nepenthes Platform: An Efficient Approach to Collect Malware. *International Workshop on Recent Advances in Intrusion Detection*, pages 165–184, 2006. doi:10.1007/11856214_9.

Radhakisan Baheti and Helen Gill. Cyber-physical systems. *The impact of control technology*, 12(1):161–166, 2011.

E Balas. Sebek: Covert glass-box host analysis. *12th USENIX Security Symposium*, 2005.

Mandrita Banerjee, Junghee Lee, and Kim-Kwang Raymond Choo. A blockchain future for internet of things security: a position paper. *Digital Communications and Networks*, 4(3): 149–160, 2018. doi:10.1016/j.dcan.2017.10.006.

Israel Barak. Cybereason's newest honeypot shows how multistage ransomware attacks should have critical infrastructure providers on high alert. *Cybereason*, jun 2020.

Matthew P Barrett. Framework for improving critical infrastructure cybersecurity. *National Institute of Standards and Technology*, 2018. doi:10.6028/NIST.CSWP.04162018.

Dustin Berman and Jonathan Butts. Towards characterization of cyber attacks on industrial control systems: Emulating field devices using Gumstix technology. *Proceedings - 2012 5th International Symposium on Resilient Control Systems, ISRCS 2012*, pages 63–68, 2012. doi:10.1109/ISRCS.2012.6309294.

Giuseppe Bernieri, Mauro Conti, and Federica Pascucci. MimePot: A model-based honeypot for industrial control networks. *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 2019-Octob:433–438, 2019. doi:10.1109/SMC.2019.8913891.

Deval Bhamare, Maede Zolanvari, Aiman Erbad, Raj Jain, Khaled Khan, and Nader Meskin. Cybersecurity for industrial control systems: A survey. *computers & security*, 89:101677, 2020. doi:10.1016/j.cose.2019.101677.

Zhuo Bi, Deji Chen, Cheng Wang, Changjun Jiang, and Ming Chen. Adopting wirelesshart for in-vehicle-networking. In *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*, pages 1027–1030. IEEE, 2015. doi:10.1109/HPCC-CSS-ICESS.2015.244.

Eli Biham, Sara Bitan, Aviad Carmel, Alon Dankner, Uriel Malin, and Avishai Woo. Rogue7: Rogue Engineering-Station Attacks on S7Simatic PLCs. Technical report, Black Hat USA, 2019.

Leyla Bilge and Tudor Dumitras. Before we knew it: an empirical study of zero-day attacks in the real world. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 833–844, Raleigh, North Carolina, USA, 2012. ACM. ISBN 978-1-4503-1651-4. doi:10.1145/2382196.2382284.

Stefano Bistarelli, Emanuele Bosimini, and Francesco Santini. A medium-interaction emulation and monitoring system for operational technology. In *The 16th International Conference on Availability, Reliability and Security*, pages 1–7, 2021. doi:10.1145/3321408.3321589.

Roland Bodenheim, Jonathan Butts, Stephen Dunlap, and Barry Mullins. Evaluation of the ability of the Shodan search engine to identify Internet-facing industrial control devices. *International Journal of Critical Infrastructure Protection*, 7(2):114–123, 2014. doi:10.1016/j.ijcip.2014.03.001. URL http://dx.doi.org/10.1016/j.ijcip.2014.03.001.

Roland C Bodenheim. Impact of the Shodan Computer Search Engine on Internet-facing Industrial Control System Devices. page 121, 2014.

Elias Bou-Harb, Walter Lucia, Nicola Forti, Sean Weerakkody, Nasir Ghani, and Bruno Sinopoli. Cyber meets control: A novel federated approach for resilient cps leveraging real cyber threat intelligence. *IEEE Communications Magazine*, 55(5):198–204, 2017. doi:10.1109/MCOM.2017.1600292CM.

BR Automation. Will the cloud replace the plc? https://www.br-automation.com/en-gb/about-us/press-room/technology-highlights/will-the-cloud-replace-the-plc/, n.d.

Pascal Brangetto and Matthijs A. Veenendaal. Influence Cyber Operations: The use of cyberattacks in support of Influence Operations. *International Conference on Cyber Conflict, CYCON*, 2016-Augus:113–126, 2016. doi:10.1109/cycon.2016.7529430.

Alan Brill. The use of internet technology by cyber terrorists & cyber criminals: The 2014 report., 2015.

Mark Bristow. ModScan - A SCADA MODBUS Network Scanner. https://defcon.org/images/defcon-16/dc16-presentations/defcon-16-bristow.pdf, 2008.

British Standards Institute. 61132-3:2013 - Programmable Controllers - Part 3: Programming Languages. Technical report, BSI, 2013.

Bundesamt für Sicherheit in der Informationstechnik. *ICS Security Compendium*. 2013.

Bundesamt für Sicherheit in der Informationstechnik. Industrial control system security. https://www.bsi.bund.de/EN/Topics/Industry_CI/ICS/Download/download_node.html, n.d.

Andrew Bushby. How deception can change cyber security defences. *Computer Fraud & Security*, 2019(1):12–14, 2019. doi:10.1016/S1361-3723(19)30008-9.

D. I. Buza, F. Juhász, G. Miru, M. Félegyházi, and T. Holczer. CryPLH: Protecting Smart Energy Systems from Targeted Attacks with a PLC Honeypot. In *International Workshop on Smart Grid Security*, pages 181–192, 2014. ISBN 9783319103280. doi:10.1007/978-3-319-10329-7_12.

Jack Caravelli. Cyber Terrorism and Covert Action. In *Cyber Security: Threats and Responses for Government and Business*, pages 1–22. 2019. ISBN 9781440861734.

Alvaro A Cárdenas, Saurabh Amin, and Shankar Sastry. Research challenges for the security of control systems. *HotSec*, 5:15, 2008.

David Carment and Dani Belo. *War's Future: The Risks and Rewards of GreyZone Conflict and Hybrid Warfare*. Canadian Global Affairs Institute, 2018.

Hasan Cavusoglu, Huseyin Cavusoglu, and Zhang Jun. Security patch management: Share the burden or share the damage? *Management Science*, 54(4):657–670, 2008. doi:10.1287/mnsc.1070.0794.

Censys. Exposure Management and Threat Hunting Solutions. https://censys.io/, n.d.

Swiss Confederation National Cybersecurity Centre. Checklist and instructions: Measures to protect industrial control systems (ICSs) . https://www.ncsc.admin.ch/ncsc/en/home/infos-fuer/infos-unternehmen/aktuelle-themen/massnahmen-schutz-ics.html, December 2020.

Saurabh Chamotra, J. S. Bhatia, Raj Kamal, and A. K. Ramani. Deployment of a low interaction honeypot in an organizational private network. *Proceedings of 2011 International Conference on Emerging Trends in Networks and Computer Communications, ETNCC2011*, pages 130–135, 2011. doi:10.1109/etncc.2011.5958501.

Check Point. Intrusion Prevention System - IPS. https://www.checkpoint.com/products/intrusion-prevention-system-ips/, n.d.

Ping Chen, Lieven Desmet, and Christophe Huygens. A study on advanced persistent threats. In *IFIP International Conference on Communications and Multimedia Security*, pages 63–72. Springer, 2014. doi:10.1007/978-3-662-44885-4_5.

Yongle Chen, Xiaowei Lian, Dan Yu, Shichao Lv, Shaochen Hao, and Yao Ma. Exploring shodan from the perspective of industrial control systems. *IEEE Access*, 8:75359–75369, 2020. doi:10.1109/ACCESS.2020.2988691.

Anton Cherepanov and Robert Lipovsky. Blackenergy-What We Really Know About the Notorious Cyber Attacks. (October):1–8, 2016.

Paul Cichonski, Tom Millar, Tim Grance, and Karen Scarfone. NIST Special Publication 800-61 Revision 2: Computer Security Incident Handling Guide. https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-61r2.pdf, 2012.

Nicholas Cifranic, Jose Romero-Mariona, Brian Souza, and Roger A Hallman. Decepti-scada: A framework for actively defending networked critical infrastructures. In *IoTBDS*, pages 69–77, 2020. doi:10.5220/0009343300690077.

David Clark. Characterizing cyberspace: past, present and future. *MIT CSAIL, Version*, 1: 2016–2028, 2010.

Claroty. Claroty Platform. https://www.claroty.com/platform, n.d.

Kevin Collier. 50,000 security disasters waiting to happen: The problem of america's water supplies. https://nbcnews.to/3gKzJ7G, June 2021.

Sean Collins and Stephen McCombie. Stuxnet: the emergence of a new cyber weapon and its implications. *Journal of Policing, Intelligence and Counter Terrorism*, 7(1):80–91, 2012. doi:10.1080/18335330.2012.653198.

Paul Cornish, David Livingstone, Dave Clemente, and Claire Yorke. *On cyber warfare*. Chatham House London, 2010.

Martin Courtney. States of cyber warfare. *Engineering and Technology*, 12(3):22–25, 2017. doi:10.1049/et.2017.0300.

J. Adam Crain and Sergey Bratus. Bolt-On Security Extensions for Industrial Control System Protocols: A Case Study of DNP3 SAv5. *IEEE Security & Privacy*, 13(3):74–79, 2015. doi:10.1109/msp.2015.47.

Cas Cremers, Martin Dehnel-Wild, and Kevin Milner. Secure authentication in the grid: A formal analysis of DNP3 SAv5. *Journal of Computer Security*, 27(2):203–232, 2019. doi:10.3233/JCS-181139.

Steven I Davis. Artificial intelligence at the operational level of war. *Defense & Security Analysis*, 38(1):74–90, 2022. doi:10.1080/14751798.2022.2031692.

Phyllis M Deane and Phyllis M Deane. *The first industrial revolution*. Cambridge University Press, 1979. ISBN 9780521296090. doi:10.1017/cbo9780511622090.

Kelley L Dempsey, L A Johnson, Matthew A Scholl, Kevin M Stine, Alicia Clay Jones, Angela Orebaugh, Nirali S Chawla, and Ronald Johnston. NIST Special Publication 800-137: Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations. https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-137.pdf, 2011.

Department for Business. Energy & Industrial Strategy. Civil Nuclear Cyber Security Strategy. https://www.gov.uk/government/publications/civil-nuclear-cyber-security-strategy, 2017.

Department of Homeland Security. DHS Works with Critical Infrastructure Owners and Operators to Raise Awareness of Cyber Threats. pages 1–3, 2016.

Richard Derbyshire, Benjamin Green, Daniel Prince, Andreas Mauthe, and David Hutchison. An Analysis of Cyber Security Attack Taxonomies. *Proceedings - 3rd IEEE European Symposium on Security and Privacy Workshops, EURO S and PW 2018*, pages 153–161, 2018. doi:10.1109/eurospw.2018.00028.

Richard Derbyshire, Benjamin Green, and David Hutchison. "Talking a different Language": Anticipating adversary attack cost for cyber risk assessment. *Computers & Security*, 103: 102163, 2021a. doi:10.1016/j.cose.2020.102163.

Richard Derbyshire, Benjamin Green, and David Hutchison. "talking a different language": Anticipating adversary attack cost for cyber risk assessment. *Computers & Security*, 2021b. doi:10.1016/j.cose.2020.102163.

Joe Devanny, Tim Stevens, Andrew Dwyer, and Amy Ertan. The national cyber force that britain needs? 2021.

Debabrata Dey, Atanu Lahiri, and Guoying Zhang. Optimal policies for security patch management. *INFORMS Journal on Computing*, 27(3):462–477, 2015. doi:10.1287/ijoc.2014.0638.

Roberto di Lallo, Federico Griscioli, Gabriele Lospoto, Habib Mostafaei, Maurizio Pizzonia, and Massimo Rimondini. Leveraging sdn to monitor critical infrastructure networks in a smarter way. In *2017 IFIP/IEEE symposium on integrated network and service management (IM)*, pages 608–611. IEEE, 2017. doi:10.23919/INM.2017.7987341.

Paul Didier, Fernando Macias, James Harstad, Rick Antholine, Scott A. Johnston, Sabina Piyevsky, Dan Zaniewski, Steve Zuponcic, Mark Schillace, and Gregory Wilcox. Converged Plantwide Ethernet (CPwE) Design and Implementation Guide. page 564, 2011.

Yoram Dinstein. *War, aggression and self-defence*. Cambridge University Press, 2017. doi:10.1017/9781108120555.

Jules Pagna Disso, Kevin Jones, and Steven Bailey. A plausible solution to scada security honeypot systems. In *2013 Eighth International Conference on Broadband and Wireless Computing, Communication and Applications*, pages 443–448. IEEE, 2013. doi:10.1109/bwcca.2013.77.

DNP. Overview Of DNP3 Protocol. https://www.dnp.org/About/Overview-of-DNP3-Protocol, n.d.

Michael Dodson, Alastair R Beresford, and Daniel R Thomas. When will my plc support mirai? the security economics of large-scale attacks against internet-connected ics devices. In *2020 APWG Symposium on Electronic Crime Research (eCrime)*, pages 1–14. IEEE, 2020. doi:10.1109/eCrime51433.2020.9493257.

Dragos. The Dragos Industrial Cybersecurity Platform. https://www.dragos.com/platform/, n.d.

Zakarya Drias, Ahmed Serhrouchni, and Olivier Vogel. Taxonomy of attacks on industrial control protocols. In *2015 International Conference on Protocol Engineering (ICPE) and International Conference on New Technologies of Distributed Systems (NTDS)*, pages 1–6, New Jersey, 2015. IEEE. doi:10.1109/notere.2015.7293513.

Nitul Dutta, Nilesh Jadav, Nirali Dutiya, and Dhara Joshi. Using Honeypots for ICS Threats Evaluation. *Recent Developments on Industrial Control Systems Resilience. Studies in Systems, Decision and Control*, 255:175–196, 2020. doi:10.1007/978-3-030-31328-9_9.

Matthias Eckhart, Andreas Ekelhart, Arndt Lüder, Stefan Biffl, and Edgar Weippl. Security development lifecycle for cyber-physical production systems. In *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 3004–3011, New Jersey, 2019. IEEE. doi:10.1109/iecon.2019.8927590.

Jung Ho Eom, Nam Uk Kim, Sung Hwan Kim, and Tai Myoung Chung. Cyber military strategy for cyberspace superiority in cyber warfare. *Proceedings 2012 International Conference on Cyber Security, Cyber Warfare and Digital Forensic, CyberSec 2012*, pages 295–299, 2012. doi:10.1109/cybersec.2012.6246114.

Kelvin T Erickson. Programmable logic Controllers. *IEEE Potentials*, 15(1):14–17, 1996. ISSN 0278-6648. doi:10.1109/45.481370.

European Commission. The Directive on security of network and information systems (NIS Directive). https://ec.europa.eu/digital-single-market/en/network-and-information-security-nis-directive, 2019. URL https://ec.europa.eu/digital-single-market/en/network-and-information-security-nis-directive.

European Network and Information Security Agency (ENISA). Protecting Industrial Control Systems: Recommendations for Europe and Member States. https://www.enisa.europa.eu/publications/protecting-industrial-control-systems.-recommendations-for-europe-and-member-states, December 2011.

Dave Evans. The internet of everything: How more relevant and valuable connections will change the world. *Cisco Internet Business Solutions Group*, 2012.

M Fabro, E Gorski, N Spiers, J Diedrich, and D Kuipers. Recommended practice: improving industrial control system cybersecurity with defense-in-depth strategies. *DHS Industrial Control Systems Cyber Emergency Response Team*, 2016.

James P. Farwell and Rafal Rohozinski. Stuxnet and the future of cyber war. *Survival*, 53(1): 23–40, 2011. doi:10.1080/00396338.2011.555586.

Xuan Feng, Qiang Li, Haining Wang, and Limin Sun. Characterizing industrial control system devices on the internet. In *2016 IEEE 24th International Conference on Network Protocols (ICNP)*, pages 1–10. IEEE, 2016. doi:10.1109/ICNP.2016.7784407.

Pietro Ferretti, Marcello Pogliani, and Stefano Zanero. Characterizing background noise in ICS traffic through a set of low interaction honeypots. *Proceedings of the ACM Conference on Computer and Communications Security*, pages 51–61, 2019. doi:10.1145/3338499.3357361.

FireEye. Endpoint Security Software and Solutions. https://www.fireeye.com/solutions/hx-endpoint-security-products.html, n.d.

Oliver Fitton. Cyber Operations and Gray Zones: Challenges for NATO. *Connections: The Quarterly Journal*, 15(2):109–119, 2016. doi:10.11610/connections.15.2.08.

Sarah Fluchs. The Top 20 Secure PLC Coding Practices Project. https://bit.ly/34DqoHi, 2020.

Fortinet. FortiGate Intrusion Prevention System (IPS). https://www.fortinet.com/products/ips, n.d.

Selena Frye. Infographic: The life cycle of a server. https://www.techrepublic.com/blog/data-center/infographic-the-life-cycle-of-a-server/, 2013.

Justin K Gallenstein. Integration of the network and application layers of automatically-configured programmable logic controller honeypots. Technical report, Air Force Institute of Technology, 2017.

Jim Garamone. Esper Describes DOD's Increased Cyber Offensive Strategy. https://www.defense.gov/explore/story/Article/1966758/esper-describes-dods-increased-cyber-offensive-strategy/, 2019.

Joseph Gardiner, Barnaby Craggs, Benjamin Green, and Awais Rashid. Oops i did it again: Further adventures in the land of ics security testbeds. In *Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy*, pages 75–86, 2019. doi:10.1145/3338499.3357355.

Joseph Gardiner, Adam Eiffert, Peter Garraghan, Nicholas Race, Shishir Nagaraja, and Awais Rashid. Controller-in-the-middle: Attacks on software defined networks in industrial control systems. In *Proceedings of the 2th Workshop on CPS&IoT Security and Privacy*, pages 63–68, 2021. doi:10.1145/3462633.3483979.

Dennis J Gaushell and Henry T Darlington. Supervisory control and data acquisition. *Proceedings of the IEEE*, 75(12):1645–1658, 1987. doi:10.1109/PROC.1987.13932.

GCHQ. New National Cyber Security Centre becomes operational. https://www.gchq.gov.uk/news/new-national-cyber-security-centre-becomes-operational, 2016.

Hamid Reza Ghaeini and Nils Ole Tippenhauer. Hamids: Hierarchical monitoring intrusion detection system for industrial control systems. In *Proceedings of the 2nd ACM Workshop on Cyber-Physical Systems Security and Privacy*, pages 103–111, 2016. doi:10.1145/2994487.2994492.

Ahmed Ghappour. Searching places unknown: Law enforcement jurisdiction on the dark web. *Stanford Law Review*, 69:1075, 2017.

Danielle Gonzalez, Fawaz Alhenaki, and Mehdi Mirakhorli. Architectural security weaknesses in industrial control systems (ics) an empirical study based on disclosed software vulnerabilities. In *2019 IEEE International Conference on Software Architecture (ICSA)*, pages 31–40. IEEE, 2019. doi:10.1109/ICSA.2019.00012.

Benjamin Green, Daniel Prince, Jerry Busby, and David Hutchison. The impact of social engineering on industrial control system security. In *Proceedings of the First ACM Workshop on Cyber-Physical Systems-Security and/or PrivaCy*, pages 23–29, 2015. doi:10.1145/2808705.2808717.

Benjamin Green, David Hutchison, Sylvain Andre Francis Frey, and Awais Rashid. Testbed diversity as a fundamental principle for effective ICS security research. *Serecin*, 2016.

Benjamin Green, Marina Krotofil, and Ali Abbasi. On the Significance of Process Comprehension for Conducting Targeted ICS Attacks. In *Proceedings of the 3nd ACM Workshop on Cyber-Physical Systems Security and Privacy*, pages 57–67, New York, 2017. ACM. doi:10.1145/3140241.3140254.

Benjamin Green, Richard Derbyshire, William Knowles, James Boorman, Pierre Ciholas, Daniel Prince, and David Hutchison. {ICS} testbed tetris: Practical building blocks towards a cyber security resource. In *13th USENIX Workshop on Cyber Security Experimentation and Test (CSET 20)*, 2020.

Benjamin Green, Richard Derbyshire, Marina Krotofil, William Knowles, Daniel Prince, and Neeraj Suri. Pcaad: Towards automated determination and exploitation of industrial systems. *Computers & Security*, 110:102424, 2021. doi:10.1016/j.cose.2021.102424.

Greenbone. Openvas - open vulnerability assessment scanner. https://www.openvas.org/, 2022.

Jeremy Greenwood. *The third industrial revolution: Technology, productivity, and income inequality*. Number 435. American Enterprise Institute, 1997. ISBN 978-0-8447-7093-2.

Haoran Gu, Yingxu Lai, Yipeng Wang, Jing Liu, Motong Sun, and Beifeng Mao. Deids: a novel intrusion detection system for industrial control systems. *Neural Computing and Applications*, pages 1–18, 2022. doi:10.1007/s00521-022-06965-4.

Adam Hahn. Operational technology and information technology in industrial control systems. *Cyber-security of SCADA and other industrial control systems*, pages 51–68, 2016. doi:10.1007/978-3-319-32125-7_4.

Robert J Hall. An internet of drones. *IEEE Internet Computing*, 20(3):68–73, 2016. doi:10.1109/mic.2016.59.

Michael Haney and Mauricio Papa. A framework for the design and deployment of a SCADA Honeynet. *ACM International Conference Proceeding Series*, pages 121–124, 2014. doi:10.1145/2602087.2602110.

Shane Harris. *@ War: The rise of the military-internet complex*. Houghton Mifflin Harcourt, 2014. ISBN 978-0544570283.

Wayne Harrop and Ashley Matteson. Cyber resilience: A review of critical national infrastructure and cyber-security protection measures applied in the uk and usa. In *Current and Emerging Trends in Cyber Operations*, pages 149–166. Springer, 2015. doi:10.1057/9781137455550_10.

Vahid Hassanpour, Sedighe Rajabi, Zeinab Shayan, Zahra Hafezi, and Mohammad Mehdi Arefi. Low-cost home automation using arduino and modbus protocol. In *2017 5th International Conference on Control, Instrumentation, and Automation (ICCIA)*, pages 284–289. IEEE, 2017. doi:10.1109/ICCIAutom.2017.8258694.

Stephanie Hepburn and Rita J Simon. *Human trafficking around the world: Hidden in plain sight*. Columbia University Press, 2013. ISBN 9780231161459.

Stephen Hilt, Federico Maggi, Charles Perine, Lord Remorin, Martin Rösler, and Rainer Vosseler. Caught in the Act : Running a Realistic Factory Honeypot to Capture Real Threats. https://documents.trendmicro.com/assets/white_papers/wp-caught-in-the-act-running-a-realistic-factory-honeypot-to-capture-real-threats.pdf, 2020.

T Holczer, M Félegyházi, and L Buttyán. The design and implementation of a plc honeypot for detecting cyber attacks against industrial control systems. In *Proceedings of International Conference on Computer Security in a Nuclear World: Expert Discussion and Exchange*, 2015.

Thorsten Holz and Frederic Raynal. Detecting honeypots and other suspicious environments. In *Proceedings from the sixth annual IEEE SMC information assurance workshop*, pages 29–36. IEEE, 2005. doi:10.1109/IAW.2005.1495930.

Jason Hong. The state of phishing attacks. *Communications of the ACM*, 55(1):74–81, 2012. doi:10.1145/2063176.2063197.

Yan Hu, An Yang, Hong Li, Yuyan Sun, and Limin Sun. A survey of intrusion detection on industrial control systems. *International Journal of Distributed Sensor Networks*, 14(8): 1550147718794615, 2018. doi:10.1177/1550147718794615.

Philip Hunter. Regular patching cycles. *Computer Fraud and Security*, 2006(3):6–7, 2006. doi:10.1016/S1361-3723(06)70319-0.

Hvistendahl, Mara. China's Hacker Army. https://foreignpolicy.com/2010/03/03/chinas-hacker-army/, 2010.

IEC. IEC 62443-2-1:2010. https://webstore.iec.ch/publication/7030, 2010.

IEC. IEC 62443. https://www.isa.org/standards-and-publications/isa-standards/isa-iec-62443-series-of-standards, 2019.

INCIBE. Protocols and network security in ICS infrastructures. https://www.incibe-cert.es/en/publications/guides/ics-network-security, 2017.

INCIBE. Industrial honeypot implementation guide. https://www.incibe-cert.es/en/blog/industrial-honeypot-implementation-guide, 2019.

Industrial Cyber Security Center and Kaspersky Lab. Establishing Zones. https://www.cci-es.org/documents/10694/613683/ESTABLISHING+ZONES+AND+CONDUITS.pdf/abebfe81-305f-4be6-9625-0cb86dacf5db, 2019.

International Electrotechnical Commission. Find out more about IEC61850. https://iec61850.dvl.iec.ch/, n.d.

ISO and IEC. ISO/IEC 31010 : 2019 BSI Standards Publication Risk management – Risk assessment. 2019.

ISO/IEC. BS EN ISO/IEC 27001. https://www.iso.org/standard/27001, 2013.

ISO/IEC. BS EN ISO/IEC 27019:2017. https://www.iso.org/standard/68091.html, 2017.

Lech Janczewski and Andrew Colarik. *Cyber warfare and cyber terrorism*. Idea Group Inc, 2007. ISBN 978-1-59140-991-5.

Robert M Jaromin. Emulation of Industrial Control Field Device Protocols. pages 1–185, 2013.

Forough Ja'fari, Seyedakbar Mostafavi, Kiarash Mizanian, and Emad Jafari. An intelligent botnet blocking approach in software defined networks using honeypots. *Journal of Ambient Intelligence and Humanized Computing*, 12(2):2993–3016, 2021. doi:10.1007/s12652-020-02461-6.

Peter H. Jenney. ICS Software Protection. In Christopher Laing, Atta Badii, and Paul Vickers, editors, *Securing Critical Infrastructures and Critical Control Systems: Approaches for Threat Protection*, pages 217–239. IGI Global, Hershey, PA, 2013. doi:10.4018/978-1-4666-2659-1.ch009.

Arthur Jicha, Mark Patton, and Hsinchun Chen. SCADA honeypots: An in-depth analysis of Conpot. *IEEE International Conference on Intelligence and Security Informatics: Cybersecurity and Big Data, ISI 2016*, (May 2013):196–198, 2016. doi:10.1109/isi.2016.7745468.

Peng Jie and Liu Li. Industrial control system security. *Proceedings - 2011 3rd International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2011*, 2: 156–158, 2011. doi:10.1109/IHMSC.2011.108.

Joint Chiefs of Staff. Joint Publication 3-0: Joint Operations, JP 3-0. https://irp.fas.org/doddir/dod/jp3_0.pdf, 2017.

Joint Task Force Transformation initiative. Security and privacy controls for federal information systems and organizations. *NIST Special Publication*, 800(53):8–13, 2013. doi:10.6028/nist.sp.800-53r4.

Mohammad Karami and Damon McCoy. Understanding the emerging threat of ddos-as-a-service. In *Presented as part of the 6th USENIX Workshop on Large-Scale Exploits and Emergent Threats*, 2013.

Athina Karatzogianni. Blame it on the russians: Tracking the portrayal of russian hackers during cyber conflict incidents. In *Violence and War in Culture and the Media*, pages 237–262. Routledge, 2013. ISBN 9780203143308.

Eugene Kaspersky. The man who found stuxnet–sergey ulasen in the spotlight. *Nota Bene*, 2011.

Marian M. Kendrick and Zaki A. Rucker. Energy-Grid Threat Analysis Using Honeypots. https://apps.dtic.mil/dtic/tr/fulltext/u2/1080307.pdf, 2019.

Mohammad Taha Khan, Joe DeBlasio, Geoffrey M Voelker, Alex C Snoeren, Chris Kanich, and Narseo Vallina-Rodriguez. An empirical analysis of the commercial vpn ecosystem. In *Proceedings of the Internet Measurement Conference 2018*, pages 443–456, 2018. doi:doi.org/10.1145/3278532.3278570.

Rafiullah Khan, Peter Maynard, Kieran McLaughlin, David Laverty, and Sakir Sezer. Threat Analysis of BlackEnergy Malware for Synchrophasor based Real-time Control and Monitoring in Smart Grid. pages 53–63, 2016. doi:10.14236/ewic/ics2016.7.

Si-Jung Kim, Do-Eun Cho, and Sang-Soo Yeo. Secure model against apt in m-connected scada network. *International Journal of Distributed Sensor Networks*, 10(6):594652, 2014. doi:10.1155/2014/594652.

Todd Klessman, Mary Ellen Callahan, and Chief Privacy Officer. Chemical facility anti-terrorism standards (cfats) personnel surety program. 2011.

Eric D Knapp and Joel Thomas Langill. *Industrial Network Security: Securing critical infrastructure networks for smart grid, SCADA, and other Industrial Control Systems*. Syngress, 2014. ISBN 978-0124201149.

William Knowles, Daniel Prince, David Hutchison, Jules Ferdinand Pagna Disso, and Kevin Jones. A survey of cyber security management in industrial control systems. *International journal of critical infrastructure protection*, 9:52–80, 2015. doi:10.1016/j.ijcip.2015.02.002.

Sam Kottler, Mehdy Khayamy, Syed Rafay Hasan, and Omar Elkeelany. Formal verification of ladder logic programs using nusmv. In *SoutheastCon 2017*, pages 1–5, New Jersey, 2017. IEEE. doi:10.1109/secon.2017.7925390.

Neil Krawetz. Anti-Honeypot Technology. *IEEE Security & Privacy*, 2(1):76–79, 2004. doi:10.1109/msecp.2004.1264861.

D. T. Kuehl. From cyberspace to Cyberpower: Defining the Problem. In D. In Franklin, D. Kramer, H. S. Stuart, and K. W Larry, editors, *Cyberpower and national security*. 2009. doi:10.2307/j.ctt1djmhj1.

Stipe Kuman, Stjepan Gros, and Miljenko Mikuc. An experiment in using IMUNES and Conpot to emulate honeypot control networks. *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2017 - Proceedings*, pages 1262–1268, 2017. doi:10.23919/mipro.2017.7973617.

David Kushner. The real story of stuxnet. *IEEE Spectrum*, 50(3):48–53, 2013. doi:10.1109/mspec.2013.6471059.

Ralph Langner. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security and Privacy*, 9 (3):49–51, 2011. doi:10.1109/msp.2011.67.

Robert Larkin, Juan Lopez, and Jonathan Butts. Evaluation of traditional security solutions in the SCADA environment. *Proceedings of the 7th International Conference on Information Warfare and Security: ICIW*, pages 399–403, 2012.

Heiner Lasi, Peter Fettke, Hans Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business and Information Systems Engineering*, 6(4):239–242, 2014. doi:10.1007/s12599-014-0334-4.

Stephan Lau, Johannes Klick, Stephan Arndt, and Volker Roth. POSTER: Towards highly interactive honeypots for industrial control systems. *Proceedings of the ACM Conference on Computer and Communications Security*, 24-28-Octo:1823–1825, 2016. doi:10.1145/2976749.2989063.

Cheng Lei, Hong-Qi Zhang, Jing-Lei Tan, Yu-Chen Zhang, and Xiao-Hu Liu. Moving target defense techniques: A survey. *Security and Communication Networks*, 2018, 2018. doi:10.1155/2018/3759626.

Barry M Leiner, Vinton G Cerf, David D Clark, Robert E Kahn, Leonard Kleinrock, Daniel C Lynch, Jon Postel, Larry G Roberts, and Stephen Wolff. A brief history of the internet. *ACM SIGCOMM Computer Communication Review*, 39(5):22–31, 2009. doi:10.1145/1629607.1629613.

Shujun Li and Roland Schmitz. *A novel anti-phishing framework based on honeypots*. IEEE, 2009. doi:10.1109/ECRIME.2009.5342609.

Christina Schori Liang. Unveiling the" united cyber caliphate" and the birth of the e-terrorist. *Georgetown Journal of International Affairs*, pages 11–20, 2017. doi:10.1353/gia.2017.0032.

Herbert S Lin. Offensive cyber operations and the use of force. *J. Nat'l Sec. L. & Pol'y*, 4:63, 2010.

Samuel Litchfield, David Formby, Jonathan Rogers, Sakis Meliopoulos, and Raheem Beyah. Rethinking the honeypot for cyber-physical systems. *IEEE Internet Computing*, 20(5): 9–17, 2016. doi:10.1109/mic.2016.103.

Oscar Ljungkrantz and Knut Akesson. A study of industrial logic control programming using library components. In *2007 IEEE International Conference on Automation Science and Engineering*, pages 117–122, New Jersey, 2007. IEEE. ISBN 142441153X. doi:10.1109/coase.2007.4341783.

Efrén López-Morales, Carlos Rubio-Medrano, Adam Doupé, Yan Shoshitaishvili, Ruoyu Wang, Tiffany Bao, and Gail-Joon Ahn. Honeyplc: A next-generation honeypot for industrial control systems. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 279–291, 2020. doi:10.1145/3372297.3423356.

Peter Loshin. *Practical anonymity: Hiding in plain sight online*. Newnes, 2013. ISBN 9780124104426.

Stephen Lukasik. Why the arpanet was built. *IEEE Annals of the History of Computing*, 33 (3):4–21, 2010. doi:10.1109/mahc.2010.11.

Xupeng Luo, Qiao Yan, Mingde Wang, and Wenyao Huang. Using mtd and sdn-based honeypots to defend ddos attacks in iot. In *2019 Computing, Communications and IoT Applications (ComComAp)*, pages 392–395. IEEE, 2019. doi:10.1109/ComComAp46287.2019.9018775.

Sam Maesschalck, Vasileios Giotsas, and Nicholas Race. World wide ICS honeypots: A study into the deployment of conpot honeypots. In *Proceedings of the Seventh Annual Industrial Control System Security (ICSS) Workshop*, 2021. doi:10.1145/3462633.3483979.

Sam Maesschalck, Vasileios Giotsas, Benjamin Green, and Nicholas Race. Don't get stung, cover your ICS in honey: How do honeypots fit within industrial control system security. *Computers & Security*, 114, 2022. ISSN 0167-4048. doi:10.1016/j.cose.2021.102598.

Leandros A. Maglaras, Ki Hyung Kim, Helge Janicke, Mohamed Amine Ferrag, Stylianos Rallis, Pavlina Fragkou, Athanasios Maglaras, and Tiago J. Cruz. Cyber security of critical infrastructures. *ICT Express*, 4(1):42–45, 2018. doi:10.1016/j.icte.2018.02.001.

Ryan C. Maness and Brandon Valeriano. The Impact of Cyber Conflict on International Interactions. *Armed Forces & Society*, 42(2):301–323, 2016. doi:10.1177/0095327X15572997.

Angelos K. Marnerides, Vasileios Giotsas, and Troy Mursch. Identifying infected energy systems in the wild. *e-Energy 2019 - Proceedings of the 10th ACM International Conference on Future Energy Systems*, pages 263–267, 2019. doi:10.1145/3307772.3328305.

Daisuke Mashima, Binbin Chen, Prageeth Gunathilaka, and Edwin Lesmana Tjiong. Towards a grid-wide, high-fidelity electrical substation honeynet. *2017 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, (October):89–95, 2017. doi:10.1109/SmartGridComm.2017.8340689.

Daisuke Mashima, Yuan Li, and Binbin Chen. Who's scanning our smart grid? empirical study on honeypot data. In *2019 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2019. doi:10.1109/GLOBECOM38437.2019.9013835.

John Matherly. Complete guide to shodan. 2015.

Wataru Matsuda, Mariko Fujimoto, Tomomi Aoyama, and Takuho Mitsunaga. Cyber security risk assessment on industry 4.0 using ics testbed with ai and cloud. In *2019 IEEE conference on application, information and network security (AINS)*, pages 54–59. IEEE, 2019. doi:10.1109/AINS47559.2019.8968698.

Gary McGraw. Cyber war is inevitable (unless we build security in). *Journal of Strategic Studies*, 36(1):109–119, 2013. doi:10.1080/01402390.2012.742013.

Stephen McLaughlin, Charalambos Konstantinou, Xueyang Wang, Lucas Davi, Ahmad Reza Sadeghi, Michail Maniatakos, and Ramesh Karri. The Cybersecurity Landscape in Industrial Control Systems. *Proceedings of the IEEE*, 104(5):1039–1057, 2016. doi:10.1109/JPROC.2015.2512235.

Emma McMahon, Mark Patton, Sagar Samtani, and Hsinchun Chen. Benchmarking vulnerability assessment tools for enhanced cyber-physical system (cps) resiliency. In *2018 IEEE International Conference on Intelligence and Security Informatics (ISI)*, pages 100–105, New Jersey, 2018. IEEE. doi:10.1109/isi.2018.8587353.

Thomas Miller, Alexander Staves, Sam Maesschalck, Miriam Sturdee, and Benjamin Green. Looking back to look forward: Lessons learnt from cyber-attacks on industrial control systems. *International Journal of Critical Infrastructure Protection*, 35, 2021. doi:10.1016/j.ijcip.2021.100464.

Ariana Mirian, Zane Ma, David Adrian, Matthew Tischer, Thasphon Chuenchujit, Tim Yardley, Robin Berthier, Joshua Mason, Zakir Durumeric, J. Alex Halderman, and Michael Bailey. An Internet-wide view of ICS devices. *2016 14th Annual Conference on Privacy, Security and Trust, PST 2016*, pages 96–103, 2016. doi:10.1109/pst.2016.7906943.

MITRE. Att&ck for industrial control systems. https://attack.mitre.org/versions/v11/matrices/ics/, 2020.

MITRE. Unauthorized Command Message. https://collaborate.mitre.org/attackics/index.php/Technique/T0855, 2021.

Modbus. MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b3. https://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf, 2012.

Iyatiti Mokube and Michele Adams. Honeypots: Concepts, approaches, and challenges. *Proceedings of the Annual Southeast Conference*, 2007:321–326, 2007. doi:10.1145/1233341.1233399.

Joel Mokyr and Robert H Strotz. The second industrial revolution, 1870-1914. *Storia dell'economia Mondiale*, 21945(1), 1998.

Gijs Molenaar and Stephen Preeker. Welcome to python-snap7's documentation! https://python-snap7.readthedocs.io/en/latest/, 2013. URL https://python-snap7.readthedocs.io/en/latest/.

Charles More. *Understanding the industrial revolution*. Routledge, 2002. doi:10.4324/9780203136973.

Srinivas Mukkamala, Andrew Sung, and Ajith Abraham. Cyber security challenges: Designing efficient intrusion detection systems and antivirus tools. In V. Rao Vemuri, editor, *Enhancing Computer Security with Smart Technology*, pages 125–163. Auerbach, 2005. doi:10.1201/9780849330452.ch6.

MushMush Foundation. Installation on host using Virtualenv - Conpot 0.6.0 documentation. https://conpot.readthedocs.io/en/latest/installation/install.html, 2018.

Nitin Naik, Paul Jenkins, Roger Cooke, and Longzhi Yang. Honeypots that bite back: A fuzzy technique for identifying and inhibiting fingerprinting attacks on low interaction honeypots. In *2018 IEEE International Conference on fuzzy systems (FUZZ-IEEE)*, pages 1–8. IEEE, 2018. doi:10.1109/FUZZ-IEEE.2018.8491456.

Ellen Nakashima and Joby Warrick. Stuxnet was work of U.S. and Israeli experts, officials say. https://www.washingtonpost.com/world/national-security/stuxnet-was-work-of-us-and-israeli-experts-officials-say/2012/06/01/gJQAlnEy6U_story.html, 2012.

NATO. NATO - Official text: Brussels Summit Declaration issues by the Heads of State and Government participating in the meeting of the North Atlantic Council in Brussels. https://bit.ly/2XAW1S3, 2018.

Oscar Navarro, Servilio Alonso Joan Balbastre, and Stefan Beyer. Gathering intelligence through realistic industrial control system honeypots. In *International Conference on Critical Information Infrastructures Security*, pages 143–153. Springer, 2018. doi:10.1007/978-3-030-05849-4_11.

NCSC. Risk Management Guidance. https://www.ncsc.gov.uk/collection/risk-management-collection, 2018.

NCSC. Cyber Assessment Framework. https://www.ncsc.gov.uk/collection/caf/cyber-assessment-framework, 2019.

Lee Neitzel and Bob Huba. Top ten differences between ICS and IT cybersecurity. *InTech Magazine*, (June), 2014.

Lily Hay Newman. What israel's strike on hamas hackers means for cyberwar. https://bit.ly/3tuRKdL, 2019.

NIST. NIST Special Publication 800-30 Revision 1 - Guide for Conducting Risk Assessments. *NIST Special Publication*, 2012.

Alexander Nochvay. CODESYS Runtime, a PLC control framework. Technical report, Kaspersky, 2019.

NXLog. Industrial Control System protocols. https://docs.nxlog.co/userguide/integrate/industrial-control-system-protocols.html, n.d.

Luciana Obregon. Secure Architecture for Industrial Control Systems. *SANS Institute Information Reading Room*, 2015.

ODVA. CIP Safety. https://www.odva.org/technology-standards/distinct-cip-services/cip-safety/, n.d.

Office for Nuclear Regulation. Preparation for and Response to Cyber Security Events. https://www.onr.org.uk/operational/tech_asst_guides/cnss-tast-gd-7.5.pdf, 2021.

Official Shodan Twitter account. Update on the state of honeyscore. https://twitter.com/shodanhq/status/1311661444765806593?lang=en, 2020.

Philip O'Kane, Sakir Sezer, and Kieran McLaughlin. Obfuscation: The hidden malware. *IEEE Security & Privacy*, 9(5):41–47, 2011. doi:10.1109/MSP.2011.98.

Parmy Olson. *We are anonymous*. Random House, 2013. ISBN 978-0434022083.

ONR Security Assessment Principles (SyAPs). Office of nuclear regulation. http://www.onr.org.uk/syaps/, n.d.

OSSEC. Protocols and network security in ICS infrastructures. https://www.ossec.net/, n.d.

Paralax. Awesome Honeypots. https://github.com/paralax/awesome-honeypots, n.d.

Nicole Perlroth. Hackers Are Targeting Nuclear Facilities, Homeland Security Dept. and F.B.I. Say. https://www.nytimes.com/2017/07/06/technology/nuclear-plant-hack-report.html, 2017.

AB Robert Petrunić. Honeytokens as active defense. In *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1313–1317. IEEE, 2015. doi:10.1109/MIPRO.2015.7160478.

Pew Research Center. Public Sees U.S. Power Declining as Support for Global Engagement Slips. https://www.people-press.org/2013/12/03/public-sees-u-s-power-declining-as-support-for-global-engagement-slips/#top-threats, 2013.

R Piggin and I Buffey. Active defence using an operational technology honeypot. In *The 11th International Conference on System Safety*, London, 2016. doi:10.1049/cp.2016.0860.

PLCDev. Symbol Table Allowed Addresses and Data Types. http://www.plcdev.com/symbol_table_allowed_addresses_and_data_types, 2020.

Dimitrios Pliatsios, Panagiotis Sarigiannidis, Thanasis Liatifis, Konstantinos Rompolos, and Ilias Siniosoglou. A novel and interactive industrial control system honeypot for critical smart grid infrastructure. *IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, CAMAD*, 2019-Septe:1–6, 2019. doi:10.1109/CAMAD.2019.8858431.

Stanislav Ponomarev and Travis Atkison. Industrial control system network intrusion detection by telemetry analysis. *IEEE Transactions on Dependable and Secure Computing*, 13(2):252–260, 2015. doi:10.1109/TDSC.2015.2443793.

Stanislav Ponomarev and Travis Atkison. Industrial Control System Network Intrusion Detection by Telemetry Analysis. *IEEE Transactions on Dependable and Secure Computing*, 13(2):252–260, 2016. doi:10.1109/TDSC.2015.2443793.

Georgios Portokalidis and Herbert Bos. Sweetbait: Zero-hour worm detection and containment using low-and high-interaction honeypots. *Computer Networks*, 51(5):1256–1274, 2007. doi:10.1016/j.comnet.2006.09.005.

Portuguese National Cybersecurity Centre. *National Cybersecurity Framework*. 2020.

Niels Provos. Honeyd: A Virtual Honeypot Daemon. *Proceedings of the 10th DFNCERT Workshop*, pages 1–7, 2003.

Public Safety Canada. Industrial Control System (ICS) Cyber Security: Recommended Best Practices. 2012.

Steve Ranger. Security surprise: Four zero-days spotted in attacks on researchers' fake networks. https://www.zdnet.com/article/security-four-zero-day-attacks-spotted-in-attacks-against-honeypot-systems/, 2020.

Awais Rashid, Joseph Gardiner, Benjamin Green, and Barnaby Craggs. Everything is awesome! or is it? cyber security risks in critical infrastructure. In *International Conference on Critical Information Infrastructures Security*, pages 3–17. Springer, 2019. doi:10.1007/978-3-030-37670-3_1.

Sandeep Gogineni Ravindrababu and Jim Alves-Foss. Automated detection of configured sdn security policies for ics networks. In *Sixth Annual Industrial Control System Security (ICSS) Workshop*, pages 31–38, 2020. doi:10.1145/3442144.3442148.

Owen Redwood, Joshua Lawrence, and Mike Burmester. A Symbolic Honeynet Framework for SCADA System Threat Intelligence. In *ICCIP 2015: Critical Infrastructure Protection IX*, volume 466, pages 103–118, 2015. ISBN 9783319265667. doi:10.1007/978-3-319-26567-4_7.

Jeremy Richmond. Evolving battlefields: Does stuxnet demonstrate a need for modifications to the law of armed conflict. *Fordham Int'l LJ*, 35:842, 2011.

Thomas Rid. Cyber war will not take place. *Journal of strategic studies*, 35(1):5–32, 2012. doi:10.1080/01402390.2011.608939.

Thomas Rid and Ben Buchanan. Attributing cyber attacks. *Journal of Strategic Studies*, 38 (1-2):4–37, 2015. doi:10.1080/01402390.2014.977382.

Jamie Riden. Using Nepenthes Honeypots to Detect Common Malware. https://www.symantec.com/connect/articles/using-nepenthes-honeypots-detect-common-malware, 2006.

Mohammad Riftadi and Fernando Kuipers. P4i/o: Intent-based networking with p4. In *2019 IEEE Conference on Network Softwarization (NetSoft)*, pages 438–443. IEEE, 2019. doi:10.1109/NETSOFT.2019.8806662.

M Riley, JA Dlouhy, and B Gruley. Russians are suspects in nuclear site hackings, sources say. *Bloomberg*, 2017.

Rae Ritchie. Maersk: Springing back from a catastrophic cyber-attack. https://www.i-cio.com/management/insight/item/maersk-springing-back-from-a-catastrophic-cyber-attack, August 2019.

Andres Robles-Durazno, Naghmeh Moradpoor, James McWhinnie, Gordon Russell, and Inaki Maneru-Marin. PLC memory attack detection and response in a clean water supply system. *International Journal of Critical Infrastructure Protection*, 26:100300, 2019. ISSN 1874-5482. doi:10.1016/j.ijcip.2019.05.003.

Nicholas AM Rodger. *The safeguard of the sea: a naval history of Britain 660-1649*. Penguin UK, 2004. ISBN 978-0140297249.

William J Ronan. English and american courts and the definition of war. *Am. J. Int'l L.*, 31: 642, 1937. doi:10.2307/2190674.

Clifford Rosborough, Colin Gordon, and Brian Waldron. All About Eve: Comparing DNP3 Secure Authentication With Standard Security Technologies for SCADA Communications. 2019.

Ron Ross, Victoria Pillitteri, Richard Graubart, Deborah Bodeau, and Rosalie McQuaid. Developing cyber resilient systems: A systems security engineering approach. Technical report, National Institute of Standards and Technology, 2019.

Neil C Rowe, Binh T Duong, and E John Custy. Fake honeypots: A defensive tactic for cyberspace. In *Proc. of the IEEE Workshop on Information Assurance*, pages 223–230, 2006. doi:10.1109/IAW.2006.1652099.

Neil C. Rowe, Thuy D. Nguyen, Marian M. Kendrick, Zaki A. Rucker, Dahae Hyun, and Justin C. Brown. Creating effective industrial-control-system honeypots. *American Journal of Management*, 20(2):112–123, 2020.

Fabio Rugge. "mind hacking": Information warfare in the cyber age. *ISPI, January*, 20(319): 1–8, 2018.

Lilly Ryan. Defence in depth. https://openpracticelibrary.com/practice/defence-in-depth/, 2022.

Viktor Schiffer, DJ Vangompel, and R Voss. The common industrial protocol (cip) and the family of cip networks. *Milwaukee, Wisconsin, USA, ODVA*, 2006. doi:10.1201/b17365-10.

Klaus Schwab. *The fourth industrial revolution*. Currency, 2017. ISBN 978-0241300756.

Justin Searle. Plcscan. https://github.com/meeas/plcscan, 2015. Last accessed: 21 March 2022.

Martin A Sehr, Marten Lohstroh, Matthew Weber, Ines Ugalde, Martin Witte, Joerg Neidig, Stephan Hoeme, Mehrdad Niknami, and Edward A Lee. Programmable logic controllers in the context of industry 4.0. *IEEE Transactions on Industrial Informatics*, 17(5):3523–3533, 2020. doi:10.1109/TII.2020.3007764.

Alexandru Vlad Serbanescu, Sebastian Obermeier, and Der Yeuan Yu. A flexible architecture for Industrial Control System honeypots. *SECRYPT 2015 - 12th International Conference on Security and Cryptography, Proceedings; Part of 12th International Joint Conference on e-Business and Telecommunications, ICETE 2015*, 04:16–26, 2015. doi:10.5220/0005522500160026.

Abraham Serhane, Mohamad Raad, Raad Raad, and Willy Susilo. Plc code-level vulnerabilities. In *2018 International Conference on Computer and Applications (ICCA)*, pages 348–352. IEEE, 2018. doi:10.1109/comapp.2018.8460287.

Abraham Serhane, Mohamad Raad, Raad Raad, and Willy Susilo. Programmable logic controllers based systems (plc-bs): vulnerabilities and threats. *SN Applied Sciences*, 1(8): 924, 2019. doi:10.1007/s42452-019-0860-2.

Dimitrios Serpanos and Theodoros Komninos. The cyberwarfare in ukraine. *Computer*, 55 (7):88–91, 2022. doi:10.1109/MC.2022.3170644.

Sajjan Shiva, Sankardas Roy, and Dipankar Dasgupta. Game theory for cyber security. *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence*, 2010. doi:10.1145/1852666.1852704.

Shodan. Industrial Control Systems. https://bit.ly/3Jh3kAG, 2020.

Shodan. Honeypot or not? https://honeyscore.shodan.io/, n.d.a.

Shodan. Industrial control systems. https://www.shodan.io/explore/category/industrial-control-systems, n.d.b.

Siemens. Library of general functions (LGF) for SIMATIC STEP 7 (TIA Portal) and SIMATIC S7-1200 / S7-1500. https://sie.ag/3aeTFKz, 2019a. URL https://sie.ag/3aeTFKz.

Siemens. Ipv6 in automation technology. https://assets.new.siemens.com/siemens/assets/api/uuid:67742ab6-e541-48fa-bf89-6a4994368a9b/version:1560767181/whitepaper-ipv6-6zb5530-0dh02-0ba0-en.pdf, 2019b.

Siemens. Totally Integrated Automation Portal. https://new.siemens.com/global/en/products/automation/industry software/automation-software/tia-portal.html, 2020a.

Siemens. SIMATIC S7-300 - Proven Multiple Times! https://sie.ag/3ol428k, 2020b.

Siemens. SIMATIC WinCC V7. https://new.siemens.com/global/en/products/automation/industry-software/automation-software/scada/simatic-wincc-v7.html, 2020c.

Siemens. Machine level hmi. https://new.siemens.com/global/en/products/automation/simatic-hmi/panels.html, 2022. Last accessed: 21 March 2022.

Johan Sigholm. Non-state actors in cyberspace operations. *Journal of Military Studies*, 4(1): 1–37, 2013. doi:10.1515/jms-2016-0184.

Paulo Simões, Tiago Cruz, Jorge Gomes, and Edmundo Monteiro. On the use of honeypots for detecting cyber attacks on industrial control networks. In *Proc. 12th Eur. Conf. Inform. Warfare Secur. ECIW*, volume 2013, 2013. ISBN 9781909507340.

Paulo Simões, Tiago Cruz, Jorge Proença, and Edmundo Monteiro. Specialized honeypots for scada systems. In *Cyber Security: Analytics, Technology and Automation*, pages 251–269. Springer, 2015. doi:10.1007/978-3-319-18302-2_16.

Skhomenko, Andrey. Andrey skhomenko on twitter. https://twitter.com/JohnEskimSmith/status/1311550364823556096, 2020.

Miroslaw Skrzewski. Network Malware Activity – A Viewfrom Honeypot Systems. In A. Kwiecień, P. Gaj, and P Stera, editors, *Computer Networks*, pages 198–206. Springer, Berlin, 2012. ISBN 9783642312168. doi:10.1007/978-3-642-31217-5_22.

Clifton L Smith. Understanding concepts in the defence in depth strategy. In *IEEE 37th Annual 2003 International Carnahan Conference onSecurity Technology, 2003. Proceedings.*, pages 8–16. IEEE, 2003. doi:10.1109/CCST.2003.1297528.

Tomas Sochor and Matej Zuzcak. High-interaction linux honeypot architecture in recent perspective. In *International Conference on Computer Networks*, pages 118–131. Springer, 2016. doi:10.1007/978-3-319-39207-3_11.

Lance Spitzner. *Honeypots: Tracking Hackers*. Addison Wesley, Reading, MA, 2002. ISBN 0321108957.

Lance Spitzner. The honeynet project: Trapping the hackers. *IEEE Security and Privacy*, 1 (2):15–23, 2003. doi:10.1109/msecp.2003.1193207.

Shreyas Srinivasa, Jens Myrup Pedersen, and Emmanouil Vasilomanolakis. Gotta catch'em all: a multistage framework for honeypot fingerprinting. *Digital Threats: Research and Practice*, 2021. doi:10.1145/3584976.

John Stone. Cyber war will take place! *Journal of Strategic Studies*, 36(1):101–108, 2013. doi:10.1080/01402390.2012.730485.

Keith Stouffer, Joe Falco, Karen Scarfone, et al. Guide to industrial control systems (ics) security. *NIST special publication*, 800(82), 2011. doi:10.6028/NIST.SP.800-82r2.

Keith Stouffer, Victoria Pillitteri, Suzanne Lightman, Marshall Abrams, and Adam Hahn. Guide to Industrial Control Systems (ICS) Security NIST Special Publication 800-82 Revision 2. *NIST Special Publication 800-82 rev 2*, pages 1–157, 2015.

Keith Stougger, Victoria Pillitteri, Suzanne Lightman, Marshall Abrams, and Adam Hahn. NIST Special Publication 800-82: Guide to Industrial Control Systems (ICS) Security, Revision 2. https://bit.ly/3lqq6Qu, 2015.

Yanbin Sun, Zhihong Tian, Mohan Li, Shen Su, Xiaojiang Du, and Mohsen Guizani. Honeypot identification in softwarized industrial cyber–physical systems. *IEEE Transactions on Industrial Informatics*, 17(8):5542–5551, 2020. doi:10.1109/TII.2020.3044576.

H. I. Sutton. Positions of two nato ships were falsified near russian black sea naval base. https://news.usni.org/2021/06/21/positions-of-two-nato-ships-were-falsified-near-russian-black-sea-naval-base, June 2021.

Tenable. Nessus Vulnerability Assessment. https://www.tenable.com/products/nessus, 2022.

The Economist. War in the fifth domain. https://www.economist.com/briefing/2010/07/01/war-in-the-fifth-domain, 2010.

The European Commission. Directive (EU) 2016/1148 of the European Parliament and of the Council of 6 July 2016 concerning measures for a high common level of security of network and information systems across the Union. *OJ L*, 194(19.7):2016, 2016.

The European Parliament and The Council of The European Union. The Directive on security of network and information systems (NIS Directive). https://ec.europa.eu/digital-single-market/en/network-and-information-security-nis-directive, 2019.

The Honeynet Project. *Know Your Enemy: Revealing the Security Tools, Tactics, and Motives of the Blackhat Community*. Addison-Wesley, 2001. ISBN 978-0201746136.

The Netherlands Ministry of Defence. Defensie Cyber Strategie 2018 – Investeren in digitale slagkracht Nederland. https://www.defensie.nl/downloads/publicaties/2018/11/12/defensie-cyber-strategie-2018, 2018.

The White House. Remarks by the President on Securing Our Nation's Cyber Infrastructure. https://obamawhitehouse.archives.gov/the-press-office/remarks-president-securing-our-nations-cyber-infrastructure, 2009.

The White House. Executive order on improving the nation's cybersecurity. https://www.whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity/, May 2021.

Jay Thom, Yash Shah, and Shamik Sengupta. Correlation of cyber threat intelligence data across global honeypots. In *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 0766–0772. IEEE, 2021. doi:10.1109/CCWC51732.2021.9376038.

ThreatStop. Black Energy. *Security Report*, 2016.

FUJITA Tomonori. Introduction to ryu sdn framework. *Open Networking Summit*, pages 1–14, 2013.

Michail Tsikerdekis, Sherali Zeadally, Amy Schlesener, and Nicolas Sklavos. Approaches for preventing honeypot detection and compromise. In *2018 Global Information Infrastructure and Networking Symposium (GIIS)*, pages 1–6. IEEE, 2018. doi:10.1109/giis.2018.8635603.

Akihiro Tsuchiya, Francisco Fraile, Ichiro Koshijima, Angel Ortiz, and Raul Poler. Software defined networking firewall for industry 4.0 manufacturing systems. *Journal of Industrial Engineering and Management (JIEM)*, 11(2):318–333, 2018. doi:10.3926/jiem.2534.

Yoshiharu Tsuzaki and Yasuo Okabe. Reactive configuration updating for intent-based networking. In *2017 International Conference on Information Networking (ICOIN)*, pages 97–102. IEEE, 2017. doi:10.1109/ICOIN.2017.7899484.

Andrea Tundis, Eric Marc Modo Nga, and Max Mühlhäuser. An exploratory analysis on the impact of shodan scanning tool on the network attacks. In *The 16th International Conference on Availability, Reliability and Security*, 2021. doi:10.1145/3465481.3469197.

US-CERT. ICS Alert (ICS-ALERT-14-281-01E) Ongoing Sophisticated Malware Campaign Compromising ICS (Update E). https://www.us-cert.gov/ics/alerts/ICS-ALERT-14-281-01B, 2016.

US Nuclear Regulatory Commission. *Cyber security programs for nuclear facilities*. US Nuclear Regulatory Commission, Office of Nuclear Regulatory Research, 2010.

Nikolaos Vakakis, Odysseas Nikolis, Dimosthenis Ioannidis, Konstantinos Votis, and Dimitrios Tzovaras. Cybersecurity in smes: The smart-home/office use case. In *2019 IEEE 24th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pages 1–7. IEEE, 2019. doi:10.1109/CAMAD.2019.8858471.

Emmanouil Vasilomanolakis, Shreyas Srinivasa, and Max Muhlhauser. Did you really hack a nuclear power plant? An industrial control mobile honeypot. *2015 IEEE Conference on Communications and NetworkSecurity, CNS 2015*, pages 729–730, 2015. doi:10.1109/cns.2015.7346907.

Emmanouil Vasilomanolakis, Shreyas Srinivasa, Carlos Garcia Cordero, and Max Mühlhäuser. Multi-stage attack detection and signature generation with ics honeypots. In *NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium*, pages 1227–1232. IEEE, 2016. doi:10.1109/NOMS.2016.7502992.

Adrian Venables, Siraj Ahmed Shaikh, and James Shuttleworth. A model for characterizing cyberpower. In *International Conference on Critical Infrastructure Protection*, pages 3–16. Springer, 2015. doi:10.1007/978-3-319-26567-4_1.

Amir Vera, Jamiel Lynch, and Christina Carrega. Someone tried to poison a florida city by hacking into the water treatment system, sheriff says. https://cnn.com/2021/02/08/us/oldsmar-florida-hack-water-poison, February 2021.

Alexander Vetterl and Richard Clayton. Bitter Harvest: Systematically Fingerprinting Low- and Medium-interaction Honeypots at Internet Scale. *12th USENIX Workshop on Offensive Technologies (WOOT 18)*, 2018.

Nick Von Tunzelmann. Historical coevolution of governance and technology in the industrial revolutions. *Structural Change and Economic Dynamics*, 14(4):365–384, 2003. doi:10.1016/S0954-349X(03)00029-8.

SM Wade. SCADA Honeynets: The attractiveness of honeypots as critical infrastructure security tools for the detection and analysis of advanced threats. *Iowa State University*, page 67, 2011.

Brandon Wang, Xiaoye Li, Leandro P. De Aguiar, Daniel S. Menasche, and Zubair Shafiq. Characterizing and modeling patching practices of industrial control systems. *SIGMETRICS 2017 Abstracts - Proceedings of the 2017 ACM SIGMETRICS / International Conference on Measurement and Modeling of Computer Systems*, 1(1):9, 2017. doi:10.1145/3084455.

Cliff Wang and Zhuo Lu. Cyber deception: Overview and the road ahead. *IEEE Security & Privacy*, 16(2):80–85, 2018. doi:10.1109/MSP.2018.1870866.

Yong Wang, Jinpeng Wei, and Karthik Vangury. Bring your own device security issues and challenges. *2014 IEEE 11th Consumer Communications and Networking Conference, CCNC 2014*, pages 80–85, 2014. doi:10.1109/ccnc.2014.6866552.

Zhiqiang Wang, Jianyi Zhang, Gefei Li, Qixu Liu, Yaping Chi, Tao Yang, and Wei Zhou. Honeynet construction based on intrusion detection. *ACM International Conference Proceeding Series*, 2019. doi:10.1145/3331453.3360983.

Haroon Wardak, Sami Zhioua, and Ahmad Almulhem. Plc access control: a security analysis. In *2016 World Congress on Industrial Control Systems Security (WCICSS)*, pages 1–6, New Jersey, 2016. IEEE. doi:10.1109/wcicss.2016.7882935.

Nathalie Weiler. Honeypots for distributed denial-of-service attacks. In *Proceedings. Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pages 109–114. IEEE, 2002. doi:10.1109/ENABL.2002.1029997.

Joe Weiss. Industrial control system (ics) cyber security for water and wastewater systems. In *Securing Water and Wastewater Systems*, pages 87–105. Springer, 2014. doi:10.1007/978-3-319-01092-2_3.

Lee J. Wells, Jaime A. Camelio, Christopher B. Williams, and Jules White. Cyber-physical security challenges in manufacturing systems. *Manufacturing Letters*, 2(2):74–77, 2014. doi:10.1016/j.mfglet.2014.01.005.

A. J. Widdowson and P. B. Goodliff. CHEAT, an approach to incorporating human factors in cyber security assessments. In *10th IET System Safety and Cyber-Security Conference*, pages 1–5, Bristol, 2015. doi:10.1049/cp.2015.0298.

Kyle Wilhoit and Stephen Hilt. The GasPot Experiment: Unexamined Perils in Using Gas-Tank-Monitoring Systems. https://documents.trendmicro.com/assets/wp/wp_the_gaspot_experiment.pdf, 2015.

Marcus Willett. Assessing cyber power. *Survival*, 61(1):85–90, 2019. doi:10.1080/00396338.2019.1569895.

Jessica Wolfendale. Defining war. *Soft War: The Ethics of Unarmed Conflict*, page 16, 2017. doi:10.1017/9781316450802.004.

Wenfeng Xia, Yonggang Wen, Chuan Heng Foh, Dusit Niyato, and Haiyong Xie. A survey on software-defined networking. *IEEE Communications Surveys & Tutorials*, 17(1):27–51, 2014. doi:10.1109/COMST.2014.2330903.

Hui Xu, Yangfan Zhou, Jiang Ming, and Michael Lyu. Layered obfuscation: a taxonomy of software obfuscation techniques for layered security. *Cybersecurity*, 3(1):1–18, 2020. doi:10.1186/s42400-020-00049-3.

Wei Xu, Yaodong Tao, and Xin Guan. The landscape of Industrial Control Systems (ICS) devices on the internet. *2018 International Conference on Cyber Situational Awareness, Data Analytics and Assessment, CyberSA 2018*, 2018. doi:10.1109/CyberSA.2018.8551422.

Tarun Yadav and Arvind Mallari Rao. Technical aspects of cyber kill chain. In *International symposium on security in computing and communication*, pages 438–452. Springer, 2015. doi:10.1007/978-3-319-22915-7_40.

Ka-Ping Yee. Aligning security and usability. *IEEE Se*, 2(5):48–55, 2004. doi:10.1109/MSP.2004.64.

Jianzhou You, Shichao Lv, Yichen Hao, Xuan Feng, Ming Zhou, and Limin Sun. Characterizing Internet-Scale ICS Automated Attacks Through Long-Term Honeypot Data. In J. Zhou, X. Luo, Q. Shen, and Z. Xu, editors, *Information and Communications Security*, volume 11999, pages 71–88. Springer, Cham, 2019. ISBN 9783030415785. doi:10.1007/978-3-030-41579-2_5.

Mohammad-Reza Zamiri-Gourabi, Ali Razmjoo Qalaei, and Babak Amin Azad. Gas what? i can see your gaspots. studying the fingerprintability of ics honeypots in the wild. In *Proceedings of the fifth annual industrial control system security (ICSS) workshop*, pages 30–37, 2019. doi:10.1145/3372318.3372322.

Eviatar Zerubavel. *Hidden in plain sight: The social structure of irrelevance*. Oxford University Press, 2015. ISBN 9780199366620. doi:10.1093/acprof:oso/9780199366606.001.0001.

Feng Zhang, Shijie Zhou, Zhiguang Qin, and Jinde Liu. Honeypot: A Supplemented Active Defense System for Network Security. *Parallel and Distributed Computing, Applications and Technologies, PDCAT Proceedings*, pages 231–235, 2003.

Chunhui Zhao and Sujuan Qin. A research for high interactive honepot based on industrial service. *2017 3rd IEEE International Conference on Computer and Communications, ICCC 2017*, 2018-Janua(June):2935–2939, 2018. doi:10.1109/CompComm.2017.8323069.

Ye Zhou. Chameleon: Towards adaptive honeypot for internet of things. In *Proceedings of the ACM Turing Celebration Conference-China*, pages 1–5, 2019. doi:10.1145/3321408.3321589.