ICS Testbed Tetris: Practical Building Blocks Towards a Cyber Security Resource

Benjamin Green, Richard Derbyshire, William Knowles, James Boorman, Pierre Ciholas, Daniel Prince, David Hutchison
School of Computing and Communications, Lancaster University, United Kingdom
{b.green2, r.derbyshire1, w.knowles, j.boorman, p.ciholas, d.prince, d.hutchison}@lancaster.ac.uk

Abstract

Cyber attacks on Critical National Infrastructure (CNI) can be hugely detrimental to society, notably via compromising Industrial Control Systems (ICS) that underpin core CNI functions. In order to explore in-depth ICS Cyber Security challenges, testbeds are an essential tool, avoiding the need to experiment exclusively on live systems. However, ICS testbed creation is a complex multidisciplinary challenge, with a plethora of conflicting requirements. This paper, based on over six years of ICS testbed research and development that spans multiple diverse applications, proposes a flexible high-level model that can be adopted to support ICS testbed development. This is complemented by a baseline set of practical implementation guidance incorporating related and emerging technologies. As a collective, the model and implementation guidance offers a go-to guide for a wide range of end-users. Furthermore, it provides a coherent foundational structure towards establishing an online "living" resource, which can be expanded over time through broader community engagement.

1 Introduction

Industrial Control Systems (ICS) underpin operational processes responsible for the safe continued operation of industrial facilities, some of which are classified as Critical National Infrastructure (CNI) (e.g. water and energy) [55].

To better understand the complex construct of ICSs, the Purdue Enterprise Reference Architecture (commonly referred to as the "Purdue Model") [123] is often employed to provide a high-level view of fundamental features, separated into distinct zones. Cisco's enhanced version of the Purdue Model breaks down ICSs into four zones: A *Safety Zone* to manage safety functions; a *Manufacturing Zone* to monitor, control, and automate operational processes; a *Demilitarised Zone* providing a buffer between the *Manufacturing Zone* and *Enterprise Zone*; and finally an *Enterprise Zone* providing IT services and access to operational data via the *Demilitarised Zone* [21].

Cyber attacks targeting ICS manufacturing zones can result in widespread CNI service disruption [41], with risk to human life amplified through the targeting of safety zones [48]. Therefore, attacks targeting ICSs, via increasing numbers of component vulnerabilities and exploits, presents a significant concern [50]. While this is recognised by ICS operators, with over three quarters surveyed believing attackers are very or quite likely to target core operations, only 23% are compliant with baseline security guidance/regulations [94].

Although legal requirements have been introduced as a mechanism to improve the cyber security posture of CNI, research is required towards the continued development of defensive techniques, providing a feedback loop into relevant supporting standards/guidelines [73]. However, the use of real-world facilities to conduct research is limited due to the impact that such activities could have on operational processes [40]. Therefore, the use of testbeds is required. We use the term "testbed" to encompass all forms of environmental replication, including simulated, emulated, and physically replicated environments.

ICS testbeds fulfil a wide range of requirements, including research, proof-of-concepting, cyber exercising, and threat intelligence, across varying degrees of size and complexity. Through existing academic and industry engagement in this space, we have designed and built multiple ICS testbed environments to support these requirements. While existing research, including that of our own [37, 39, 40, 80], offers insight into the design and build of testbeds, it does not provide a clear, high-level model, accompanied by options for practical implementation, as a single go-to guide spanning a range of requirements. This paper targets such deficiencies through the identification of fundamental design considerations, and the introduction of a flexible, high-level model for ICS cyber security testbed design.

In order to take our model and deliver a testbed that supports end-user requirements, we include a baseline set of practical implementation options with supporting references. However, as it is not possible to provide a complete overview of all technical implementation options within a single publication, an incomplete picture is formed that will lose value over time. Therefore, to bolster the supporting technical implementation surrounding our model, we propose an online "living" resource, used to aggregate all practical implementation options we have applied to date. Furthermore, this acts as a foundation towards the creation of a community space which can evolve over time, with input from others who have developed ICS testbeds, thus overcoming the limitations of a single publication.

The remainder of this paper is structured as follows. Section 2 discusses related work. Section 3 outlines design considerations. Section 4 posits our high-level testbed model. Section 5 provides baseline options for the practical implementation of our model. Section 6 details the concept of our "living" resource. Section 7 concludes the paper and discusses areas for future work.

2 Related Work

Related work within this space is typically formulated around three themes: surveys, theoretical concepts, and focused example implementations.

Our early work explored the field of ICS risk management [56]. A key finding highlighted challenges in applied methodologies, often lacking empirical evidence, described as a result of researchers' inability to access real-world data. This has resulted in the use of testbeds, a review of which revealed a broad mix of implementations. In a subsequent review of smart grid testbeds, Cintuglu et al., [17] identified one third were developed with a cyber security focus. Holm et al., [44] analysed multiple implementation approaches (virtualisation, simulation, emulation, and hardware) and device coverage. Where as Wu and Kobara [125] focused on challenges aligned to the use of simulation-based approaches alone. Finally, our more recent work examined the use of testbeds, their implementation, and evaluation processes, to extract a set of design considerations [6].

While surveys help us understand adopted approaches, they are presented at a high-level, offering little in the way of guidance to support practical testbed development. Work presenting conceptual viewpoints begins to address this issue. Christiansson and Luiijf [15] discussed the concept of a European ICS security testbed from multiple perspectives (strategy, potential problems, end requirements, etc.). Our foundational work [39, 80] defined key concepts for ICS testbed design, centralised around flexibility, credibility, reliability, and diversity. Design considerations have been discussed by Joonsoo et al., [54] for cyber exercise testbeds, and Maynard et al., for experimental testbeds [62]. These are comparable to our more recent work [40], extended by Craggs et al., [28] to produce a reference architecture. Alternative reference architectures have also been proposed by Kavallieratos et al., [52] and Giani et al., [38]. Rigid concepts based on how the authors have developed their own facilities can result in a lack of depth, and offer limited details for practical implementation. From our experience, a form of tunnel vision can emerge, resulting in discussion aligned to a single approach.

Work focusing on implementation typically aligns to specific approaches, also resulting in a narrow viewpoint [29]. Focusing on large-scale deployments are little more than aspirational for most, with insufficient detail for end-to-end replication [61]. The use of network diagrams, associated devices, protocols, etc., begin to bridge the gap, however they are highly rigid [11]. This is reflected in our previous work [37, 40], where details were provided within the context of lessons learnt and aligned to a network diagram, presenting a challenge if used as a sole reference.

The existing work described in this section is largely limited due to its focus on rigid theoretical concepts and highly focused implementations. The remainder of this paper will leverage existing resources one should consult prior to the use of our flexible, high-level model. Our model will be complemented with a set of baseline implementation options, and the introduction of an online living resource, aiding in the practical development of testbeds to meet end-user requirements. Collectively, this will address the challenges identified across existing work, and better support the security community towards the continued development and evolution of ICS testbeds.

3 Design Considerations

ICS testbed development can be complex. It is therefore essential to clearly define and follow a set of design considerations aligned to specific end-user requirements, while remaining mindful of how such requirements may change over time.

Existing work details a broad selection of design considerations [37, 39, 40, 62, 80]. However our recent survey [6] offers the most comprehensive overview to date. This work provides a set of characteristics (see Table 1) one should consider when outlining testbed objectives (TBO), architecture (TBA), and/or evaluation process (TBE).

Characteristic	TBO	TBA	TBE
Fidelity		✓	
Modularity	✓	✓	
Diversity		✓	
Interoperability		✓	
Monitoring and Logging	✓	✓	
Openness	✓	✓	
Scalability/Extensibility		✓	
Flexibility/Adaptability	✓	✓	
Repeatability/Reproducibility	✓	✓	✓
Measurability&Measurement Accuracy		✓	✓
Cost-effectiveness	✓	✓	✓
Isolation/Safe Execution	✓	✓	
Usability	✓	✓	
Complexity		✓	

Table 1: Testbed Characteristics

The value placed on each characteristic should be driven by end-user requirements. For example, the UK Office for Nuclear Regulation [111] provides a set of security assessment principles [110] operators must incorporate into their systems and processes. This involves a requirement for exercising established processes, using live scenarios where practical. Dependent upon exercise aims, developing a comprehensive testbed across each characteristic may not be necessary. For example, where the focus of an exercise is aligned to human response, a testbed's fidelity requirements would typically be lower than an exercise focusing on the use of security tooling.

As a research tool, the diversity of systems, for example, becomes less of a concern when undertaking research on devices from a single vendor [47]. Compared to monitoring and logging capability, of utmost importance providing data capture capabilities for further analysis [97].

Industry based proof-of-concept/system trials [81], as with testbeds for research purposes, can be highly focused. The implementation/testing of an intrusion detection/prevention system (IDS/IPS) supporting a selection of industrial protocols, for example. This could dictate a requirement for the diversity of equipment to generate network traffic, while increasing fidelity by avoiding simulation techniques.

Threat intelligence, as demonstrated by a recent Honeypot implementation [43], further highlights the importance of building upon relevant characteristics. This work focused heavily on fidelity, in an attempt to attract real-worlds threat actors. The addition of broad interoperability and modularity, for example, would not have provided significant benefits given this defined objective.

The characteristics listed here should be reviewed as an initial step in forming an understanding of testbed design requirements. The following section introduces our flexible, high-level model, used to provide a stepping stone from design requirements to practical implementation.

4 High-Level Model

Work on ICS testbeds to date has focused on the design and implementation of specific architectural or device-level constructs. While this provides value, it is often too rigid and narrow in focus to support broader goals, differing levels of technical expertise, varying budgets, etc. This section introduces our high-level model, abstracted from rigid architectures, much the same way as the Purdue Model abstracts from real-world ICS deployments.

Figure 1 provides a graphical representation of our model. As an abstract resource, its use is non-prescriptive, supporting the understanding and selection of appropriate features to deliver against testbed requirements. The following subsections describe each of the model's five layers. These layers are supported by a safety and security wraparound, representing a key requirement for any testbed, ensuring end-user safety is upheld, and the testbed is protected from unauthorised access and disruption. The Cyber Security Experiment Lifecycle [70] and its four core elements (*Design*, *Instantiation*, *Execution*, and *Analysis*) will be used to augment layer descriptions.

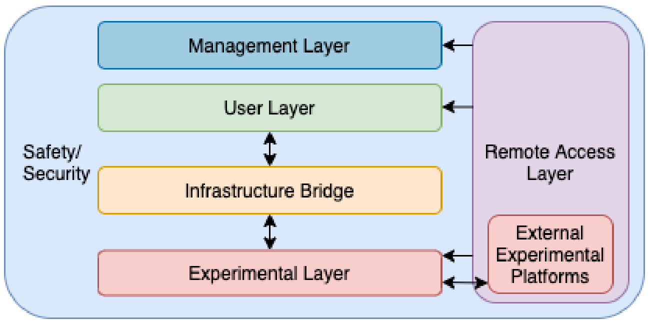

Figure 1: High-Level Model

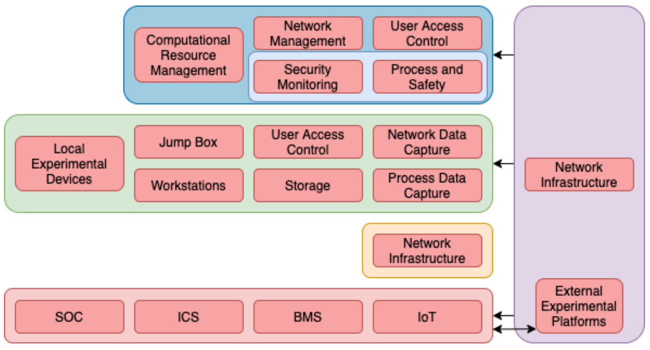

Figure 2: Model Breakdown

4.1 Management Layer

The *Management Layer* acts as central point from which oversight of a testbed can be established. This includes general functionality, from server management to network management, but extends to include the safety and security wraparound. Experiment *Instantiation* and *Execution* is partially covered by this layer. For example, management access to a hypervisor [122] can be used to deploy virtual machines within the *Experimental Layer* as per experiment *Design* requirements. If deployed in isolation from other layers as per our model, a clear demarcation between usable infrastructure and operational oversight can be established, meaning management activities are undertaken in complete isolation from experiment activities. However, this may not be practical in smaller deployments.

4.2 User Layer

The *User Layer* presents an opportunity to centralise and manage a set of resources, which testbed users may require to support experiment *Design*, and to capture and *Analyse* experiment data. The functionality of this layer can be viewed as the management of experiment tooling and systems with visibility over/access to the *Experimental Layer*. Therefore, experiment *Instantiation* and *Execution* are also supported by this layer through access into the *Experimental Layer* via the *Infrastructure Bridge*. For example, this could include accessing *Experimental Layer* devices, configuring, enabling, and orchestrating (*Executing*) their operation as per experiment *Design* requirements. This layer may take many forms, particularly in smaller deployments.

4.3 Infrastructure Bridge

The *Infrastructure Bridge* provides a network infrastructure to control the flow of data between the *User Layer* and the *Experimental Layer*, supporting experimental *Instantiation* and *Execution* from the *User Layer*. This can be of benefit when conducting high-risk research within the *Experimental Layer*, limiting the spread of malicious artefacts onto *User Layer* systems.

4.4 Experimental Layer

Forming the core element of any ICS testbed is the *Experimental Layer*, where all ICS components are deployed supporting experiment *Instantiation* and *Execution* as per associated *Design* requirements. Considering the Purdue Model as a high-level representation of ICS components, the *Experimental Layer* is where these are deployed. In addition to ICSs, we also consider related and emerging technologies encroaching into the ICS space. This includes Building Management Systems (BMS) and the Internet of Things (IoT). As this model is designed to support cyber security testbed design, additional related systems may also be required here, Security Operations Centres (SOC) for example.

4.5 Remote Access Layer

Based on the size and access requirements, a *Remote Access Layer* can be implemented accommodating interconnectivity for external *Experimental Layers* (e.g. partner organisations, mobile facilities, and remote worker simulation), external *User Layer* access, and remote *Management Layer* access, supporting all elements of the experiment lifecycle. Where one may participate in a larger federated set of testbeds between a group of partnering organisations, it could be that our model is singular, with remote external experimental platforms all managed within one holistic environment. Alternatively, each member may operate every layer of our model independently, and share *Experimental Layer* resources.

4.6 Summary

With the exception of the *Experimental Layer*, not all of the aforementioned model layers will be required within every testbed, much like each level of the Purdue Model and real-world ICS deployments. This is an important point to note, with the subsequent section applying our model as a foundation, and detailing specific attributes one can align to each layer should they be required.

To provide a bridge between our model and baseline practical implementation details, Figure 2 has been created to depict optional building blocks one may wish to consider implementing across each layer within the model. Here you can also see how the practical management of a safety and security wraparound falls within the scope of the management layer. As with the general management of core testbed elements, overarching safety and security properties should remain largely invisible to testbed users.

5 Baseline Practical Implementation

The following subsections take our model layers and constituent building blocks from Figure 2, to offer baseline practical implementation options. As discussed in Section 2, existing work providing details towards practical implementation techniques are highly focused on how the authors have built their own facilities. Here, we take existing academic work, commercial products, open-source tooling, programming libraries, personal experience, etc. to offer a broad range of practical implementation options, spanning all model layers.

As discussed across the previous sections, the practical implementation guidance provided here offers a baseline, and is structurally aligned the model breakdown in Figure 2. This provides a starting point towards the practical development of an ICS testbed, but more importantly, demonstrates the type of information an online living resource could contain, this will be discussed further in Section 6.

Due to the practical nature of this section, we have also created Figure 3 depicting an example network architecture aligned to our model. As with Figure 2, this is included to stimulate and support discussion. This is not a mandated practical application of the model. Additional practical network architectures for ICS environments can be found in documentation by NIST [107] and Kaspersky Labs [51].

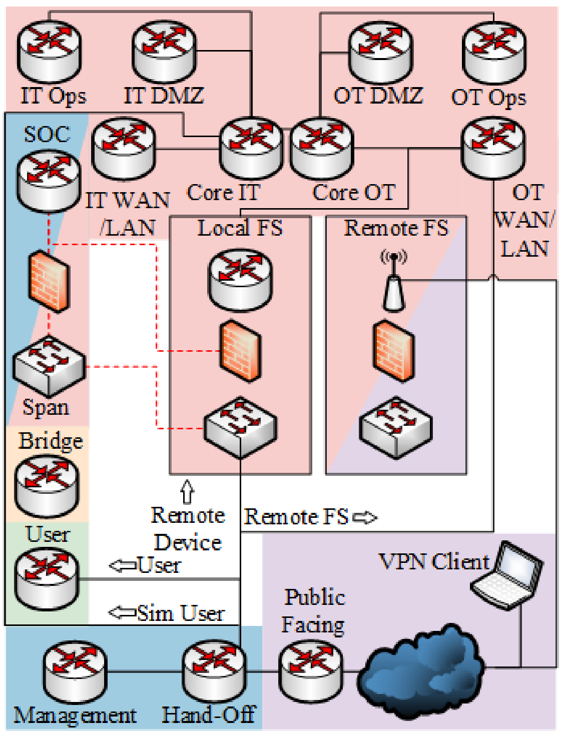

Figure 3: Example ICS Testbed Network Architecture

5.1 Management Layer

The following section details practical implementation techniques for elements within the *Management Layer*. When developing this layer it is important to consider all elements that underpin the testbed, and how they can be managed in isolation from other layers.

5.1.1 User Access Control

The *Management Layer* underpins all testbed capability. As such, user access control forms a critical component, allowing authorised testbed managers to make changes supporting user requirements at a physical and virtual level.

Lightweight Directory Access Protocol (LDAP) [127] via Microsoft's Active Directory (AD) [30], can be used as a centralised point for authentication with physical or virtualised computational resources [116], network devices [19], etc.

For physical access control, one can refer to guidance provided by CPNI [27].

5.1.2 Computational Resource Management

The implementation of desktop and server based instances, running across each layer of a testbed, can be physical or virtual. As per Figure 3, the management layer could be deployed as a single network zone with all physical devices connected allowing for wide-spread access.

Where virtualisation is adopted, a single physical host running numerous virtual hosts will be established. A dedicated network interface must be allocated for setup/ongoing management. Taking vSphere (ESXI) [122] as an example, should an environment require more than one physical host, vCentre [119] can be used to centrally manage all vSphere instances. vSphere and vCentre also provides APIs for the development of custom tooling [121].

Non-virtualised physical deployments may require a dedicated network interface, with an appropriate service enabling system-level access. This could be something as simple as RDP [63] or SSH [64].

Broader server and infrastructure monitoring tools including Splunk [105] may also provide benefits for larger deployments.

5.1.3 Network Resource Management

When accessing independent network devices, a dedicated network interface for use on the management layer could be adopted. Alternatively an IP to console server may be more cost-effective and prevent unwanted bridging [75].

The use of VLANs offers multiple benefits, particularly in virtualised environments [120]. Network connections can be passed via a centralised switch, collecting all VLANs and feeding them via a single trunk to a virtualisation server/s. The use of a centralised switch also provides a single point of contact to isolate network zones, supports the ability to switch between a physical and SDN based implementation [59], establish mirrored data feeds from specific ports [20], etc.

Building a digital twin of the network infrastructure using tools such as Packet Tracer [23] can support change management processes. In addition, general networking monitoring tools including Nagios [72] may also provide value.

We have also included a Hand-Off router in Figure 3. This could be used to manage all external connectivity, with well structured static routing [22], access control lists [24], intrusion detection/prevention (IDS/IPS) [13], etc. The separation between a public facing router managing encrypted external connectivity, and the non-encrypted internal network, allows for a greater level of visibility into traffic flows (IPs, ports, protocols, etc.) for a IDS/IPS to analyse.

5.1.4 Process and Safety

A set of devices and processes may be required to feed *Experimental Layer* systems with data. The use of these must be controlled, ensuring users are safe from physical harm at all times. Mandating testbed users undertake basic electrical safety training [108], follow wiring recommendations including power separation [88], and encase all live equipment in suitable panels [106], can act as a starting point supporting this critical requirements.

Out-of-loop safety systems, with a security wraparound (see Section 5.1.5) preventing unauthorised command execution between systems could also be employed, much the same as real-world safety systems [84, 85]. Additional formal guidance is available in related standards, including ISO 62061 [10].

The use of remote resources should be conducted under a strict set of procedures agreed between both parties. Cameras may also add value, providing a secondary verification of process states. This can be summarised as a holistic socio-technical view, with blended safety and security controls.

5.1.5 Security Monitoring

With the *Management Layer* acting as a gate-keeper to all testbed resources, security monitoring falls within its scope, ensuring testbed compromise by unauthorised persons is not realised, and experimentation is appropriately isolated. Concepts used in the development of a Security Operations Centre (SOC) within the *Experimental Layer* could be applied here (see Section 5.5.4).

5.2 Infrastructure Bridge Layer

This layer performs one task, supporting connectivity between the *User Layer*, and the *Experimental Layer*. Using Figure 3 as an example, this can be as simple as including one router and subnet onto which devices in both layers connect. In practical terms, this means a workstation in the *User Layer* will have two interfaces, one in the *User Layer* network, the other in the *Infrastructure Bridge Layer* network. Likewise, a workstation in the *Experimental Layer* will have two interfaces, one in an *Experimental Layer* network, the other in the *Infrastructure Bridge Layer* network. This is where *Experimental Layer* networks can be complex; it is essential that the devices communicate with one another over their respective experimental network subnets, rather than over the *Infrastructure Bridge* network. This can be managed through the introduction of local firewalls permitting connectivity with user layer devices only [65, 115]. Further complexity and additional networking components could be added to the *Infrastructure*

Bridge Layer, to manage *User Layer* access to *Experimental Layer* resources beyond the baseline described here.

5.3 User Layer

The following subsections provide a breakdown of each element within the *User Layer*. When developing this layer, it is important to consider all elements and tooling that testbed users may require to conduct their work, reducing the need for additional bolt-on systems and services which could impact the testbeds overall manageability.

5.3.1 Workstations

Including workstations (physical or virtual) in a *User Layer* allows for the integration of all supporting hardware and software as a centralised resource pool. These workstations could be Windows or Linux based, dependent upon requirements. For example, the Kali Linux distribution [96] comes pre-built with a number of tools that testbed users may require. Additional tools could include network analysis [124], scripting [87], reverse engineering [2], debugging [67], and ICS vendor software [102].

5.3.2 User Access Control

The use of physical and virtual workstations within the *User Layer* allows for scalability and the integration of local experimental devices (see Section 5.3.7), storage, data capture points, etc. User access control to these resources is critical, particularly where parallel activities may be under way across the *Experimental Layer*. The approaches described within Section 5.1.1 are applicable here, centralising the management of resources based on user profiles.

5.3.3 Jump Box

Jump boxes represent a central point from which system users access the *Experimental Layer* via the *Infrastructure Bridge Layer* (with a network interface sitting on each network as described in Section 5.2). Authentication with a jump box via approaches described in Section 5.1.1 would also be appropriate here.

5.3.4 Storage

The data generated by user experiments may require substantial storage capacity. While user workstations (physical or virtual) may have sufficient internal capacity, data management is key. The approaches described within Section 5.1.1 can be used to manage resources on a user-by-user basis. Additional storage can be implemented through the use of external devices, with platforms such as vSphere affording seamless integration of iSCSI devices [117] and network attached storage [118].

5.3.5 Network Data Capture

Network data capture is typically achieved through the use of network mirrors/spans. Section 5.1.3 described the use of a centralised network switch. Implementing an approach such as this, where every network connection passes through a centralised switch, allows all associated traffic to be captured. Where switches reside beyond the core network, within each field site for example (see Section 5.5.1), having them feed back to a centralised switch would also offer significant value. In order to provide ease of access to mirrored/span traffic, VLANS can be used. Aligning each link to a specific VLAN and trunking it to a centralised location, such as vSphere, allows for *User Layer* workstations to be mapped to specific VLANS, affording visibility of all associated traffic with appropriate tooling (e.g. Wireshark [124]). This supports a critical requirement of testbed users in the analysis of holistic network traffic. Furthermore, this traffic can also be fed into SOC systems (see Section 5.5.4).

5.3.6 Process Data Capture

The ability to access data from operational processes deployed across the *Experimental Layer* from the *User Layer*, may be required to validate results seen directly within the *Experimental Layer* itself. Using software defined radio (SDR) technologies, researchers can capture radio traffic from wireless sensor network devices (see Section 5.3.7). This can be used in collaboration with network data captures (see Section 5.3.5) to validate operational process sates. The use of cameras to monitor physical operational processes can also further validation efforts. In addition, a more comprehensive option would be to deploy secondary sensors and controllers within the *User Layer*, providing a guaranteed view of operational process states where *Experimental Layer* devices are under attack. Should an out-of-loop safety system be deployed as part of the *Management Layer* (see Section 5.1.4), exporting its data for use in the *User Layer* could present an alternate option.

5.3.7 Local Experimental Devices

In order to capture certain data types, or interact with aspects of the *Experimental Layer*, particularly with wireless technologies, testbed users will need to implement additional, specialised hardware. This could include SDR, WiFi, Bluetooth, and Infra-Red. For example, a Blade RF [74] could be deployed in the *User Layer* to support SDR capabilities.

5.4 Remote Access Layer

Figure 3 provides an example of integrating external connectivity with a testbed environment. Here a single, public facing router is depicted, which would be used to aggregate all external connectivity into the testbed. This router could act as a traditional, centralised VPN connection point for all external connectivity, adopting a "hub-and-spoke" architecture [18]. Alternatively, a single tunnel to a centralised platform such as Perimeter 81 [1] can be used to managed all external connectivity. In this model, Perimeter 81 acts as the public facing entry point into the testbed, supporting static VPN and desktop-based client connectivity models.

Figure 3 also includes a hand-off router; this can be used to introduce an additional layer between external and internal systems, with an open link between it and the public facing router (non-encrypted traffic that can be analysed by an IDS/IPS system). In this example, the hand-off would be responsible for access into each testbed layer. Access to the *Management Layer* represents the greatest risk, where an additional level of authentication between the hand-off and management router may be required. For external user access, directly reachable devices could be limited to *User Layer* workstations. *Experimental Layer* access can become more complicated dependent upon requirements. Figure 3 provides three example paths, one as a simulated user (mimicking a real-world operator of the ICS infrastructure), one as a remote field site (providing an entire external network of systems), and one as a single remote device being introduced into an existing area of the network.

5.5 Experimental Layer

The following subsections provide a breakdown of each element within the experimental layer. Figure 3 provides a more comprehensive picture of how the replicated ICS infrastructure could look, acting as an additional example reference point throughout this section.

5.5.1 ICS

When developing core ICS elements of the *Experimental Layer*, an appropriate network architecture should be selected, onto which appropriate devices and software packages can be deployed. Resources including NIST 800-82 [107] can be referenced for practical examples of ICS architectures found in real-world environments. An additional example of this can be seen in Figure 3, where a selection of network zones have been defined.

While each operational ICS deployment is unique, adopted hardware and software is similar, varying in scale and complexity to meet operational requirements. To support our discussion, examples are provided spanning core zones within the previously described "extended" Purdue Model [21], with the *Manufacturing Zone* and *Demilitarised Zone* grouped for simplicity. In Figure 3 the "Local/Remote FS" networks represent geographically dispersed field sites (water treatment works, pumping stations, etc.) that form the foundation of any *Manufacturing Zone*, the "Operational Technology (OT)" networks represent elements of the *Manufacturing Zone* and *Demilitarised Zone*, and the "Information Technology (IT)" networks represent elements of the *Enterprise Zone*.

Manufacturing Zone and Demilitarised Zone: The physical operational process resides at the lowest level of an ICS. There are two primary options for process replication, physical and virtual. A physical self-build can be conducted based on existing documented approaches [89]. Alternatively, two off-the-shelf options are available: large scale processes that operate in the same way as real-world systems [42], or mini replication kits [35]. Simulated options are available both commercially [32] and as open-sourced projects [36], or can be self developed through the use of communications libraries supporting common industrial protocols [71].

Programmable Logic Controllers (PLC) and Remote Terminal Units (RTU) represent the heart of manufacturing zone activities, providing direct interaction with operational processes delivering monitoring, control, and automation capability. Physical devices can be procured from leading vendors including Rockwell [91] . Some also offer starter kits with all relevant supporting configuration software and Human Machine Interfaces (HMIs) [99]. Alternatively, commercial PLC simulation software is available [103], or the aforementioned library [71] could be used to self develop a simulated controller, with the option of physical I/O through microcontroller development boards/single board computers [7].

HMIs used to monitor and control operational processes via PLCs are bundled within starter kits. They can also be procured separately in a range of sizes from leading vendors including Rockwell [90] . Within larger-scale deployments these devices are replaced with desktop-based software alternatives [101]. The aforementioned library could also be used to self develop a HMI [71].

Historian software used to collect data for analytical purposes is available from leading vendors, including Kepware [53] . Once again, the aforementioned libraries could be used to self develop a historian [71], with the addition of supplementary libraries supporting protocols for connectivity with datacentre/cloud based systems (e.g. OPC [79]).

When considering data-centre based environments aggregating data from multiple field sites, commercial solutions are available from leading vendors, including Schneider [31], that may also include demo licensing options. Open sourced options are also available, including RapidSCADA [93] . Moving into the IoT/IIoT space, products from Thingworx [86] offer additional benefits, such as cloud-based operation, and increased interconnectivity with a wider variety of devices, such as those noted in Section 5.5.3. Finally, as with the aforementioned examples, existing libraries can be used to self develop systems of this nature [71, 79].

In addition to the ICS focused packages described here, the inclusion of more generic network and user management systems discussed across previous sections could also be deployed. Security monitoring capability forms an additional key consideration, see Section 5.5.4 for related implementation details.

Enterprise Zone: The *Enterprise Zone* largely contains common IT-based system, from general user workstations, to email servers [68], file share servers [69], and update servers [66]. However, the inclusion of bespoke software packages to interface with *Demilitarised Zone* systems may also required. For example, a mirror of Schneider's ClearSCADA platform could be deployed in the *Demilitarised Zone* (isolated away from broader field-site connectivity utilised by a primary ClearSCADA server) to provide *Enterprise Zone* access to operational data. This platform can be accessed via a web-interface, or with a client application [31]. In addition, the inclusion of Internet connectivity within this zone would be common in real-world scenarios. Where direct connectivity to the Internet is less desirable, simulated alternatives are available [45]. As with the *Demilitarised Zone*, the inclusion of generic network, user management, and security systems discussed throughout this paper could also be deployed here.

5.5.2 BMS

BMSs are traditionally abstracted and divided into 3 zones: the *Field Zone*, the *Automation Zone*, and the *Management Zone* [16]. We use these three zones to breakdown discussions here. As previously described, BMSs are related to ICS, therefore may be of interest to testbed users. Both BMS and IoT systems could be deployed in a single isolated network zone, or become fully integrated into core ICS elements of the *Experimental Layer*.

Field Zone: This zone is where physical elements under the BMSs control reside. These include Heating Ventilation and Air Conditioning (HVAC), lighting, fire and smoke detection, and access control. Physical elements can be deployed as large-scale real-world systems [33], scaled down replications [34], or simulated using field-level communication protocol libraries [83] used to communicate with *Automation Zone* devices.

Automation Zone: Controllers form the core component of this zone, providing an interface to monitor, control, and automate operational processes, similar to PLCs and RTUs in ICSs. Physical devices can be procured from leading vendors such as Trend Controls [112]. Some ICS vendors also provide BMS-specific devices, including Siemens [100] . Alternatively controllers can be self developed using microcontroller development boards/single board computers [7], or virtualised using protocol libraries for communications between the *Field Zone* [83] and the *Management Zone* [9].

Management Zone: Data aggregation from all controllers can be centralised for supervision, monitoring, and logging purposes within this zone. Traditionally, this is developed as a supervisory server [113] with direct network access to controllers and a historian server [25]. Additional engineering workstations [26], HMIs, and backup servers may also feature in this zone dependent upon operational requirements. Supervision servers are often proprietary, this can lead to wider compatibility issues. To address this issue, and to better unify BMSs, Tridium's Niagara Fox [114] uses add-ons (drivers) to communicate with a broad range of vendor devices using both open-source and proprietary protocols.

5.5.3 IoT

The Internet of Things (IoT) is vastly heterogeneous [3], with devices and solutions being developed and deployed for seemingly every conceivable application [5]. There are many different architectures available for IoT, however for the purposes of implementation within the *Experimental Layer*, the following commonly utilised 3-tiered architecture [12] is used to separate and guide relevant discussion: *Perception Zone*, *Network Zone*, and *Application Zone*.

Perception Zone: This zone consists of devices used to sense and interact with their surrounding environments. Physical devices can be procured from leading vendors including Yale [126], or can be self developed using open source kits [77]. Commercial simulation options are also available [4].

Network Zone: IoT devices use a variety of network protocols. Devices that are computationally constrained use bespoke protocols [104], compared to gateways and less computationally constrained devices operating over standardised/conventional protocols [49]. There exist both commercial [98] and open source [78] solutions supporting a variety of IoT communication protocols. Simulation of these is also made possible through commercial products [109] and programming libraries [82].

Application Zone: This zone contains traditional and non-traditional devices aggregating and utilising generated data. Implementations available within this zone vary greatly, and are largely depend on the given application area. For example, there are both commercial [92] and open source [76] solutions available for home automation. In-house development of IoT solutions within this space is achievable, with libraries available to support communication using related IoT protocols [60]. Virtual application development and testing is also available commercially [46].

5.5.4 SOC

The Security Operations Centre (SOC), within the *Experimental Layer*, is intended to support attack detection, centralised security monitoring, and analysis during experimentation. Data ingested and analysed by the SOC can be split into two domains, endpoint and network. Endpoint data is captured from desktop and server based systems throughout the *Experimental Layer* via the use of installed agents, configured to feed back crucial information such as processes, communications, and user logins to the SOC's server. There are numerous options for endpoint monitoring and detection, such as AT&T's AlienVault Unified Security Management [8]. The inclusion of network data into the SOC allows for a more holistic view, and provides better coverage over devices on which endpoint agents cannot be installed. Network traffic, captured from network mirrors/spans in the *Experimental Layer* (see Section 5.3.5), can be fed into the existing SOC server to be parsed. However, one may have more success passing the network traffic to specialised devices capable of ICS, BMS, and IoT protocol recognition [14].

5.6 Layer Agnostic

Systems built with a diverse toolset can be deployed across multiple testbed layers to perform a range of tasks, from maintenance to experimentation. For example, a Kali Linux [96] system can provide adversary simulation in the *Experimental Layer*, capture and analyse network traffic in the *User*

Layer, and undertake routine offensive security testing of the testbed's own defences in the *Management Layer*. Access to ICS device configuration/programming software across testbed layers may also form a key requirement, and where licenses are tied to a single system, continued re-deployment of systems between layers becomes critical.

6 Living Resource

The previous sections have taken our high-level model and transformed it first into a set of building blocks (Figure 2), then baseline practical implementation guidance. While this provides a starting point for those looking to build/enhance their own testbed environments, due to page limitations it becomes infeasible to detail all options towards the practical development of a testbed. Furthermore, as a publication ages over time, the relevance and value of its associated practical guidance is devalued.

Using our model and the initially proposed set of constituent building blocks, Section 5's baseline implementation guidance set provides a clear foundational structure onto which additional information can be added over time. We therefore present the concept of an online "living" resource for ICS testbed development, currently hosted publicly on GitHub [57]. This resource is currently presented as a web page and is structured in alignment to Section 5, including introductory definitions of the model and its associated layers. Taking the practical guidance from Section 5 as a base, we have added additional details and references based on our own research and experience in testbed development. This demonstrates added value brought to the security community through a potentially limitless living resource, which has the ability to grow and evolve over time.

While the current living resource offers additional value in its current form, it is based on the research and experience of those within the Security Lancaster research institute. This presents a limiting factor. To overcome this limitation, we aim to build an open information sharing community to support its continued development. Initially, this will be through a dedicated mailbox to which other testbed owners can submit suggestions for new building blocks across each layer (an extension of Figure 2), and/or accompanying practical implementation guidance.

Security Lancaster is in the process of deploying TIDE-H (Threat Intelligence Data Exchange Hub) [58]. TIDE-H will act as a central repository for datasets spanning a range of security themes, one of which ties directly to ICSs, making it an ideal platform for a more interactive community to be housed (beyond a simple mailbox). Furthermore, it allows for the integration and sharing of additional ICS testbed artefacts (configuration files, network traffic captures, tooling, etc.).

The formation of TIDE-H also enables broader integration with other information sharing hubs, such as SEARCCH (due to launch in Summer 2020) [95], thus supporting the growth of a broader community-driven space within the context of cyber security research.

7 Conclusion and Future Work

ICS testbeds are fast becoming an essential tool for the in-depth exploration of cyber security challenges towards the protection of CNI. As an extension of existing work in this space, we have introduced a flexible high-level model used to support the design and development of ICS testbeds, accompanied by baseline practical implementation options. Collectively we have provided a set of building blocks the cyber security community can adopt to shape their own testbed environments.

In aligning discussions to our proposed model, we have illustrated its non-prescriptive application, similar to the Purdue model in real world deployments. Acting as a supporting structure, our model can be built upon based on end-user requirements. To further the value of our model and baseline practical implementation options, we have established an online living resource, initially offering additional practical implementation options based on our own research and experience within this field. This resource forms the foundations of a community-driven effort, where additional building blocks can be added to our high-level model, along with associated practical implementation guidance.

Since the creation of our model, we have rebuilt Lancaster University's ICS testbed to support each layer. This has improved the usability of our testbed, with lower barriers to entry for new users, and enabled a broader set of concurrent research activities to be undertaken. In addition, we are using the model to support testbed design for industry-driven product evaluation activities.

Future work will look to port our initial living online resource into the TIDE-H platform. Our vision is that this living resource provides a foundation towards the creation of a community-driven, evolving implementation guide, but more than that, allows for additional testbed artefacts and data repositories to be shared within the community. Furthermore, with community support comes the ability to further validate our model across a range of application domains.

Acknowledgements
This research was supported, in part, by EC H2020 CONCORDIA GA# 830927, and by the Security Lancaster Institute.

In addition, the authors are grateful to the anonymous reviewers, who have helped to realise the potential of this publication through their encouragement of our online living resource concept.

References

[1] Perimeter 81. Transforming Secure Access for the Remote Workforce. https://bit.ly/2ZpwiKQ, 2020.

[2] National Security Agency. GHIDRA. https://bit.ly/307Idw2, 2020.

[3] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of

Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys and Tutorials*, 17(4):2347–2376, 2015.

[4] Amazon. Amazon AWS IoT Device Simulator. https://amzn.to/3gSNFJG, 2019.

[5] Kenneth Li Minn Ang and Jasmine Kah Phooi Seng. Application Specific Internet of Things (ASIoTs): Taxonomy, Applications, Use Case and Future Directions. *IEEE Access*, 7:56577–56590, 2019.

[6] Uchenna D Ani, Jeremy M Watson, Benjamin Green, Barnaby Craggs, and Jason Nurse. Design considerations for building credible security testbeds: A systematic study of industrial control system use cases. *arXiv preprint arXiv:1911.01471*, 2019.

[7] Arduino. Arduino. https://bit.ly/38Vmw6c, 2020.

[8] AT&T. Endpoint Detection and Response (EDR) | AlienVault | AT&T Cybersecurity. http://soc.att.com/3ept8Lf, 2020.

[9] BACNet. BACNet: Developer help. https://bit.ly/3j3uyyw, 2020.

[10] BSI. 62061-1:2010: Guidance on the Application of ISO 13849-1 and IEC 62061 in the Design of Safety-Related Control Systems for Machinery. Technical report, 2010.

[11] R. Candell, T. Zimmerman, and K. Stouffer. An industrial control system cybersecurity performance testbed. *National Institute of Standards and Technology. NISTIR*, 8089, 2015.

[12] Yassine Chahid, Mohamed Benabdellah, and Abdelmalek Azizi. Internet of things security. *2017 International Conference on Wireless Technologies, Embedded and Intelligent Systems, WITS 2017*, 2017.

[13] Check Point. Intrusion Prevention System (IPS). https://bit.ly/30bczh0, 2020.

[14] Check Point. Product Trials. https://bit.ly/3esdqz5, 2020.

[15] H. Christiansson and E. Luiijf. Creating a european scada security testbed. In *International Conference on Critical Infrastructure Protection*, pages 237–247. Springer, 2007.

[16] Pierre Ciholas, Aidan Lennie, Parvin Sadigova, and Jose M Such. The security of smart buildings: a systematic literature review. *arXiv preprint arXiv:1901.05837*, 2019.

[17] M.H. Cintuglu, O.A. Mohammed, K. Akkaya, and A.S. Uluagac. A survey on smart grid cyber-physical system testbeds. *IEEE Communications Surveys & Tutorials*, 19(1):446–464, 2017.

[18] Cisco. Configuring a Hub-and-Spoke Site-to-Site VPN with Cisco ISA500 Series Security Appliances. https://bit.ly/2DAnKYY, 2011.

[19] Cisco. AAA LDAP Configuration Guide, Cisco IOS Release 15M&T. https://bit.ly/38V36ye, 2018.

[20] Cisco. Catalyst Switched Port Analyzer (SPAN) Configuration Example. https://bit.ly/3ez59JS, 2019.

[21] Cisco. Networking and Security in Industrial Automation Environments. https://bit.ly/3h0CA9s, 2019.

[22] Cisco. Chapter: Configuring Static Routing. https://bit.ly/38Tt5pU, 2020.

[23] Cisco. Cisco Packet Tracer. https://bit.ly/38V3tc6, 2020.

[24] Cisco. Configuring IP Access Lists. https://bit.ly/2CcysVb, 2020.

[25] Reliable Controls. RC-Archive. https://bit.ly/2CtpaE9, 2020.

[26] Reliable Controls. RC-Studio. https://bit.ly/2ZqYIUx, 2020.

[27] CPNI. Physical Security. https://bit.ly/2WibAe1, 2020.

[28] B. Craggs, A. Rashid, C. Hankin, R. Antrobus, O. Şerban, and N. Thapen. A reference architecture for iiot and industrial control systems testbeds. In *2nd Conference on Living in the Internet of Things 2019*, United Kingdom, 2018. Institution of Engineering and Technology (IET).

[29] C.M. Davis, J.E. Tate, H. Okhravi, C. Grier, T.J. Overbye, and D. Nicol. Scada cyber security testbed development. In *2006 38th North American Power Symposium*, pages 483–488. IEEE, 2006.

[30] Brian Desmond, Joe Richards, Robbie Allen, and Alistair G Lowe-Norris. *Active Directory: Designing, Deploying, and Running Active Directory*. " O'Reilly Media, Inc.", 2008.

[31] Schneider Electric. ClearSCADA. https://bit.ly/3emeqoo, 2020.

[32] factoryio. Next-Gen PLC Training: 3D Factory Simulation. https://bit.ly/3fw2C4q, 2020.

[33] Fischertechnik. Electro Air Products | Broughton EAP. https://bit.ly/3j1OORf, 2020.

[34] Fischertechnik. Starter Set for micro:bit - fischertechnik. https://bit.ly/2DGZaWw, 2020.

[35] fischertechnik. Training models. https://bit.ly/307LviT, 2020.

[36] David Formby, Milad Rad, and Raheem Beyah. Lowering the barriers to industrial control system security with {GRFICS}. In *2018 {USENIX} Workshop on Advances in Security Education ({ASE} 18)*, 2018.

[37] 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.

[38] Annarita Giani, Gabor Karsai, Tanya Roosta, Aakash Shah, Bruno Sinopoli, and Jon Wiley. A testbed for secure and robust scada systems. *ACM SIGBED Review*, 5(2):1–4, 2008.

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

[40] Benjamin Green, Anhtuan Lee, Rob Antrobus, Utz Roedig, David Hutchison, and Awais Rashid. Pains, gains and plcs: ten lessons from building an industrial control systems testbed for security research. In *10th {USENIX} Workshop on Cyber Security Experimentation and Test ({CSET} 17)*, 2017.

[41] A. Greenberg. 'crash override': The malware that took down a power grid. *URL http://bit.ly/2raojOf, Wired Magazine, retrieved*, pages 09–20, 2017.

[42] Gunt. Physical / chemical water treatment. `https://bit.ly/2OleiLl`, 2020.

[43] Stephen Hilt, Federico Maggi, Perine Charles, Remorin Lord, Martin Rösler, and Rainer Vosseler. Caught in the Act: Running a Realistic Factory Honeypot to Capture Real Threats. Technical report, 2020.

[44] H. Holm, M. Karresand, A. Vidström, and E. Westring. A survey of industrial control system testbeds. In *Nordic Conference on Secure IT Systems*, pages 11–26. Springer, 2015.

[45] Thomas Hungenberg and Matthias Eckert. About the INetSim project . `https://bit.ly/2ZrrPan`, 2020.

[46] IOTIFY. IoT Application Simulation. `https://bit.ly/32f5FtI`, 2018.

[47] William Jardine, Sylvain Frey, Benjamin Green, and Awais Rashid. Senami: Selective non-invasive active monitoring for ics intrusion detection. In *Proceedings of the 2nd ACM Workshop on Cyber-Physical Systems Security and Privacy*, pages 23–34, 2016.

[48] B. Johnson, D. Caban, M. Krotofil, D. Scali, N. Brubaker, and C. Glyer. Attackers Deploy New ICS Attack Framework "TRITON" and Cause Operational Disruption to Critical Infrastructure. `https://bit.ly/3fEsomM`, 2017.

[49] Juniper Networks. Understanding the IEEE 802.11 Standard for Wireless Networks. `https://juni.pr/3euchXR`, 2018.

[50] Kaspersky Lab ICS CERT. Threat Landscape for Industrial Automation Systems. Technical report, 2019.

[51] Kaspersky Labs. Establishing Zones and Conduits. Technical report, 2018.

[52] Georgios Kavallieratos, Sokratis K Katsikas, and Vasileios Gkioulos. Towards a cyber-physical range. In *Proceedings of the 5th on Cyber-Physical System Security Workshop*, pages 25–34, 2019.

[53] Kepware. KEPServerEX: Connects disparate devices and applications, from plant control systems to enterprise information systems. `https://bit.ly/2ZoLyrk`, 2020.

[54] Joonsoo Kim, Kyeongho Kim, and Moonsu Jang. Cyber-physical battlefield platform for large-scale cybersecurity exercises. In *2019 11th International Conference on Cyber Conflict (CyCon)*, volume 900, pages 1–19. IEEE, 2019.

[55] E.D. Knapp and J.T. Langill. *Industrial Network Security: Securing critical infrastructure networks for smart grid, SCADA, and other Industrial Control Systems*. Syngress, 2014.

[56] W. Knowles, D. Prince, D. Hutchison, J.F.P. Disso, and K. Jones. A survey of cyber security management in industrial control systems. *International journal of critical infrastructure protection*, 9:52–80, 2015.

[57] Security Lancaster. ICS Testbeds: An Online Living Resource. `http://ics-testbed.co.uk`, 2020.

[58] Security Lancaster. Research. `https://bit.ly/38Wsvb0`, 2020.

[59] Yong Li and Min Chen. Software-defined network function virtualization: A survey. *IEEE Access*, 3:2542–2553, 2015.

[60] Light, Roger. Python MQTT Implementation. `https://bit.ly/2WiF6A4`, 2019.

[61] A.P. Mathur and N.O. Tippenhauer. Swat: a water treatment testbed for research and training on ics security. In *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*, pages 31–36. IEEE, 2016.

[62] Peter Maynard, Kieran McLaughlin, and Sakir Sezer. An open framework for deploying experimental scada testbed networks. In *5th International Symposium for ICS & SCADA Cyber Security Research 2018 5*, pages 92–101, 2018.

[63] Microsoft. Remote Desktop - Allow access to your PC. `https://bit.ly/2WkWa8N`, 2018.

[64] Microsoft. Installation of OpenSSH For Windows Server 2019 and Windows 10. `https://bit.ly/38UP3bZ`, 2019.

[65] Microsoft. Basic Firewall Policy Design. `https://bit.ly/3fxFgLC`, 2020.

[66] Microsoft. Deploy Windows Server Update Services. `https://bit.ly/3fu6keP`, 2020.

[67] Microsoft. Download Debugging Tools for Windows. `https://bit.ly/38VlTcQ`, 2020.

[68] Microsoft. Install Exchange Mailbox servers using the Setup wizard. `https://bit.ly/3gYQHw9`, 2020.

[69] Microsoft. Overview of file sharing using the SMB 3 protocol in Windows Server. `https://bit.ly/391OvkL`, 2020.

[70] Jelena Mirkovic, Terry V Benzel, Ted Faber, Robert Braden, John T Wroclawski, and Stephen Schwab. The deter project: Advancing the science of cyber security experimentation and test. In *2010 IEEE International Conference on Technologies for Homeland Security (HST)*, pages 1–7. IEEE, 2010.

[71] Gijs Molenaar and Stephen Preeker. Welcome to python-snap7's documentation! `https://bit.ly/`

2WjDQNg, 2013.
[72] Nagios. What Is Nagios? https://bit.ly/2Wi3Ecm, 2020.
[73] National Cyber Security Centre. NCSC NIS Guidance. https://bit.ly/2OoX7bD, 2019.
[74] nuand. bladeRF. https://bit.ly/38Ws8gC, 2020.
[75] opengear. Serial Console Servers. https://bit.ly/3eteXVm, 2020.
[76] OpenHab. Smart Home Automation. https://bit.ly/3fEywvi, 2020.
[77] OpenSource. How to Build Custom IoT Hardware With Arduino. https://red.ht/2ZruUaB, 2018.
[78] OpenZwave. Open Source Zwave Gateway. https://bit.ly/3ft8yLb, 2020.
[79] Oroulet. Python OPC-UA Documentation. https://bit.ly/3et3FAP, 2015.
[80] Ben Paske, Benjamin Green, David Hutchison, and Daniel Prince. Design and construction of an industrial control system testbed. In *PG Net-The 15th Annual PostGraduate Symposium on the Convergence of Telecommunications, Networking and Broadcasting*, 2014.
[81] Dale Peterson. Post Game Analysis: S4x19 ICS Detection Challenge. https://bit.ly/32gwD4j, 2019.
[82] Prag, Micke. Python3 Zwave Implementation. https://bit.ly/2Onmdb5, 2020.
[83] Profibus. Profibus. https://bit.ly/3fuaovk, 2020.
[84] Profinetuniversity.com. Industrial Safety Basics-Machine Safety. https://bit.ly/2DHICh9, 2020.
[85] Profinetuniversity.com. Safety Basics – Conventional vs Combined Safety. https://bit.ly/2DHILRJ, 2020.
[86] PTC. ThingWorx Industrial IoT Solutions Platform. https://bit.ly/2DyzxXI, 2020.
[87] Python. About. https://bit.ly/308QCz5, 2020.
[88] Rhyshaden.com. Power Separation Guidelines. https://bit.ly/3h2a735, 2020.
[89] 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.
[90] Rockwell. Human Machine Interface: Visualization for Smart Manufacturing. https://bit.ly/3gYCUWi, 2020.
[91] Rockwell. Programmable Controllers. https://bit.ly/3j08fdd, 2020.
[92] Samsung. Smart Home Automation. https://bit.ly/3gRIGci, 2020.
[93] Rapid SCADA. Rapid SCADA. https://bit.ly/3fnRPcl, 2020.
[94] W. Schwab and M. Poujol. The State of Industrial Cybersecurity 2018. Technical report, 2018.
[95] SEARCCH. Sharing Expertise and Artifacts for Reuse through a Cybersecurity Community Hub. https://bit.ly/3ft9ejH, 2020.
[96] Offensive Security. About Kali Linux. https://bit.ly/392txSY, 2020.
[97] Alexandru Vlad Serbanescu, Sebastian Obermeier, and Der-Yeuan Yu. Ics threat analysis using a large-scale honeynet. In *3rd International Symposium for ICS & SCADA Cyber Security Research 2015 (ICS-CSR 2015) 3*, pages 20–30, 2015.
[98] Siemens. Siemens IoT2040 Intelligent Gateway. https://sie.ag/2Wiy3Yg, 2016.
[99] Siemens. 6AV6651-7KA02-3AA4. https://sie.ag/2WhVQaK, 2020.
[100] Siemens. Siemens: Building automation and control systems | Building technology. https://sie.ag/3h1jDUr, 2020.
[101] Siemens. SIMATIC WinCC V7. https://sie.ag/2OoZqLP, 2020.
[102] Siemens. Software in the TIA Portal . https://sie.ag/3gWTJRx, 2020.
[103] Siemens. TRIAL Download SIMATIC S7-PLCSIM Advanced V2.0 SP1. https://sie.ag/32iv1aq, 2020.
[104] Silicon Labs. Z-Wave Specification. https://bit.ly/3fs46wx, 2020.
[105] splunk. Server Monitoring Software and Tools. https://splk.it/3gWmSMB, 2020.
[106] Shelly Stazzone. Industrial Control Panel Design Guide: Schematics, Standards, Design Considerations & More. https://bit.ly/3euKDtH, 2020.
[107] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn. Guide to Industrial Control Systems (ICS) Security. Technical report, 2015.
[108] Study.com. List of Free Online Electrician Courses, Classes and Learning Materials. https://bit.ly/2Zps0TJ, 2020.
[109] Tetcos. Simulate Internet of Things Using NetSim. https://bit.ly/38TxZTQ, 2017.
[110] The Office for Nuclear Regulation. Security Assessment Principles for the Civil Nuclear Industry. Technical report, 2017.
[111] The Office for Nuclear Regulation. About ONR. https://bit.ly/2B0SWzO, 2020.
[112] TrendControls. TrendControls: HVAC Controls | Building Energy Management Systems. https://bit.ly/3j3usa8, 2020.
[113] TrendControls. TrendControls: IQ Vision Supervisor. https://bit.ly/2WijsfA, 2020.
[114] Tridium. Tridium | Harness the power of the Internet of Things with Niagara. https://bit.ly/2CBBnGJ, 2020.
[115] Uvmwareiscsibuntu. Firewall. https://bit.ly/2AWbXDm, 2020.
[116] vmware. Active Directory LDAP Server and OpenL-

DAP Server Identity Source Settings. https://bit.ly/30bbypc, 2018.

[117] vmware. Configuring iSCSI Adapters and Storage. https://bit.ly/38RZmxI, 2020.

[118] vmware. Understanding Network File System Datastores. https://bit.ly/2Wzbq23, 2020.

[119] vmware. vCenter Server. https://bit.ly/3gVjlhI, 2020.

[120] vmware. VLAN Configuration. https://bit.ly/3gYVaPC, 2020.

[121] vmware. VMware API and SDK Documentation. https://bit.ly/32kDjyj, 2020.

[122] vmware. vSphere. https://bit.ly/3gWPNAf, 2020.

[123] T.J. Williams. The purdue enterprise reference architecture. *Computers in industry*, 24(2-3):141–158, 1994.

[124] Wireshark. About Wireshark. https://bit.ly/392tF4U, 2020.

[125] J. Wu and K. Kobara. Comparison of tools and simulators for control system security studies. In *IEEE 10th International Conference on Industrial Informatics*, pages 45–50. IEEE, 2012.

[126] Yale. Smart Home Security. https://bit.ly/2Ol7BJ6, 2020.

[127] Kurt Zeilenga et al. Lightweight directory access protocol (ldap): Technical specification road map. Technical report, RFC 4510, June, 2006.