http://dx.doi.org/10.14236/ewic/ICS2018.11

An Open Framework for Deploying Experimental SCADA Testbed Networks

Peter Maynard, Kieran McLaughlin and Sakir Sezer
Centre for Secure Information Technologies
Queen's University Belfast
Queen's Road, Belfast,
BT3 9DT, UK
{*p.maynard, kieran.mclaughlin, s.sezer*} *@qub.ac.uk*

A scalable framework for automatically deploying locally (or remotely) a number of virtual machines that replicate a Supervisory Control And Data Acquisition (SCADA) network is proposed. This includes multiple virtual hosts emulating sensors and actuators, with a Human Machine Interface (HMI) controlling the hosts. The presented framework contains a collection of automation scripts which build and deploy a variable number of virtual machines, pre-configured to act as either a Remote Terminal Unit (RTU), HMI or Data Historian. The presented work includes a standards compliant implementation of IEC 60870-5-104 (IEC104) and OPC Unified Architecture (OPC-UA), with the capability to support other protocols such as Modbus-TCP (Modbus) and IEC61850. This allows researchers to build testbeds that can be configured to replicate real-world deployments of SCADA networks. The framework builds upon open source libraries and is released under the Free Software Foundation approved licence, GNU General Public License version 3.

Keywords: *Dataset, ICS, IEC 60870-5-104, OPC Unified Architecture, Open-Source, SCADA, TestBed*

1. INTRODUCTION

This paper describes a framework and set of tools to enable the creation and deployment of a virtualised SCADA testbed. It defines eight requirements for an Industrial Control System (ICS) testbed, and investigates four possible use cases for the proposed framework. The purpose and contribution of this work is to address the lack of SCADA testbeds, and datasets available to the research community. Finally, this work provides a sample dataset created using the framework, that can be used for verifying network Intrusion Detection System (IDS) and other research experiments that require an openly accessible and reproducible dataset for ICS research.

Critical systems are becoming increasingly interconnected, with new targeted threats to Operational Technology (OT) being identified regularly. It is important to understand these threats in order to develop effective countermeasures. Testbed networks are used to analyse threats to ICS/SCADA systems, and to verify the effectiveness of countermeasures, though there is a lack of open testbeds that can facilitate this. One might consider taking advantage of a live SCADA site for identifying threats and verifying countermeasures. While this might be fine for enterprise systems, it is not for ICS, due to their critical nature. Many operators and vendors are unwilling to deploy untrusted and unproven components onto their networks. Another issue with using live data is anonymising packet contents. Anonymising captured packets is not as simple as replacing IP headers. Identifiable information can be stored in many locations, for example, a telnet session or within NetBIOS packets. These fields may leak internal IP address ranges, naming standards and MAC address that can uniquely identify vendor equipment and hardware configurations, and can potentially identify an industrial site with a high degree of confidence. Existing ICS testbeds have been built to replicate real industrial sites, that focus on specific network protocols and use physical hardware to replicate a real world process. While this can provide researchers with a realistic dataset replicating real world sites, it does not address other domains and communication protocols. Typically such testbeds are not reproducible by researchers and usage of them is authorised strictly by their owners, which inhibits scientific innovation. The need for an open reproducible testbed framework becomes clear when developing novel systems for ICSs. An example is Wang et al. (2016) who developed a Hierarchical online IDS for

SCADA networks, which they verified using KDD99 and the Mississippi State University datasets Morris et al. (2011). The KDD99 dataset often used for verifying IDS, discussed in Gowadiya and Jain (2015), is nearly twenty years old, thus technology and threats have changed since the dataset was released. The Mississippi State University provides access to packet captures upon request and despite being a sophisticated, accessible dataset, it is not reproducible unless one purchases the physical equipment, and gains an understanding of the process. It is difficult to quantitatively compare between IDS approaches for ICS, due to a lack of open and diverse datasets which can be used to emulate various use cases and protocols. While not all testbeds use physical ICS hardware, it is common to find testbeds which emulate network architecture and simulate process data. Using network emulation software to emulate the network adds an additional complexity to the deployment of a testbed, as the researcher will have to understand how to configure it, and may not interact with existing physical infrastructure. It also reduces the testbed's ability to accurately represent ICS fidelity, and prevents innovation of the underlying network infrastructure. Domain fidelity is the ability of a testbed to mimic a real Industrial Control System system as close and accurately as possible, defined in Green et al. (2017).

To address this challenge, we present an open framework for deploying SCADA testbed networks intended for use by researchers. The framework has been implemented and the source code is available for public access under a Free Software Foundation approved licence, GNU General Public License v3. The framework presented in this paper provides a flexible foundation for use in many applications, e.g. Smart-Grid, Power Generation and Distribution, control of gas and oil infrastructure, etc. Existing testbeds tend to focus on emulated networks communicating via Modbus, while the framework currently supports IEC104 and OPC-UA, with scope for other protocols such as Modbus and IEC61850. It also supports interaction with both emulated and physical networks, along with simulated processes and physical hardware.

The paper is broken down into: related work discussing the existing literature of ICS testbeds; four use cases supported by the framework; followed by an overview of the framework and what it currently consists of; then a review of the published dataset and testbed deployment steps; Finally it concludes with a discussion of ongoing work.

Contributions: This paper proposes a novel open source framework, for creating, deploying and managing SCADA testbeds. It discusses the desired features of a testbed, along with four use cases of the framework. Finally, the paper presents a 150 minute packet capture produced using the framework, generated with a configuration consisting of five RTUs, a HMI and a Data Historian. The virtual nodes are controlled and monitored with IEC104 and OPC-UA.

2. RELATED WORK

A number of existing publications address the issue of testbeds for SCADA systems, either directly or indirectly. This section will discuss literature which solely proposes and discusses an ICS testbed, or created one as a by-product of their experiments.

Many testbeds attempt to strike a balance between network functionality and accurately representing a high level of ICS domain fidelity. Davis et al. (2006) used a high voltage power systems simulator called Power World Simulator[1], and combined it with the network and attack emulator RINSE, described in Liljenstam et al. (2005), to create a SCADA testbed. Davis used the field bus protocol Modbus to interface with the process simulator, then performed and analysed the effects of a network based Denial of Service (DOS) attack. This paper highlights the need for SCADA testbeds to clearly understand the effects of such attacks on critical infrastructure, and how they might affect the processes being controlled. The use of the network emulator RINSE allows for a large scale real-time human/machine-in-the-loop network simulation of attack and defence techniques, which would require extensive physical equipment and expertise if built from scratch. These types of experiments are desired for developing modern ICS testbeds, with a focus on network infrastructure. However, using network emulators lowers the level of domain fidelity that can be represented due to the limitation of the emulators, to accurately represent a real network. Another testbed was proposed by Mallouhi et al. (2011) designed for analysing the security of SCADA control systems. They again use both Power World Simulator and Modbus. They performed a DOS and Man-in-the-middle based attacks, sending illegal Modbus commands and perform TCP based SYN and ACK flooding. Mallouhi used the network simulator OPNET[2], which restricts the level of fidelity and the testbeds's ability to interact with physical devices and networks.

Chromik et al. (2017) used the Mosaik[3] smart-grid simulator to simulate the power distribution

[1]Power World Simulator - `https://www.powerworld.com/products/simulator/overview/`
[2]OPNET Network Simulator - `http://opnetprojects.com/opnet-network-simulator/`
[3]Mosaik - `https://mosaik.offis.de/`

network and the control network, running Modbus-TCP. They proposed a decentralised process-aware monitoring system using the Bro IDS. Using process simulators allows for quick deployment of multiple types of devices and supports different use cases. There is a drawback of using the Mosaik simulator when performing the security checks defined in the RTU logic, resulting in a non-negligible delay when a change is passed between mutually dependent simulators. The advantage is that it can be used alongside physical hardware. Alves et al. (2016) examines the fidelity of a virtual SCADA testbed and a physical testbed. They study the effects of an attack to both testbeds. Then compare the outcome of both virtual and physical. They simulate a gas pipeline, which involves a Programmable Logic Controller (PLC) and HMI to maintain the pressure inside the simulated pipe. Simulink/Matlab was used to simulate the process data within the virtual systems. OpenPLC was used in both testbeds, which is an open source PLC that supports Modbus and all five IEC61131-3 programming languages. It was deployed on a Raspberry PI using an extension board called UniPi. They noticed a difference in real-time behaviour between the two PLC deployments. This is due the virtual PLC running in a hypervisor and scheduling the process to wakeup at an undefined time. This increases the maximum deviation of the cycle time to 4.44ms compared to the physical system delay of 0.3ms. While it does not affect their experimental simulation it is an important metric to consider when developing a testbed.

Reaves and Morris (2012) proposed an open Python based framework for creating virtual devices and process simulators, for use in an ICS testbed. They have virtualused the logic behind the device to replicate PLC ladder logic. This allows for a more realistic operation. They also support integration with physical hardware, and have configured it to replicate an ICS laboratory configuration, described in Morris et al. (2011). Unlike other frameworks they did not use a network simulator such as OPNET or RINSE, due to the additional complexity they introduce. They support the use of Modbus-TCP within the testbed as well as virtual and physical serial communications. This framework lacks the support of other protocols such as IEC104 and OPC-UA, though the authors stated it is possible to expand to other protocols. Testbeds that use network emulators and process simulators are quick and easy to deploy and reproduce. However, they lack the ability to accurately replicate real world systems to a high level of domain fidelity. Green et al. (2017) have created a physical ICS testbed which supports a number of Siemens and CISCO devices, and is described in a second paper by the same authors, Green et al. (2017), along with ten lessons they learnt while developing and utilising a SCADA testbed. They performed two attacks on the testbed, one network based and the other host based, describing the steps an adversary would follow if it were a real system. Green highlights the domain specific knowledge required to compromise a SCADA system, and how the attacker may gain this information by interacting with the ICS testbed. This testbed has a high level of domain fidelity, but is not easily reproducible without having to purchase a number of physical devices.

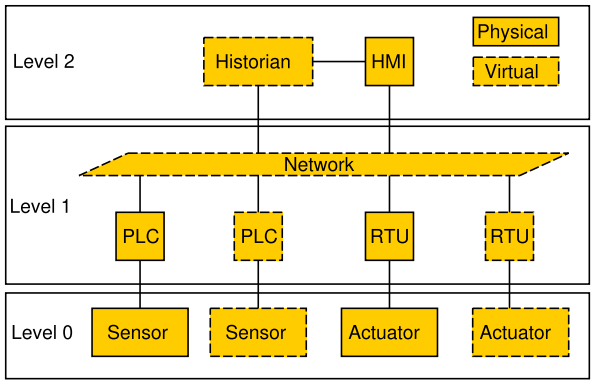

Figure 1: *A highlevel architecture diagram of a SCADA network. Showing a testbeds possible configuration, that can interact with both virtual and physical devices.*

A typical SCADA network is shown in figure 1, showing the bottom three architecture layers that can be thought of as: Layer 0 Process; Layer 1 Control; and Layer 2 Supervisory. Figure 1 highlights the possible combinations of physical and virtual devices. For example, a physical HMI may communicate with one or more physical PLCs as well as one or more virtual RTUs.

Cintuglu et al. (2017) have performed a survey of existing cyber-physical smart-grid testbeds. Based on their findings they proposed a set of taxonomies and guidelines for the development of future testbeds. Also, they identified a set of common attacks ranging from precision insider, rogue software to database attack and malformed packets. Cintuglu focused specifically on smart-grid testbeds rather than generic SCADA networks, while smart-grid systems support many of the same fieldbus protocols (Modbus, DNP3, OPC-UA, IEC16850 and C37.118) as generic SCADA, the wider literature lacks work focusing on IEC104.With a combination of virtual, physical and hybrid testbeds, only a few allowed external access for researchers. Likewise many of the testbeds were non-reproducible.

As highlighted in the introduction section there is a lack of modern ICS datasets. Rodofile et al. (2017) has defined ten requirements for attack dataset

creation specific for SCADA networks. These are listed below:

- **R1**: Able to parse SCADA protocol messages.
- **R2**: Able to replicate the SCADA protocol stack.
- **R3**: Able to sniff local SCADA network traffic.
- **R4**: Inject anomalous SCADA protocol messages into the network.
- **R5**: Modify protocol message data in real-time.
- **R6**: Provide a protocol master service for masquerading.
- **R7**: Provide a protocol slave service for masquerading.
- **R8**: Provide SCADA network discovery/reconnaissance to target SCADA applications.
- **R9**: Able to replay previous SCADA protocol messages.
- **R10**: Able to flood a SCADA service with anomalous messages.

However, the requirements proposed by Rodofile et.al mostly focused on the needs of network based attacks, that can be performed on traditional IT such as, Replay, Man-in-the-middle (MITM) and DOS. They lack the requirements that directly relate to the operation of a SCADA environment, e.g. an additional requirement might be the ability to replicate to a high degree of fidelity a real world process. The proposed framework's scope is focused on the creation of virtualised nodes and support the interaction with physical networks and devices, along with process simulators and network emulators. It addresses the ten points discussed above, and supports the operation of four use cases, discussed below.

3. USE CASE

There are a number of reasons for building a testbed. While this paper does not attempt to discuss them all, it does propose four common examples which this framework is capable of supporting: Packet Generation; Attack Simulations; Agent Benchmarking and Extending Limited Hardware.

3.1. Packet Generation

The objective of the first use case is to generate network traffic from a large number of devices, with a high level of domain fidelity, where ideally process information would be included in the traffic. As highlighted earlier, creating datasets from a live system is not always possible: it is labour intensive, one needs to locate a suitable tap point and gain approval from the plant operators. This may be restricted by the plant operation and policies, as interference with a working network may lead to unforeseen circumstances and leak identifiable information. Packet generation can be used to test proposed changes to the SCADA network before being deployed into the live system. It can also perform stress testing of devices using legitimate looking packets. Interesting research use-cases include experimentation with different networking paradigms, such as Information Centric Networking (ICN) or IPv6, that have not been applied to ICS networks.

3.2. Attack Simulations

This case considers simulation of complex attacks on SCADA systems, and aims to perform risk analysis of a replicated ICS network, without adversely affecting the live system. The testbed may be used by red teams in an attempt to compromise testbed nodes, whilst analysing the consequence and getting full packet capture analysis. If they are successful, the process can be re-performed with additional countermeasures in place, allowing testing of the new countermeasures in the context of security and how it may affect the site processes. The network captures of the read team exercise can be published as an open datasets for verifying IDS which can be reproduced and confirm the results by other researchers.

3.3. Agent Benchmarking

The objective of this use case is to support the benchmarking of agent host based systems, which are typically not performed on live systems due to vendor restrictions. Unless the agent is trusted by vendors, it is often prohibited from being deployed, with a risk of breach of contract. By using a testbed that accurately represents the real industrial site, it is possible to monitor the use of agent based software without causing disruptions. Provided the testbed is freely modifiable, functionally accurate and can integrate with physical hardware, it would be possible to perform a benchmark of the agent.

3.4. Extending Limited Hardware

The final case is to extend an existing physical testbed to include communication protocols and configurations which were not possible with the existing hardware. This could require the use of network emulators, to extend the networking equipment, and process simulators, to extend the

process control equipment. By coupling virtual and physical hardware together, it is possible to create a complex and highly realistic testbed, allowing for large deployments which combine multiple protocols and devices to be created and analysed. This use case could be used alongside the others to enhance their results.

4. THE TESTBED FRAMEWORK

This section introduces the framework and the technical features that were derived from the literature review and also integrates the ten points made by Rodofile et al. (2017) in section 2. The features are discussed below, and table 1 shows each of the state-of-the-art testbeds and compares supported features.

- *Reproducible*: Is the testbed described in enough detail which allows it to be re-deployed and used by another researcher? Or has it been built using physical equipment and can only be used in that one location?
- *Scalability*: Is the testbed able to scale? Meaning can it support a large arbitrary number of devices without having to re-design the system. This supports the large number of devices seen within a SCADA network.
- *Domain Fidelity*: What level of ICS domain fidelity can be represented using the testbed? A high level of fidelity typically means it uses physical hardware and is configured in a realistic manner. A high level of domain fidelity allows the testbed to support use cases that require an accurate representation of a SCADA network.
- *Process Simulation*: Does the testbed support the use of process simulators? This allows the end nodes, in the absence of real data, to simulate and generate process data that mimics real process data.
- *Network Emulation*: Does the testbed support the use of network emulators? Network emulation creates virtualised network infrastructure, which might not be possible to achieve using limited physical equipment. Emulated networks allow for configuration of different scenarios such as packet delay and loss, which would be harder to create on physical networks.
- *Physical Network*: Can the testbed be integrated with physical network infrastructure, or does it only support a specific interface? Physical infrastructure compared to virtualised, acts in a different manner and allows interaction with existing devices.
- *Physical Devices*: Does it support the interfacing with physical ICS equipment, which allows for a higher level of fidelity? Like physical networks it is hard to virtualise all the nuances associated with a device.
- *Multi-Protocol*: Does the testbed support more than one field bus protocol, or plans to support more in the future? Supporting a range of protocols can more accurately represent a SCADA network, as it is common for networks to consist of many devices from different vendors, which support multiple protocols.

Table 1: *A table showing a comparison of state-of-the-art testbed features against this proposed framework. High (H) Medium (M).*

	Davis et al. (2006)	Mallouhi et al. (2011)	Reaves and Morris (2012)	Alves et al. (2016)	Chromik et al. (2017)	Green et al. (2017)	**Proposed Framework**
Reproducible	✗	✗	✓	✓	✓	✗	✓
Scalable	✓	✗	✓	✓	✓	✗	✓
Domain Fidelity	M	M	H	M	M	H	H
Process Simulation	✓	✓	✓	✓	✓	✗	✓
Network Emulation	✓	✓	✗	✗	✗	✗	✓
Physical Network	✓	✗	✓	✓	✓	✓	✓
Physical Devices	✗	✗	✓	✓	✗	✓	✓
Multi Protocol	✗	✗	✗[4]	✗	✗	✓	✓

4.1. Overview of Framework

The framework is a collection of scripts which deploy and configure virtual machines (Nodes) for use in a SCADA testbed. It is not tied to a specific vendor or technology, and containerisation can be used instead of virtulisation if desired. The choice of operating system and architecture is configurable. By Default Oracle Virtual Box is used to create the testbed nodes, supporting two networking options, 1) internal virtualised network provided by virtualbox, or 2) bridging the node to the hosts network interface. Both have their advantages and can be used for different use cases, e.g. to interact with existing network infrastructure or a network

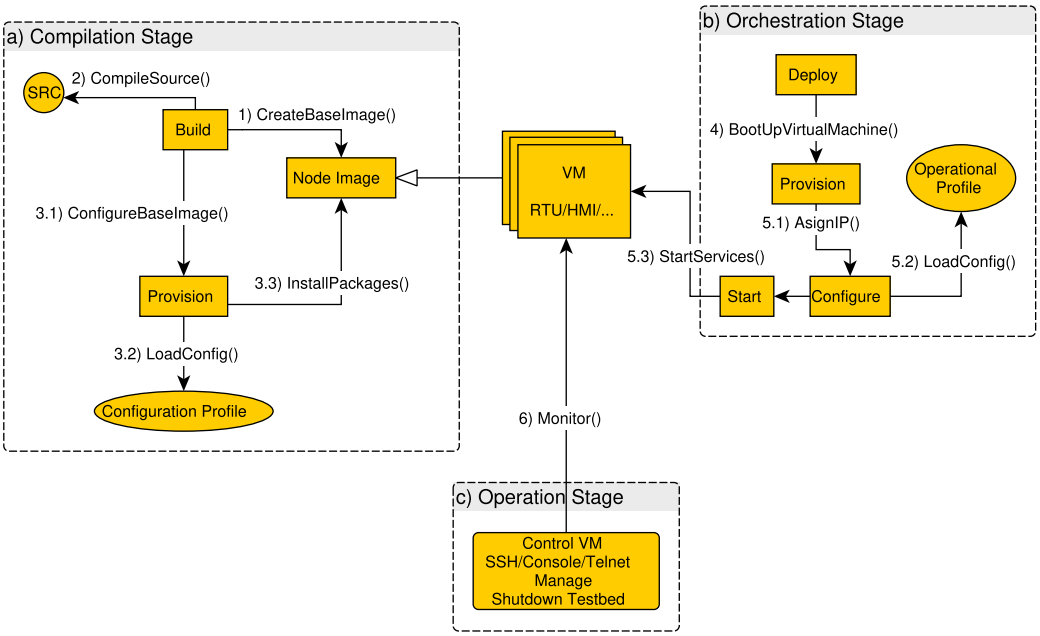

Figure 2: *Communication diagram of the three main stages of the framework.*

emulators, one would use the second option to bridge the node to the host interface. Since the testbeds can interact with physical infrastructure, it supports interoperability with industrial hardware, such as PLCs and RTUs. The presented version of the framework supports IEC104 and OPC-UA allowing native communication, with scope of adding additional protocols. None of the testbeds reviewed explicitly supports IEC104, despite the protocol's wide usage throughout Europe and Asia. This may be attributed to a lack of accessible libraries for IEC104 compared to other protocols. This framework currently uses the well known implementation of IEC104, OpenMUC[4] as it is compatible with the GPL licence, unlike other implementations such as IndigoSCADA[5], which is distributed with a non Free Software Foundation approved licence.

Two terms which will be used throughout the remainder of the paper are: Operation Profile and Configuration Profile. Essentially there is a set of scripts which configure how each node is provisioned and how they interact with each other. Operation profile defines the deployment of nodes, simulators and configuration of the network, while a configuration profile, defines how a node should be configured to represent a certain system. e.g. HMI or RTU. The purpose of having these profiles, is to encourage the community to contribute their own profiles that represents their use cases, while reducing the level of complexity required to design and deploy a testbed. Figure 2 shows a communication diagram of the three stages of the framework: a) Compilation Stage; b) Orchestration Stage; and the c) Operation Stage. The remainder of this section discuses each stage in turn.

4.1.1. Compilation Stage

The compilation stage has two goals, the first is to compile the required software, and second is to build a base image. Each testbed node is essentially a subclass of the base image, allowing for each node to have a common set of features which can be extended upon to realise their configuration profile. The compilation stage will compile all software needed during the lifetime of the testbed, and compilation is performed in a dedicated, clean and reproducible virtual environment. This is to ensure that each node is reproducible and the binary output is not adversely affected by external factors, or issues with the host machines. This stage is a pre-requisite for the other stages, which the Orchestration and Operation stages rely upon.

4.1.2. Orchestration Stage

Once the software has been compiled and there are no failed tests, it is possible to run the orchestration stage. This stage initialises and starts a defined number of nodes. Based on the operational profile, each node has their own configuration profile. It onboards the new nodes and performs network address assignment. If it is the first boot of the node, it will be provisioned with the required packages

[4]OpenMUC - `https://www.openmuc.org/openmuc/`
[5]IndigoSCADA - `http://www.enscada.com/a7khg9/IndigoSCADA.html`

based on the configuration profile. Otherwise, a previously initialised node will be used and activated.

4.1.3. Operation Stage

After the other stages have completed, the testbed will be in a full working state. This stage allows the operators to manage, control and interact with the nodes, as well as initiate a graceful shut down of the testbed. What exactly is performed at this stage depends on the individual use case. All nodes on the network are now communicating as per their respective configuration profiles.

The framework supports an expert in the ICS domain, to design and deploy a testbed with a high level of domain fidelity. To address the level of domain fidelity which can be represented by a testbed, the framework provides the flexibility to integrate process simulators and network emulators depending on their requirements, while simultaneously supporting interaction with physical infrastructure. Currently, support for process simulators and network emulators is out of the scope of this framework, however process simulators can easily be integrated into the framework, allowing each node to have its own simulator. The reason behind this is to allow for a flexible framework that can be used for a range of use cases, without having to explicitly define a network or process simulator, but also facilitating the ability to use these programs within a testbed. The framework addresses the issue of scalability in a number of ways: 1) It can be integrated into existing network infrastructure; 2) It supports deployment of nodes locally on one or more hosts, or on cloud platforms; 3) It uses virtual machines, via the Oracle VM VirtualBox[6] hypervisor; 4) It uses open source protocol libraries which are fully standards compliant.

5. EXAMPLE SETUP

The example scenario is a common metering application, that can be found in gas and water monitoring networks, in which a sensor is routinely interrogated and a value is returned. To add a bit of variety into the network traffic, a data historian is included using OPC-UA. The example testbed consists of seven virtulised nodes, created using three configuration profiles: 1) HMI; 2) Data historian; 3) RTUx5. Once all the nodes are built and brought online, the HMI initiates communication using the IEC104 protocol to each of the five RTUs at a specified polling interval and facilitates the control of the nodes. The Data Historian also initiates communication, but via OPC-UA, and stores the data in a local database. This is connected up using physical infrastructure, a CISCO switch is used to route the packets and DHCP and DNS is

[6]Oracle VM VirtualBox - `https://www.virtualbox.org/`

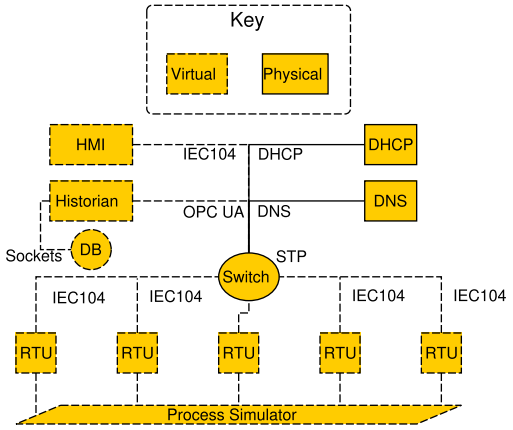

Figure 3: *Network diagram showing a sample testbed, combining physical and virtual components.*

supported by a separate physical device. Figure 3 shows the network diagram of this operation profile. The virtual nodes are deployed onto two physical machines. NIST SP 800-82r2, Stouffer et al. (2015) recommends an ICS security architecture, which includes network segmentation, boundary protection and the use of Demilitarised Zone (DMZ) for separation between enterprise network and SCADA network. Currently the framework only supports a two-zone system (no DMZ), this will be addressed in future versions to allow the use of a three-zone design, placing the RTUs in the SCADA network and the data historian inside a DMZ.

The HMI node has a Command Line Interface (CLI), which shows the status of each connected device, and can be configured manually via the CLI, or load a local configuration file, which sets the polling interval and operational commands. When the testbed is operational, packets will be routed between nodes via the switch and from the virtual machine host. This provides the operator with three points to monitor network traffic: 1) On the hypervisor machines, guaranteed to capture any packets sent to, or generated by the virtual machines; 2) On each virtual machine, this can provide a more targeted tap; 3) Span port of the network switch, which will capture all packets on the network. This has a drawback depending on the size of the network as packets may be dropped.

6. SAMPLE DATASET

The testbed used for the creation of the dataset[7], takes full advantage of the framework. It auto deploys two primary services: an HMI and Historian

[7]Access the dataset at: `https://dx.doi.org/10.6084/m9.figshare.6133457.v1`

Table 2: *A table showing the number of packets per-host broken down by type of protocol.*

Host	IP	IEC104	OPC-UA	Other	**Total**
HMI	10.50.50.150	26,158	0	17,688	43,846
Historian	10.50.50.151	0	14,695	14,927	29,622
RTU-1	10.50.50.101	3,592	2,940	5,543	12,075
RTU-2	10.50.50.102	3,665	2,941	5,876	12,482
RTU-3	10.50.50.103	3,668	2,940	5,793	12,404
RTU-4	10.50.50.104	3,690	2,940	5,771	12,404
RTU-5	10.50.50.105	3,576	930	7,933	12,442
MITM	10.50.50.99	2,390	0	3,449	5,839
SCAN	10.50.50.3	15	0	28,351	28,366

along with five RTUs. The HMI is configured every five seconds to query each of the five RTUs using IEC104. First the HMI will send an interrogation command, C_IC_NA_1 which triggers the RTU to return with a M_ME_NB_1 containing one information object containing the value of the simulated reading. This dataset only uses a basic process simulator, as it is designed to show what the framework can achieve in its current state. The data historian uses OPC-UA to query each of the RTUs every three seconds, returning the same process information as the IEC104 counterpart. The RTUs are configured to replicate the OPC profile 'Nano Embedded Device[8]' meaning it is functionally equivalent to the Core Server Facet and defines the OPC-UA TCP binary protocol as the required transport profile.

The dataset consists of 9 hosts: 1x HMI; 1x Data Historian; 5x RTU; 1x MITM Attacker; 1x Reconnaissance. The Man-in-the-middle (MITM) attack targets IEC104, which replaces the Cause of Transmission (COT) value to an invalid value. A time line of the attack is described:

- **[Host-SCAN 13:45]**: Basic network reconnaissance using a Nmap network wide scan. CMD: 'nmap -sn 10.50.50.*'
- **[Host-SCAN 13:47]**: Basic network reconnaissance looking for accessible IEC104 servers. CMD: 'nmap 10.50.50.* -p 2404'
- **[Host-SCAN 13:47]**: Full port scan of identified RTUs nodes. CMD: 'nmap 10.50.50.101 -105 -A'
- **[Host-SCAN 13:49]**: An active IEC104 scan which probes the nodes using the IEC104 protocol[9]. CMD: 'nmap -Pn -n -d –script iec-identify.nse –script-args='iec-identify.timeout=500' -p 2404 10.50.50.101-105'
- **[Host-MITM 14:19]**: Performs a MITM on RTU-1 and the HMI. CMD: 'ettercap -i enp0s8 -T -M arp -P spoof_104 /10.50.50.101/ /10.50.50.150/'

The MITM attack has been expanded from the work in Maynard et al. (2014) to support the different fields used within the IEC104 protocol. Table 2 shows a number of packets per-host, broken down by the protocol type. This dataset can be used as an example of how field bus communications look on the wire. There is a lack of both IEC104 and OPC-UA datasets, many example have a small amount of packets. Furthermore, the dataset may be used to verify the effectiveness of a network based IDS against a SCADA network. While these attacks can be detected by most traditional IT IDSs, they represent a SCADA network with a high level of domain fidelity.

6.1. Deployment Steps

This briefly describes the steps required to deploy the example testbed.

- Download the framework source code from github[10]. Detailed instructions for installing the dependencies are included in the repository.

[8]OPC Profile 'Nano Embedded Device' `https://opcfoundation-onlineapplications.org/ProfileReporting/index.htm?ModifyProfile.aspx?ProfileID=39f0d326-6c45-4f58-b834-3e22a443d8ee`

[9]`https://github.com/atimorin/PoC2013/blob/master/iec-60870-5-104/iec-60870-5-104.py`

[10]Framework Repository - `https://github.com/PMaynard/ICS-TestBed-Framework/`

- Modify the operational profile scripts to specify the number of nodes to build and deploy, along with what software will be provisioned onto the node, and network configuration: Connect to virtual network or physical. Static or Dynamic IP Address.
- Finally, execute the deployment script which will start the compilation stage followed by the orchestration stage.

7. DISCUSSION AND ONGOING WORK

Current work focuses on implementation of additional operation profiles that would automatically deploy and configure a testbed to represent a specific use cases. The more of these which are implemented, the better the framework will be able to enable researching into complex ICS attacks. Also, the development of additional configuration profiles for nodes are being undertaken to allow a broader range of ICS devices to be represented. As much of the current work focuses on generic devices such as PLCs and RTUs. The authors are planning to create operation profiles with a low barrier of entry, yet can also represent complex systems. One way to accomplish this is to use common and cheap components such as Raspberry Pi, and low end GPS modules which can be used for temporal based experiments. This framework is well positioned for experimentation with new field bus protocols and network paradigms such as Information Centric Networking[11]

An interesting future development could be the configuration of physical network switches, which may be used to automate the reconfiguration of Virtual LANs (VLANs) for a specific operational profile. This could be expanded from there to include Software Defined Networking (SDN) technology. Many testbeds do not support remote access, either due to policy or limited resources. Interconnection of testbeds is a trend which is becoming popular in other disciplines. This framework can facilitate an open decentralised platform for other researchers to connect and federate. By sharing hardware resources it would benefit the whole ICS security community.

Integrating the framework with cloud based hosting, such as Amazon Web Services (AWS) would allow for more realistic deployments as nodes can be located in remote locations. Development of an intuitive user interface to allow for the easier creation of complex testbeds could also be of benefit. e.g. Allow the creation of operation profiles using wiring and Piping and Instrumentation Diagrams (P&ID), which would be translated into orchestration stage configuration scripts.

[11] ICN `https://irtf.org/icnrg`

8. CONCLUSIONS

This paper has proposed a flexible and open framework for creating and deploying virtulised SCADA testbeds. It has defined eight features necessary in a SCADA testbed, and compared the proposed framework against existing testbeds. The proposed framework contributes to addressing the lack of modern datasets for use in verifying experimentation results, and facilitates the ability to create testbeds that: may have a high degree of fidelity; are reproducible; can scale well. A 150 minute capture is also provided, that was created using the framework for use by the wider security community. This framework provides researchers with an alternative framework that supports IEC104 and OPC-UA, compared to the more common Modbus based testbeds, such as Reaves and Morris (2012). While there is still a number of features that can be implemented into the framework, it provides a starting point for future research into ICS testbeds.

9. ACKNOWLEDGEMENTS

This work was funded by EPSRC project ADAMA, reference EP/N022866/1.

REFERENCES

Alves, T., R. Das, and T. Morris (2016). Virtualization of Industrial Control System Testbeds for Cybersecurity. In *Proceedings of the 2Nd Annual Industrial Control System Security Workshop*, ICSS '16, pp. 10–14. ACM.

Chromik, J. J., A. Remke, and B. R. Haverkort (2017). A Testbed for locally Monitoring SCADA Networks in Smart Grids. In *International Workshop Energy-Open*. IEEE.

Cintuglu, M. H., O. A. Mohammed, K. Akkaya, and A. S. Uluagac (2017). A Survey on Smart Grid Cyber-Physical System Testbeds. *19*(1), 446–464.

Davis, C. M., J. E. Tate, H. Okhravi, C. Grier, T. J. Overbye, and D. Nicol (2006). SCADA Cyber Security Testbed Development. In *2006 38th North American Power Symposium*, pp. 483–488.

Gowadiya, M. J. and A. Jain (2015). Intrusion Detection in KDD99 Dataset: A Review.

Green, B., M. Krotofil, and A. Abbasi (2017). On the Significance of Process Comprehension for Conducting Targeted ICS Attacks. In *Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy*, CPS '17, pp. 57–67. ACM.

Green, B., A. Lee, R. Antrobus, U. Roedig, D. Hutchison, and A. Rashid (2017). 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)*. USENIX Association.

Liljenstam, M., J. Liu, D. Nicol, Y. Yuan, G. Yan, and C. Grier (2005). RINSE: The real-time immersive network simulation environment for network security exercises. In *Workshop on Principles of Advanced and Distributed Simulation (PADS'05)*, pp. 119–128.

Mallouhi, M., Y. Al-Nashif, D. Cox, T. Chadaga, and S. Hariri (2011). A testbed for analyzing security of SCADA control systems (TASSCS). In *ISGT 2011*, pp. 1–7.

Maynard, P., K. McLaughlin, and B. Haberler (2014). Towards Understanding Man-In-The-Middle Attacks on IEC 60870-5-104 SCADA Networks. In *ICS-CSR*.

Morris, T., A. Srivastava, B. Reaves, W. Gao, K. Pavurapu, and R. Reddi (2011). A control system testbed to validate critical infrastructure protection concepts. *4*(2), 88–103.

Morris, T., R. Vaughn, and Y. Dandass (2011). A Testbed for SCADA Control System Cybersecurity Research and Pedagogy. In *Proceedings of the Seventh Annual Workshop on Cyber Security and Information Intelligence Research*, CSIIRW '11, pp. 27:1–27:1. ACM.

Reaves, B. and T. Morris (2012). An open virtual testbed for industrial control system security research. *11*(4), 215–229.

Rodofile, N. R., K. Radke, and E. Foo (2017). Framework for SCADA Cyber-attack Dataset Creation. In *Proceedings of the Australasian Computer Science Week Multiconference*, ACSW '17, pp. 69:1–69:10. ACM.

Stouffer, K., V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn (2015). Guide to Industrial Control Systems (ICS) Security.

Wang, H., T. Lu, X. Dong, P. Li, and M. Xie (2016). Hierarchical Online Intrusion Detection for SCADA Networks.