ASCE

High-Fidelity Cyber and Physical Simulation of Water Distribution Systems. I: Models and Data

Andrés Murillo[1]; Riccardo Taormina[2]; Nils Ole Tippenhauer[3]; Davide Salaorni[4]; Robert van Dijk[5]; Luc Jonker[6]; Simcha Vos[7]; Maarten Weyns[8]; and Stefano Galelli, M.ASCE[9]

Abstract: Numerical simulation models are a fundamental tool for planning and managing smart water networks—an evolution of water distribution systems in which physical assets are monitored and controlled by information and communication technologies. While simulation models allow us to understand the interactions between physical processes and abstract control strategies, they ignore key implementation aspects of distributed control systems, such as the required communication over digital links. As a result, the effects of anomalies and faults in the communication on the process control cannot be investigated with existing tools. In this work, we fill this gap by introducing DHALSIM (Digital HydrAuLic SIMulator), a numerical modelling platform combining EPANET-based process simulation with a network and host emulation environment, offering a high-fidelity representation of the processes occurring in the cyber domain. We illustrate DHALSIM's key functionalities by implementing it on a benchmark water distribution system, present case studies of simulated network traffic, and demonstrate how anomalies in the behavior of the communication network affect the process data received by the supervisory control and data acquisition (SCADA) server. In a companion paper, we further illustrate how DHALSIM enables research opportunities in the domain of cyber-physical security. The easily customizable and open source DHALSIM provides a "workbench" for studying smart water networks, developing digital twins, and designing a broad spectrum of engineering solutions. **DOI: 10.1061/JWRMD5.WRENG-5853.**

Author keywords: Water distribution systems; Smart urban water networks; Digital twins; EPANET; Water network tool for resilience (WNTR).

Introduction

The deployment of information and communication technologies is rapidly transforming the landscape of urban water supply, delivering more reliable and cost-effective water distribution systems—or *smart water networks*. This transition builds upon not only monitoring and control technologies, but also data analytics (Makropoulos and Savić 2019). By harnessing the information contained in large observational datasets, analytics help us solve a variety of problems, such as leakage detection, pressure management, water demand modelling, or pump scheduling (Di Nardo et al. 2021). A key role in such a transition is played by numerical simulation models, which provide a means to complement observational datasets and, most importantly, design, optimize, and test planning and management solutions before their deployment in the real world (Mala-Jetmarova et al. 2017, 2018).

With recent advances in communication technology, there is a growing chasm between the infrastructural components included in smart water networks and the numerical simulation models attempting to represent them. In particular, simulation models do not consider information and communication technologies (e.g., protocols and digital links) as well as their interactions with physical assets. This limitation is due to the original focus on physical processes and their control, motivated by isolated and centralized control approaches. With the growing integration of control into IT environments, tools such as EPANET (Rossman 2000) now need their domain, scope, and capabilities expanded to support research on smart water networks. Examples of recent steps towards a more holistic system simulation are EPANET-RTX (Hatchett et al. 2011; USEPA 2015), epanetCPA (Taormina et al. 2016, 2019), and RISKNOUGHT (Nikolopoulos et al. 2020; Nikolopoulos and Makropoulos 2022). EPANET-RTX is a real-time extension to the EPANET Hydraulic Toolkit connecting a model's controls, demands, and boundary conditions to a real-time system control and data acquisition (SCADA) historian. While EPANET-RTX allows moving data between a hydraulic solver, analytic tools, and SCADA, it does not model any specific component of the

[1]Postdoctoral Research Fellow, iTrust Centre for Research in Cyber Security, Singapore Univ. of Technology and Design, 8 Somapah Rd., Singapore 487372 (corresponding author). ORCID: https://orcid.org/0000-0001-6965-2283. Email: andres_murillo@sutd.edu.sg

[2]Assistant Professor, Faculty of Civil Engineering and Geosciences, Delft Univ. of Technology, Stevinweg 1, Delft 2628 CN, Netherlands.

[3]Professor, CISPA Helmholtz Center for Information Security, Stuhlsatzenhaus 5, Saarbrücken 66123, Germany. ORCID: https://orcid.org/0000-0001-8424-2602

[4]Ph.D. Student, Politecnico di Milano, Piazza Leonardo da Vinci, 32, Milano 20133, Italy. ORCID: https://orcid.org/0000-0001-9110-2835

[5]Undergraduate Student, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft Univ. of Technology, Van Mourik Broekmanweg 6, Delft 2628 XE, Netherlands.

[6]Undergraduate Student, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft Univ. of Technology, Van Mourik Broekmanweg 6, Delft 2628 XE, Netherlands.

[7]Undergraduate Student, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft Univ. of Technology, Van Mourik Broekmanweg 6, Delft 2628 XE, Netherlands. ORCID: https://orcid.org/0000-0003-1672-4110

[8]Undergraduate Student, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft Univ. of Technology, Van Mourik Broekmanweg 6, Delft 2628 XE, Netherlands.

[9]Professor, Pillar of Engineering Systems and Design, Singapore Univ. of Technology and Design, 8 Somapah Rd., Singapore 487372. ORCID: https://orcid.org/0000-0003-2316-3243. Email: stefano_galelli@sutd.edu.sg

Note. This manuscript was submitted on May 16, 2022; approved on December 3, 2022; published online on February 22, 2023. Discussion period open until July 22, 2023; separate discussions must be submitted for individual papers. This paper is part of the ***Journal of Water Resources Planning and Management***, © ASCE, ISSN 0733-9496.

communication network, such as the protocols used to exchange information between programable logic controller (PLCs) and SCADA. epanetCPA and RISKNOUGHT were developed to study the response of smart water networks to cyber-physical attacks. In particular, both models combine EPANET with a directed graph depicting the industrial control system (ICS), where nodes represent components (e.g., sensors, PLCs) and edges represent the elements conveying information between them (e.g., Ethernet cables). Such joint representation of the cyber and physical domains allows analysts to study how an attack originating in the cyber domain affects the behavior of the water distribution system—for instance, one could investigate the drop in water pressure caused by an attack targeting the PLC controlling a pumping station.

As mentioned above, the aforementioned models can only represent in a very simplistic way the interaction between cyber components, network, and physical system. This is because they do not fully implement the key traits of the cyber components, namely the data structures present in communication protocols (e.g., packets, messages) and the finite machine states that define the logical sequences a protocol follows. In turn, overlooking the exchange of information between the components of an ICS limits the type of data available to analysts, partially affects their realism, and hinders the depth and breadth of the analyses we are capable of (Lantz et al. 2010; Beshay et al. 2015). First, the data simulated with the aforementioned models account only for hydraulic (and water quality) processes, and do not include the information exchanged between cyber components. In contrast, being able to analyze both cyber and process data can help us understand and characterize the behavior of smart water networks, particularly when we observe anomalous behavior such as faults or even cyber attacks. Second, the simulated hydraulic processes may be unrealistically 'clean' compared to real-world SCADA data (Astarloa et al. 2020; Lu et al. 2011). That is because unexpected events within the communication network, such as communication delays between sensors and PLCs or the drop of packets, are rather common (Astarloa et al. 2020; Lu et al. 2011). In turn, 'clean' data may not be ideal when designing data-driven anomaly detection systems, such as leak detection and localization algorithms. While post processing could, to a certain degree, account for these events, post processing cannot replicate exactly the rich amount of data that is generated during such events. Thus, better simulation tools and the quality of the data they provide is critical to study the resilience of smart water networks (Marchese et al. 2020), integrate smart meter data (Shafiee et al. 2020), or develop digital twins (Alzamora et al. 2021).

To fully model the complexity of smart water networks, we must therefore see them as Industrial Control Systems, because they rely on a new cyber layer with an industrial communication network and equipment that make and implement control decisions. Since these decisions are based on the information exchanged through the network, the operating conditions of the communication network impact the behavior of the water system. Hence, deploying smart water networks requires modelling tools that are able to provide realistic communication network data. Generating such data might be difficult using simulations, because communication networks are complex and distributed systems in which network nodes feature software that queues, processes, and exchanges messages with a specific syntax. In addition, the electrical and physical conditions of the network can affect this information exchange (Chertov et al. 2006). Simulating all these interactions would be too complex and not scalable to large domains. Instead, 'emulation' techniques are generally preferred to generate realistic communication data (Lantz et al. 2010; Beshay et al. 2015). Network emulation uses virtualization to have virtual nodes running the same software that physical nodes would run. The temporal behavior of these virtual nodes is comparable with the one of physical nodes—in terms of processing, queuing, and forwarding packets (Beshay et al. 2015)—leading to realistic network data.

In this work, we address the need for cyber-physical simulation and emulation tools with DHALSIM (Digital HydrAuLic-SIMulator), our novel modeling tool for smart water networks. The main goals of DHALSIM are to: (1) provide realistic ICS traffic corresponding to high-fidelity simulation of physical processes, (2) simulate the effects of network traffic faults on distributed control and, ultimately, the physical processes, and (3) provide an end-user friendly way to generate cyber topologies complementing traditional EPANET configuration files. To reach the first two goals, DHALSIM leverages a two-way integration of EPANET and MiniCPS (Antonioli and Tippenhauer 2015), an emulation tool for ICS. As for the last goal, the version of DHALSIM presented in this work features comprehensive refactoring and major improvements that extend the functionality and usability of the prior preliminary prototype, first introduced in Murillo et al. (2020). In particular, we highlight: (1) a parser that automatically processes EPANET input files and generates an appropriate network topology, (2) the integration of epynet, a Python wrapper for EPANET that enables computationally efficient step-by-step simulations, and (3) the availability of user-friendly YAML files that allow the user to easily set up simulation experiments.

In the remainder of this manuscript, we first provide a thorough introduction to ICS and cyber data, or network packet captures, for water engineers. Considering that smart water networks are cyber-physical systems, a good understanding of ICS behavior and their bi-directional relationship with the controlled physical systems will be fundamental in the near future for water engineers. We then describe the design and implementation of DHALSIM and illustrate its key functionalities on the Anytown benchmark water distribution system, which we extend to include a cyber-physical layer with an industrial control network and equipment. Through this benchmark, we demonstrate how even small anomalies in the communication network can affect the physical processes, resulting in more noisy (and realistic) process data. Finally, we note that DHALSIM's representation of both ICS and physical processes makes it a good platform to also study the impact of cyber-physical attacks on smart water networks. Such analysis is provided in a companion paper (Murillo et al. 2022), where we illustrate how the data that DHALSIM provides may enable the development of more sophisticated intrusion detection systems that combine physical processes and network data.

Background

We begin by providing background information on ICS and the communication protocols they adopt. As we shall see, this is necessary to understand the integration of cyber and physical simulation and to explain how such integration is reflected in DHALSIM. We refer the reader to the Glossary in Appendix for additional details concerning a few technical terms used here.

Industrial Control Systems

Cyber-physical systems (CPSs) are physical and engineered systems whose operations are monitored, coordinated, controlled, and integrated by a computing and communication core (Rajkumar et al. 2010). CPSs are now present in multiple domains, such as medical, transportation, and industrial systems. ICSs are a subset of CPSs and are networked infrastructure designed to guarantee that an industrial physical process operates at all times based on a set of defined operational parameters (Humayed et al. 2017). Smart water networks are ICSs that monitor the hydraulic conditions (e.g., tank

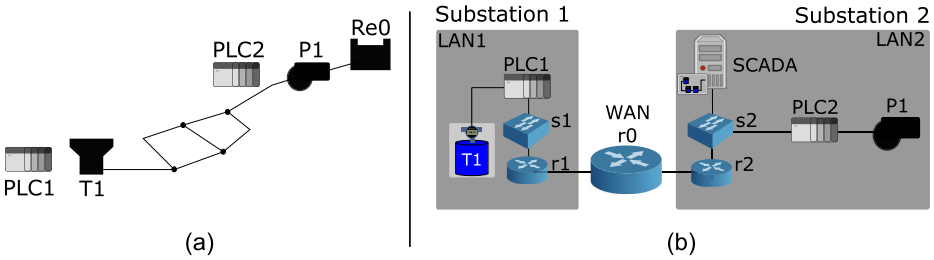

Fig. 1. Illustration of a (a) Water Distribution System; and (b) its Industrial Control System. The water level of Tank T1 is monitored by PLC1, which then relays the information to PLC2. The latter finally controls Pump P1. The water source, or reservoir, is denoted as Re0. In the cyber layer, the two PLCs are located in their own substation. This choice is explained by the spatially-distributed nature of water distribution systems. The PLCs are connected by a set of routers and switches.

water levels) and apply a predefined control logic to pumps, or other hydraulic actuators, to deliver water in sufficient quantity and quality to all customers at an appropriate pressure. Depending on the specific design and deployed technology, the ICS may also monitor water quality parameters (e.g., pH, chlorine concentration) and control additional components, such as booster pumps. The communication networks used in ICS are called industrial control networks (Galloway and Hancke 2013).

In Fig. 1, we illustrate a simple Water Distribution System (WDS) and its corresponding ICS. In this system, the level of Tank 1 (T1) is controlled by Pump 1 (P1), while two PLCs (PLC1 and PLC2), one SCADA server, and the communication network compose the ICS. PLC1 uses a sensor to measure the water level of T1; then, it sends this reading to PLC2, which uses the reading to operate P1 (e.g., to decide whether P1 needs to be turned on or off). This cycle is known as a "scan cycle" and is executed periodically in order to maintain the tank level within the desired operational parameters. In addition, the PLCs report to the SCADA server the values of different variables, such as T1 level, P1 status, P1 flow, or pressure at the junctions. Industrial control protocols (described next) are used to exchange information during these scan cycles. In addition, note that each PLC is located in a different substation. This means that each PLC is located within a Local Area Network (LAN) and both networks are connected using a Wide Area Network (WAN), represented in Fig. 1 by r0. Considering that WDSs are typically distributed across vast spatial domains, locating the PLCs in different substations is therefore a compelling network configuration. This is because due to the nature of electrical signals, PLCs need to be allocated nearby the actuators or sensors that they are controlling, hence a single substation is likely not enough for a WDS.

Industrial Control Networks

Network communications are logically divided into layers. Each layer has certain functionalities that can be offered by a specific protocol (Tanenbaum and Wetherall 2010). In this way, protocols are designed to operate at a specific layer; for example, IEEE 802.11b/g/n (commonly known as "WiFi") is a network link layer protocol. This also means that a stack of protocols must be defined to provide network applications. The name stack is used because protocols at the lower layers offer services to the upper layer and the uppermost layer is the one directly offering the final application. For instance, in the internet case of web applications, the HTTP protocol is the protocol used by browsers to offer the final content to the users. But at the same time, the HTTP protocol is supported by the combination of TCP/IP protocols. Finally, in a wireless llocal area network (WLAN), these TCP/IP protocols can be supported by IEEE 802.11. Protocols are expected to operate independently of the protocols used at the bottom or top layer. In addition, protocols are designed to satisfy specific requirements and a stack is selected also to satisfy the environment in which the stack operates.

Fig. 2 shows an example of a stack of protocols for industrial control networks. This particular stack features common industrial protocol (CIP) at the application layer to enable communications between PLCs. Using CIP, PLCs exchange messages containing the sensor readings (i.e., tank water levels) or actuator status (i.e., pump status). At the same time, CIP is supported by another protocol called "Ethernet/IP" (ENIP). Finally, CIP/ENIP are supported by TCP/IP. Notice that although the upper layer protocols are industrial protocols, the bottom protocols are the same TCP/IP used in traditional internet applications.

The stack of protocols in Fig. 2 can be used to monitor and control a system like the one shown in Fig. 1. In the scan cycle described before, PLC2 requests PLC1 the water level of T1. After receiving it, PLC2 looks up the control rules configured for P1 and then applies the appropriate decision (on whether to turn off or on the pump). Nevertheless, the process happening in the communication network domain is more complex. In this domain, the following stages have to be performed to exchange a CIP message: (1) local address resolution, (2) TCP connection establishment, (3) ENIP session registration, and (4) CIP request and response.

- The first stage ("local address resolution") does not have to be repeated at each scan cycle—it is executed only at the first cycle. At the beginning of a scan cycle, PLC2 does not have a route to send messages to PLC1, but PLC2 has configured r2 as its

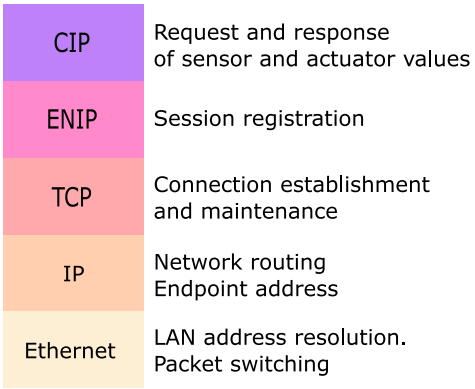

Fig. 2. CIP and ENIP protocol stack. Protocols are arranged in a stack, where lower protocols offer capabilities to upper protocols. The uppermost protocol interacts directly with the application.

default gateway. This means that PLC2 knows the IP address of r2 and uses r2 to learn the route to any other destination in the network. To access its gateway, PLC2 sends a broadcast message to s2 asking for the local media access control address (MAC) of r2. Broadcast messages are sent through all the interfaces that a switch has. This causes s2 to broadcast a request for the MAC address of r2 across LAN2. This is done using a protocol known as address resolution protocol (ARP). After the PLC2 obtains the MAC address of r2, the following stages can take place. Note that this MAC address is stored in the PLC2 cache and used for the following connections.

- The second stage ("TCP connection establishment") starts when PLC2 sends a TCP SYN (TCP-SYN) message to the IP address of PLC1. This message is sent through r2, using the MAC address resolved in the previous stage. r2 looks up in its routing table for a route to access PLC1 IP address. r2 knows that PLC1 is reachable through r1 and forwards this message to r1, through r0. Upon reception of this message, r1 forwards the message to s1. The message finally arrives at PLC1. Upon reception of the TCP-SYN message, PLC1 replies with a TCP SYN Acknowledgement (TCP-SYN-ACK) to PLC2. This message uses the same routes and path used before. Finally, PLC2 replies with an Acknowledgement (ACK) message and, when PLC1 receives it, the TCP connection is established. This mechanism is known as "three way handshake" (Tanenbaum and Wetherall 2010).
- The third stage ("ENIP session registration") starts once a TCP connection has been established. This connection is used by PLC2 and PLC1 to register an ENIP session, which is established to signal both PLCs that an exchange of CIP messages is about to take place. Then, the ENIP session is used to exchange a set of sensor readings or actuators statuses (commonly called tags) between the PLCs. To register an ENIP session, the PLCs exchange an ENIP Register Session Request and an ENIP Register Session Response message.
- Finally, with the ENIP session registered, the last stage can take place. In this stage, PLC2 sends a CIP message requesting the value of the T1 tag, and PLC1 replies to this message with a CIP response message and the T1 value. Upon reception, PLC2 closes the TCP connection. Importantly, the second to fourth stages are repeated at each scan cycle.

As shown above, there are multiple nodes, messages, and sequences followed to allow the exchange of information between PLC2 and PLC1. Multiple factors could impact these dynamics and therefore affect the information flow, ultimately impacting the hydraulic processes. We argue that accurately representing these interactions in numerical models is important in any fault resilience and cyber-security analysis, such as those required for smart water networks.

Anomalies in Industrial Control Systems

ICSs can be subject to non-malicious anomalies, such as communication disruptions or equipment malfunction. We believe that high-fidelity co-simulation environments should have the ability to represent and perform experiments that reflect such anomalies, so as to provide a more complete representation of the ICS being modeled. The need for such functionality is further stressed by the fact that being able to distinguish between malicious attacks and anomalies is still an open research challenge (Ahmed et al. 2020). Henceforth, we use the expression "events" to refer to these anomalies that are not associated to malicious activities. In this work, events are thus anomalous situations that are triggered at some point during an experiment, have a finite and known duration, and impact a network link or system in a way configured by researchers. For example, a network event could cause all packets received in a network link to be lost—a situation caused by a physical network link being cut off or by a malfunctioning network equipment.

Digital Hydraulic Simulator

We now provide details on DHALSIM's architecture as well as its design, implementation, key functionalities, and limitations. We also show how to configure simulation experiments.

Software Architecture

Fig. 3 shows the architecture of DHALSIM. The physical simulation tool is EPANET (Rossman 2000), while the network emulation tools are MiniCPS and Mininet (Antonioli and Tippenhauer 2015; Lantz et al. 2010). EPANET is used to model the physical layer of a smart water network, while Mininet and MiniCPS are used to model the cyber layer. Mininet is a platform to create virtual networks, which run inside a single host machine. With Mininet, virtual networks can be easily created to connect virtualized guests (to which we refer as Mininet nodes). A Mininet node is similar to a 'container', that is, a virtual node inside a physical machine. This virtual node has its own virtual network interfaces and can run any software installed in the host machine. MiniCPS is built on top of Mininet and provides an implementation of two popular industrial communication protocols, ENIP/CIP and Modbus. With MiniCPS, Mininet nodes can communicate between each other using ENIP/CIP or Modbus. DHALSIM uses MiniCPS and Mininet to create virtual networks. These virtual networks are composed of Mininet nodes and virtual network links. The Mininet nodes represent PLCs or SCADA and communicate using ENIP/CIP. We selected MiniCPS/Mininet to build DHALSIM, because Mininet is a widespread network experimentation tool that is easy to launch in a single machine environment, it is open source, and it is easy to extend or modify according to the development needs of DHALSIM (Lantz et al. 2010; Feamster et al. 2014; Handigol et al. 2012; Lantz and O'Connor 2015). As for the physical system, DHALSIM launches a process running an EPANET simulation. Finally, additional processes can be launched to create network events (e.g., dropping a percentage of packets at a network link during a defined number of simulation steps).

A DHALSIM experiment runs in the following way. First, the configuration files are created for the experiment—the details of which are described in the following. Second, a parser reads these configuration files, creates a Mininet network, and concurrently launches a process to run the physical simulation. The Mininet network consists of Mininet nodes, running scripts with the behavior of PLCs, or SCADA, and network links connecting these nodes. On the other hand, the physical simulation is an EPANET instance running in a step-by-step fashion. Third, during the simulation, different events or attacks can be launched. The experiment ends when the end of the physical simulation is reached and the resulting output files are stored.

Design

DHALSIM has five main components: a Parser, a Physical Simulation engine, the SQLite Database, a Network Emulation engine, and a File Generator. This subsection explains their technical details and how they interact.

Parser

The Parser is a module that reads configuration files to launch an experiment. All the configuration files use YAML (YAML Ain't

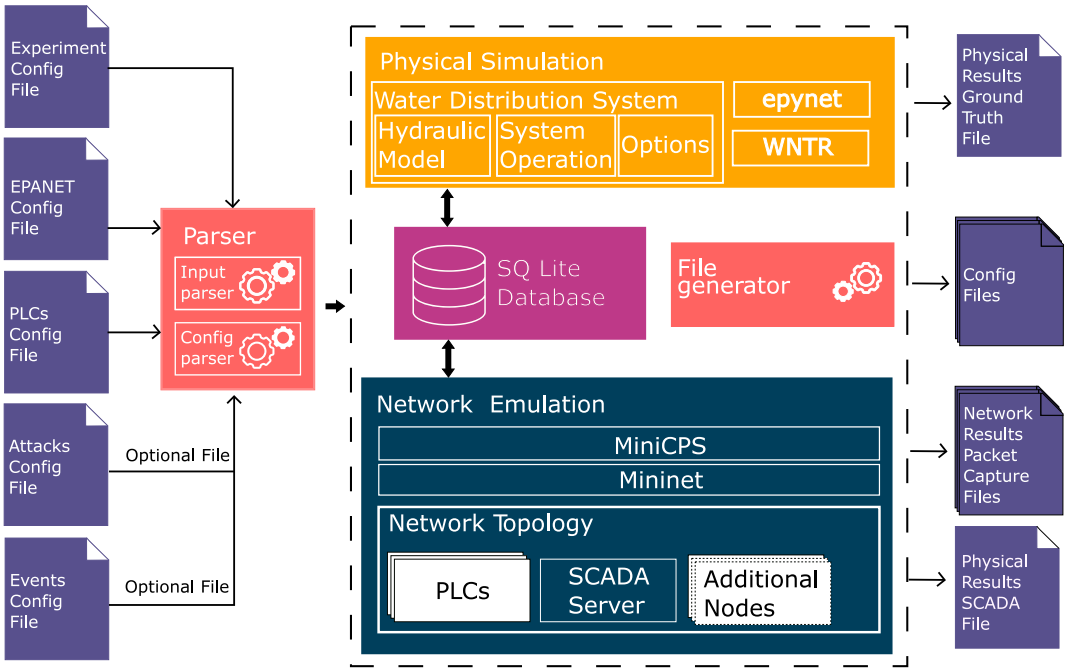

Fig. 3. DHALSIM Architecture. The architecture is composed of a parser, physical simulation, network (cyber) emulation, an SQLite Database, and a File Generator. The parser converts EPANET input files into DHALSIM topologies. The network emulation and physical simulation components run concurrently and interact using an SQLite Database. This interaction allows for synchronization of the simulation and emulation environments. The network emulation is used to represent the cyber layer of a WDS and the physical emulation runs an EPANET simulation of the WDS. The network emulation component also includes a framework to launch network attacks, described in Murillo et al. (2022). The file generator arranges all the configuration and file results in an output folder.

Markup Language). The following configuration files are processed by the parser:

- Experiment Configuration File: An experiment configuration file defines the global configuration options for a DHALSIM experiment. This file defines the EPANET input file of the experiment, the number of hydraulic time step iterations the experiment will run, the path to additional configuration files, and the type of hydraulic simulator being used. In addition, custom demand patterns and initial tank levels can be configured. To simulate less ideal conditions, Gaussian noise can be added to the sensor readings during the simulation. Finally, this file also defines the type of communication network topology used, i.e., 'simple' or 'complex'. In a simple topology, all PLCs and the SCADA server are located in the same local area network. Using the complex topology, each PLC and the SCADA server are in their own local area network. In both cases, DHALSIM automatically configures the Mininet topology or routing mechanisms to configure the network. While this is not an exhaustive list of network topologies, we note that Mininet allows for ease in prototyping network topologies, so contributors could easily develop additional templates.
- EPANET Input File: This file contains the WDS model and controls. DHALSIM automatically parses the controls to create the PLC control logic. This logic is implemented by the PLCs according to the PLCs configuration file.
- PLCs Configuration File: This is a configuration file that indicates how many PLCs the WDS has as well as the sensor or actuator handled by each PLC. The sensors in DHALSIM monitor tank levels, junction pressures, and pumps/valves flows. A sensor is a value read from a PLC and then used to apply a control rule. In addition, a sensor value can be sent to another PLC to apply a control rule. All sensor information is polled by the SCADA server. Actuators in DHALSIM are pumps and valves. The PLCs implement the control logic defined in the input file [CONTROLS] section to change the status of the actuators. The status of all actuators is also polled by the SCADA server.

These configuration files are sufficient enough to run experiments in DHALSIM that do not launch any network event. To include an event, the following optional configuration files must be provided:

- Events Configuration file: Network events are events that affect the way a network link behaves. For example, users can launch network events that cause a percentage of packets in a section of the network to be dropped, or include network delays in certain network links. Events are launched when a trigger condition is met. This condition can be based on the time since the simulation started or on the value of a given process reaching a certain value. An event has a duration that is also configured in this optional file.
- Attacks Configuration file: Attacks are configured in this optional file. Our companion paper (Murillo et al. 2022) delves into the details of attacks and their implementation.

Physical Simulation

To run the physical simulation, DHALSIM exploits two EPANET wrappers written in Python: WNTR and epynet. DHALSIM implements two workarounds to overcome current limitations of WNTR (Klise et al. 2018): (1) DHALSIM eliminates and creates new control instances at the beginning of each simulation step to enable dynamic update of the actuators settings, and (2) DHALSIM configures the WNTR simulation duration to be equal to a single hydraulic time step, and runs a number of WNTR simulations equal to the number of iterations configured in the experiment configuration File. This is done because, at the moment, WNTR does not allow

running step-by-step simulations, which are necessary for Mininet nodes to affect the system state and to enable experiments where attacks affect the physical state of the system. These workarounds impact the computational efficiency of DHALSIM experiments. Therefore, a second EPANET wrapper, epynet, has been adapted to seamlessly overcome these issues and obtain significant speedups. epynet is a Python wrapper for EPANET developed by Vitens (epynet is an object-oriented wrapper around the EPANET 2.1 community edition hydraulic network solver. https://github.com/Vitens/epynet). We modified the original package to enable step-by-step simulations and provide an interface to dynamically retrieve the properties of water network objects or change the status of the actuators.

SQLite Database

DHALSIM is a co-simulation environment in which an EPANET simulation and a MiniCPS emulation run concurrently. This means that a communication medium and a synchronization mechanism must be established between both components. MiniCPS uses an SQLite database to store the information handled by its nodes. We chose to use the same database as the communication point between the Physical Simulation and the Network Emulation. This means that the EPANET simulation reads the actuators status from the database and writes the new system state into the database. Then, the Mininet nodes read the system state from the database, run their control logic, and write the new actuator status into the database. As for the synchronization mechanism, we added special registries in the database to be used for synchronization. Such registries are a simulation master clock, used to trigger some attacks or events, and *sync* flags, used by PLCs and the physical simulation to synchronize their state and control actions. The simulation master clock is a registry that is increased every time the EPANET simulation runs a new hydraulic timestep. This clock can be used to launch attacks or events in the experiment. We chose this approach to synchronize the concurrent processes because it does not generate additional communication traffic that could introduce artifacts into simulation experiments. Additionally, the SQLite database is a stateless database, meaning that only the current values of a simulation are stored. Previous values are stored separately by the SCADA server and physical simulation, respectively. SQLite was chosen for MiniCPS as it is a relatively lightweight (easy to install and run) SQL database, and MiniCPS (and, by extension, DHALSIM) only needs basic SQL API features to be supported by the database. If a project required the distribution of a big database into a cluster, MiniCPS could be easily extended to use the same SQL queries and connect to a different database engine.

Network Emulation

The network emulation uses Mininet to launch a virtual network in which Mininet nodes use MiniCPS to run python scripts representing the PLC and SCADA behavior. The PLC behavior is composed of a *pre-loop* and *main loop*. The pre-loop is used by the PLC to initialize its variables. Additionally, the pre-loop launches a *tcpdump* capture process. TCP dump is a Linux networking tool used to capture a copy of the packets in a network interface. The case study presented in the "Results" section shows how this network information is crucial in diagnosing network or physical anomalies in cyber security experiments.

The main loop performs the following operations:

- Update local cache. Each PLC has a local cache where it stores the variables necessary for its behavior. These variables are configured in the PLC's configuration file and in the *[CONTROLS]* section of the. inp file described in the "Digital Hydraulic Simulator" section. From the perspective of a specific PLC, there are two types of variables: independent and dependent. Independent variables are "local" variables directly connected to the PLC. Dependent variables are "remote" variables, handled by a different PLC. For independent variables, the PLC performs a *get* operation that reads the values stored into the SQLite database. This reflects a PLC that is physically connected to a sensor. Dependent variables are variables not under the control of a specific PLC, but are still necessary to apply a control rule. For these variables, the PLC performs a *receive* operation. The receive operation triggers the CIP/ENIP network process to receive the tag from a different PLC, through the network, as described in the "Background" section.
- Apply a control rule. If a PLC controls an actuator (as defined in the PLCs configuration file) and that actuator has a control rule associated to it, the PLC uses the sensor values to apply the control rule. DHALSIM automatically parses the control rules defined in the *[CONTROLS]* section of an .inp file.
- Store actuator values into the SQLite database. The PLC stores the status of the actuators under its control in the SQLite database.
- Send the variables under its control to other PLCs and SCADA server. The values stored in the PLC cache are made available in the network for other PLCs or SCADA server to request. This process runs in a separate sub thread of each PLC process.

The other type of Mininet node is the SCADA server. A SCADA server is a special node in Mininet that does not implement any control logic, but periodically polls the PLCs in the network for the system state. Then, it stores those values into a data structure that is written into the *scada values* file at the end of an experiment. The inclusion of a SCADA server is useful in cyber security experiments, because some attacks could include concealing techniques to mask the impact of a cyber-physical attack (Taormina et al. 2017; Tuptuk et al. 2021). If network attacks or events are configured, additional Mininet nodes are launched. In such a case, each node runs a script for the configured network attacks or events.

File Generator

This module automatically generates documentation regarding experiments run with DHALSIM. The documentation includes all configuration files and parameters loaded into an experiment. In this way, results generated by DHALSIM can be easily replicated by other researchers.

Implementation

DHALSIM is a Python Open Source Software with MIT License and available as a Github repository at https://github.com/afmurillo/DHALSIM. DHALSIM has been developed and tested using Ubuntu 20.04 and requires Mininet, MiniCPS, WNTR, and epynet as dependencies. An automatic installation script is provided in the repository. DHALSIM is written in Python 2.7 and Python 3.8.10. The need for Python 2.7 stems from Mininet and MiniCPS, which do not yet offer full support for Python 3.8.10. The project structure is simple and has four main folders: (1) the *dhalsim* folder contains the source code of the project, (2) doc provides project documentation, (3) test provides automatic tests for the project, and (4) examples provides some smart water networks that have been developed and tested. Note that adding new topologies into DHALSIM only requires creating some simple configuration files, as explained below.

Eight topologies are already provided into the repository. These are Anytown, C-Town, KY-13, KY-14, KY-15 [all described in Jolly et al. (2014)], Minitown (Taormina et al. 2017), and WADI (Ahmed et al. 2017). Launching an experiment in DHALSIM is done through the Linux command console using the command `sudo dhalsim <experiment configuration file>`, where the text between the '< >' should be replaced with the path

and name of the experiment configuration file. The experiment runs automatically for the configured number of iterations. When the experiment finishes, an "output" folder is created with all the experiment files. The output folder has two types of files: physical results files and network files. The physical results file are .csv files containing the results of the Physical simulation (EPANET). There are two physical results files: ground truth and SCADA values. The ground truth has the real values generated by the physical simulation, whereas the SCADA values are the values that the SCADA server polled from the PLCs in the network. Notice that if an attacker launches a concealment attack like those shown in Taormina et al. (2017), the values from SCADA and ground truth may differ. The network files are the *tcpdump* capture files, which have the extension .pcap. The capture files have all the network messages seen by a Mininet node during the experiment. This provides researchers with all the network packets that a DHALSIM experiment generates during its execution. The network files and physical files have timestamps that can be used to correlate the physical and network information. In this way, researchers can see the impact that network behavior has on the physical system, and vice-versa. The output folder also has a sub folder called "configuration". This sub folder contains all the configuration files necessary to run the experiment again. These files include the experiment configuration file, additional YAML files, the .inp file used in the experiment, any additional custom demand patterns or tank levels files, and a *readme* file with additional experiment information. This configuration folder is provided to allow researchers to perform repeatable experiments and facilitate researchers to recall the conditions under which a particular result file was generated.

Key Functionalities

DHALSIM has some key functionalities that differentiate it from previous modelling tools, such as epanetCPA (Taormina et al. 2019) and RISKNOUGHT (Nikolopoulos et al. 2020):

- By integrating Mininet and MiniCPS, DHALSIM uses real networking protocols and software, while epanetCPA and RISKNOUGHT only simulate abstract network interactions between the WDS nodes. For this reason, DHALSIM can output detailed .pcap capture files with network traffic.
- Similar to epanetCPA and RISKNOUGHT, DHALSIM automatically imports any EPANET input file. A cyber-physical system for the hydraulic model can be built by creating a PLCs configuration file, as described in section "Digital Hydraulic Simulator."
- DHALSIM provides simple YAML configuration files that can be created or modified to configure and launch attacks and events in the network. This allows users who are not experts in cybersecurity or communication networks to easily use the tool for their experiments.
- DHALSIM builds on a modular architecture that provides an easy way to extend the library of available network attacks and events by simply creating new scripts and modifying some of the existing files.
- Experimental results in DHALSIM are automatically organized in an output sub folder and this sub folder includes all the configuration files used to run such experiments. This facilitates the creation of organized data sets and aids experimental reproducibility.

Limitations

The design decisions behind DHALSIM and the libraries that support it also create a few limitations:

- Mininet and MiniCPS are not yet fully supported in Python 3. This means that certain software in DHALSIM runs in Python 2.7. Overcoming this limitation would require migrating such tools into Python 3.8.10; something beyond the scope of this project. This also impedes releasing DHALSIM as a python-pip standard library. We note that the development team behind Mininet is currently working to update it to Python 3. Nevertheless the installation process of such development version still requires manual debugging and configuration. Considering that DHALSIM is a tool aimed to be used by researchers and engineers that might not have deep knowledge of Python dependencies, we decided to include the stable version of Mininet, which can be installed and tested automatically with our installation script. As for MiniCPS, it is now in the process of porting to Python 3—there is a branch that supports Python 3 and it will probably be released soon. We aim to update DHALSIM to fully support Python 3 after these releases.
- DHALSIM is a distributed system that uses different Python environments to run concurrent processes. For this reason, DHALSIM needs some synchronization mechanisms. We use an SQLite Database that MiniCPS had already deployed to synchronize the different concurrent processes. This decision was made to avoid generating additional messages that could cause artifacts in the simulations. Nevertheless, more robust synchronization mechanisms could be deployed (Eker et al. 2003; Kuhr et al. 2013). In addition, co-simulating concurrent environments is a research area on its own that goes beyond the development of DHALSIM (Gomes et al. 2018).
- DHALSIM only uses the hydraulic simulation capabilities of EPANET. In the future, it may be possible to extend such functionality so as to also simulate water quality processes.
- The current stable version of DHALSIM only supports the approach of PLCs controlling the hydraulic process. Nevertheless, we are currently developing a feature that would allow the SCADA server to centrally control the hydraulic processes.
- All network nodes in DHALSIM communicate only using the protocol stack of CIP/ENIP. This is a popular stack in industrial control protocols (Conti et al. 2021). Nevertheless, DHALSIM could be extended to support other protocol implementations. For example, MiniCPS already supports Modbus, a prominent example of field bus. Furthermore, MiniCPS can be further extended to include new protocols, thus allowing for further customization of the functionalities of DHALSIM. Similarly, a custom network topological template can be developed with the Mininet dependency by writing an additional Python script describing such topology.

Experimental Setup

Case Study

To illustrate the key functionalities of DHALSIM, we rely on the case study of Anytown, illustrated in Fig. 4. The physical layer of Anytown consists of one reservoir, two tanks (T41 and T42), and two pumps (P78 and P79) controlling the water level of the tanks. The monitoring and control process relies on three PLCs. Specifically, PLC2 and PLC3 monitor the water level of the two tanks and send this information to PLC1, which operates the pumps. Both pumps follow the same control rule and are turned on when the water level in the tanks drops below 5 m (the status of pump P78 is a function of tank T41, while the one of pump P79 depends on tank T42). The SCADA server and the three PLCs belong to separate substations (and corresponding local area networks), thereby

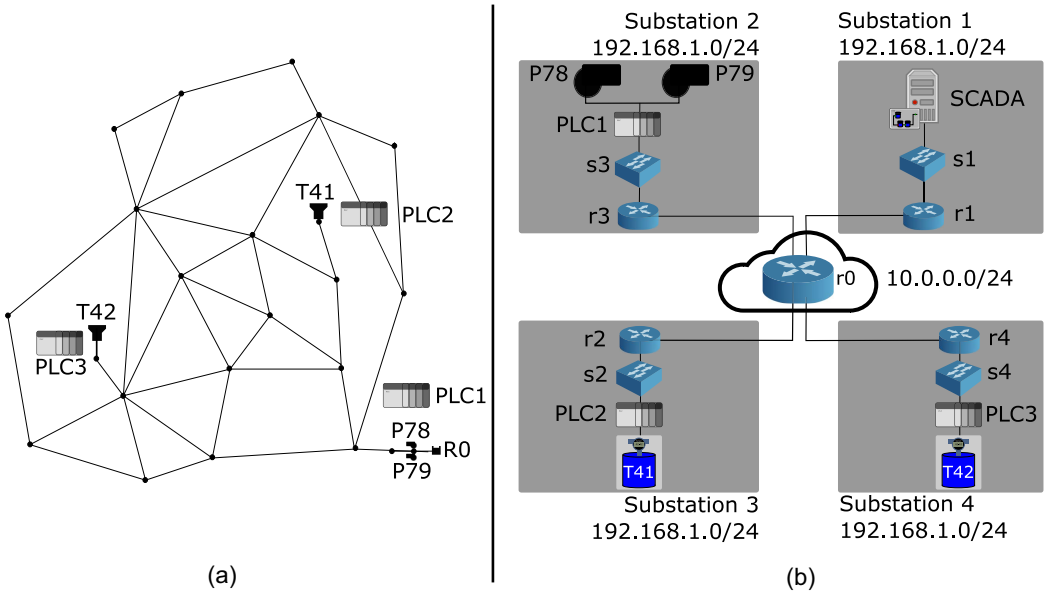

Fig. 4. (a) Physical layer; and (b) cyber-layer of the Anytown water distribution system. The physical layer consists of two tanks (T41, T42) controlled by pumps P78 and P79. The cyber layer consists of four substations connected by a router. Three PLCs control the system: PLC2 and PLC3 monitor the water level of the two tanks and send this information to PLC1, which operates the pumps. Each PLC belongs to a substation. Note the presence of an additional substation, where the SCADA server is located.

reflecting the spatially distributed nature of the water distribution system. Since each PLC belongs to a separate substation, we also use an IP assignment that is common in industrial networks. That is, all PLCs have the same IP address, but the routers use a couple of network techniques called "NAT" (Network Address Translation) and "Port Forwarding" to masquerade the IP address of their respective networks. The same IP address can be reused by both substations.

All experiments are run for one week, using an hydraulic time-step of five minutes and pressure-driven analysis. We use the same initial tank levels and demand pattern for all the experiments outlined below. Finally, the experiments are carried out on an Intel Xeon (R) 81 W-2175 CPU 2.5 GHz with 128 GB of RAM running Linux Ubuntu 20.04 (Focal Fossa). With this hardware, the run time for a single one-week simulation with DHALSIM is about 24 min.

Experiments

The goal of our experiments is twofold. First, we illustrate how the joint simulation of cyber and physical processes might yield observational data characterized by more variability than the ones observed when simulating physical processes only. This is because DHALSIM enables researchers to configure parameters like sensor noise, network delays, and packet loss that could impact the physical processes. Second, we show how typical events characterizing the operations of an ICS can lead to anomalous behaviors in the water distribution system. To reach this goal, we set up two types of experiments.

Normal Operating Conditions

In the first experiment, all components of the cyber-physical system (e.g., pumps, sensors, communication links) work in normal operating conditions. For this scenario, we run DHALSIM over an horizon of seven days, from Monday at 00:00 to Sunday at 23:59. We use this experiment to establish a baseline for the system behavior and to illustrate the interactions between the communication network and physical system.

Network Anomalies

In this experiment, we simulate the water system under varying network anomalies. In particular, we focus on the link connecting PLC1 to router r2, which is used by PLC1 to receive the values of T41 and T42. These network anomalies have a duration of 12 h (Wednesday, 06:00–18:00) and are represented by a combination of packet loss and network delay. The range of the values of packet loss is 0% to 50%, with an increment of 5%. Meanwhile, for network delay, the values vary between 0 and 4,000 ms. The increment is 250 ms for the values between 0 and 3,000 ms, followed by a single value of 4,000 ms. This leads to a total of 154 simulations. These network anomalies were implemented using a Linux network tool called *tc* (traffic control). This tool is integrated into DHALSIM.

The selected ranges were chosen to model a wide range of scenarios in which network anomalies are expected to have different impacts on the physical system response. Note that low values of packet loss and network delay are typical of a congested network or a network undergoing non-disruptive anomalies, while high values are representative of major issues in the communication network, such as the temporary unavailability of a communication link. This is because sensor readings used by the PLCs to apply control strategy and operate the water network actuators travel through the communication network; if the communication is disrupted, these messages might not arrive on time to properly control the system.

Data Analysis

Recall that DHALSIM generates both physical and network data. For the physical data, the model creates one file with ground truth values and one file with SCADA values. The former contains the values of variables handled by the EPANET simulator, namely, tank level, pressure at junctions, status and flow of valves and pumps, time-stamp, and a variable indicating the status of the water system (i.e., normal operating conditions or under attack/anomaly). The SCADA file stores the values of the variables received by the SCADA server (as specified in the PLCs configuration files). Both files have a *.csv* format. For a single one-week simulation, they

Table 1. Network features extracted from the *.pcap* files

Feature name	Feature description
Timestamp	The time at which the message was captured, stored in UNIX format
MAC source	Packet MAC address source (Layer 2)
MAC destination	Packet MAC address destination (Layer 2)
Length	Total size of the packet, measured in bytes
IP source	Packet IP address source (Layer 2)
IP destination	Packet IP address destination (Layer 2)
Protocol	Packet protocol
Command	Command that a given is performing, this is a function of the Protocol and type of packet. This is a custom that identifies special protocol messages used for specific operations, like opening or resetting connections
Command time	Time since the last command message was received, in seconds

have a size of about 1.3 MB. As for the network data, DHALSIM creates one network capture file for each PLC and SCADA server. Such file stores all network messages sent and received by that node in *.pcap* format, which allows software libraries to retrieve and process the network packets. In our case, the .pcap files processing is carried out with a software library named *scapy* (Scapy is a packet manipulation program originally designed for Python). In addition to Scapy (Kobayashi et al. 2007), we used two specific parsers for ENIP and CIP messages (Urbina et al. 2016). For a single one-week simulation, the PLC2 and PLC3 *.pcap* files have a size of 8 MB, while the *.pcap* file of PLC1 has a size of 16 MB. Scapy is a Python program and not part of Mininet. The parsers were developed by us and are part of DHALSIM.

Table 1 outlines the network features extracted from each message in the capture files. The feature "Timestamp" is stored in UNIX format, a system describing a point in time. It is the number of seconds that have elapsed since the Unix epoch, excluding leap seconds. The Unix epoch is 00:00:00 UTC on January 1, 1970. This is important because all experiments in DHALSIM have the same machine clock. This means that network and physical data can be co-analyzed, because they both share the same timestamp. Features on both MAC and IP addresses are needed to uniquely identify each node in the chosen network topology. "Length" is the total size of a packet (in bytes). The feature "Protocol" is important to produce two additional features, that is, "Command" and "Command Time". A Command is a packet exchanged under the same protocol between a pair of IP addresses. For example, every time PLC1 polls PLC2 for the sensor level of Tank T41, it first initiates a TCP connection. To initiate such connection, a TCP SYN packet must be sent. Such packets have the value "TCP SYN" in our Command network features. Finally, "Command Time" is the time in seconds since the same last Command was registered. For example, Command Time stores roughly the period (in seconds) that PLC1 polls PLC2 for the T41 level. As we shall see, analyzing these features is important to identifying and classifying certain network anomalies and cyber-security attacks. We note that there are more sophisticated approaches to process network information (Bekerman et al. 2015; Bernieri et al. 2019); however, the procedure followed here is sufficient to illustrate the main functionalities of DHALSIM.

Results

We first describe the results obtained under normal operating conditions and then illustrate how problems in a sensor and network link trickle down to the physical processes.

Normal Operating Conditions

Fig. 5 illustrates the physical and network data generated under normal operating conditions. In particular, the figure depicts the water levels of tanks T41 and T42, the number of packets corresponding to each protocol, and, finally, the number of packets corresponding to each "Command". Beginning with the tank water levels, we note a long sequence of emptying and filling cycles, driven by water demand pattern and pump operations. Such expected behavior implies that the PLCs are able to successfully monitor and control the water system by exchanging a series of messages.

What is perhaps more interesting here is the number of packets (corresponding to each protocol) exchanged during the entire simulation and during a restricted period of time (Wednesday, between 06:00 and 18:00), illustrated by the two histograms in the highlighted box. PLCs exchange sensor and actuator values using the CIP protocol, which is supported by the TCP and ENIP protocols. This means that anytime a CIP message needs to be sent, TCP and ENIP messages are exchanged to set up a CIP session. Because of this, the number of packets exchanged by TCP and ENIP protocols is higher than the number of packets exchanged by CIP and CIP CM. (CIP CM and CIP are the same type of protocol. However, the former has the *requests* for a tag value and the latter has the *response* with the tag value. For this reason, both protocols have always the same number of packets.) Another important point illustrated by the figure is that the distribution of the number of packets does not change if we focus on a relatively long period of time—one week or 12 h. This is because communication networks work with periodic patterns: the system is controlled by a series of scan cycles and each scan cycle has roughly the same communication behavior.

The last two histograms show the distribution in the number of packets grouped by the "Command" feature shown in Table 1. The commands 'S', 'SA', 'A', and 'FA' correspond the TCP messages used by TCP to establish and finish TCP connections. The number of 'A' messages is higher than the other TCP messages because each one of those messages generates an 'A' (Acknowledgement) message to acknowledge its reception—something characteristic of TCP. The 'REG' commands are ENIP messages that signal a request to create an ENIP session to exchange CIP messages. Because of this, the number of REG messages is higher than the 'number of 'DATA', 'GET T42, 'GET T41', 'GET P78', and 'GET P79' messages. 'GET T42, 'GET T41', 'GET P78', and 'GET P79' are CIP CM messages requesting the values of each one of those tags. Those messages are exchanged between PLC3-PL1, PLC2-PLC1, and PLCs-SCADA. T41 and T42 messages are exchanged between PLCs and are also sent to the SCADA server, while P78 and P79 messages are only sent to the SCADA server—and so these messages have the lowest count. Finally, 'DATA' messages are CIP with the actual value of the requested flag; their count value is therefore equal to the sum of all CIP CM messages. Importantly, the periodic behavior shown by both process and network data denotes the bi-directional relationship between physical and cyber layers: under normal operating conditions, the periodic exchange of information between sensors, PLCs, and SCADA ensures that the tanks exhibit emptying and filling cycles. As we show next, anomalies or changes in the communication network can largely impact such dynamics.

Effects of Network Anomalies

The impact of network anomalies is characterized by analyzing the changes in pressure at all junctions in Anytown. More specifically, we extract from DHALSIM the time series pertaining to normal operating conditions and each combination of packet loss and network delay (on Wednesday, between 06:00 and 18:00), calculate

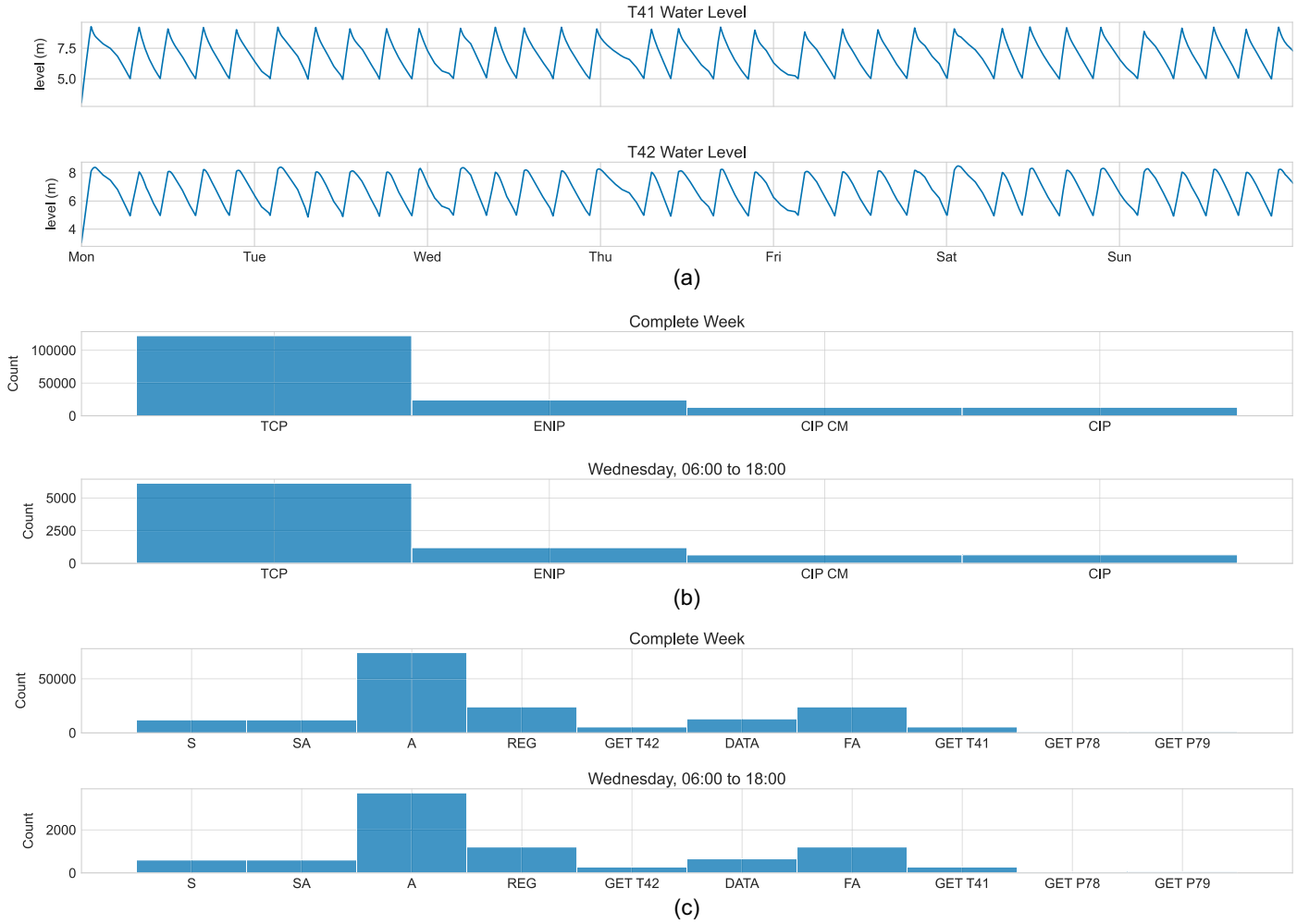

Fig. 5. Panel (a) Level of Tanks T41 and T42 under normal operating conditions. Panel (b) total number of packets grouped by protocol for the entire simulation and for Wednesday, 06:00 to 18:00. Panel (c) Command network feature for the entire simulation and for Wednesday, 06:00 to 18:00. Physical and network data generated by a one-week simulation under normal operating conditions. The first two panels, indicated by (a), illustrate the water level in tanks T41 and T42. This process is monitored and controlled by the PLCs, which exchange network messages, illustrated in the remaining panels. In particular, the two middle panels, indicated by (b), show the number of packets grouped by protocol for the entire simulation (one week) and for a shorter period (Wednesday, 06:00 to 18:00). A similar visualization is adopted for the "Command" network, indicated by (c).

the absolute value of the difference of pressure (in m) at all junctions, and then average it across space and time. This measure of disruption, illustrated in Fig. 6, highlights the presence of three main 'regions', characterized by very small (difference smaller than 1 m), slightly higher (difference between 1 and 4.00 m), and major disruptions (difference larger than 4.00 m). The heat map also shows the differential impact of network delay and packet loss. These differences are a consequence of the underlying communication network and protocols used. TCP is the transport protocol used by CIP and ENIP to transmit information about the physical system. TCP is a protocol designed to provide resilience against possible packet losses, so this is why the impact of packet losses seems to be less pronounced. For example, no value of packet loss on its own causes a complete disruption in the physical system. Meanwhile, network delays have a cutoff effect, for which values up to 2,500 ms, there is no total physical disruption, but for values of 2,500 ms and above, the physical system is disrupted—even with a packet loss of 0%. Naturally, the specific behavior illustrated in this map may change with the network topology or communication protocol, something warranting more research.

We selected three anomalies (highlighted in red in Fig. 6) to further illustrate the difference in time of the physical response of the smart water network. These anomalies are: (1) packet loss of 15% and network delay of 100 ms, (2) packet loss of 40% and network delay of 100 ms, and (3) packet loss of 25% and network delay of 2,250 ms. Fig. 7 shows the selected scenarios as well as the normal operating conditions. Note the major disruption caused by the third network anomaly, during which the behavior of the water system is totally disrupted. This disruption happens because PLC1 fails to receive updated information from readings T41 and T42, causing PLC1 to incorrectly operate pumps P78 and P79; as a consequence, tanks T41 and T42 are empty during the anomaly. A less pronounced effect is caused by anomalies 1 and 2, with less harsh anomalous network conditions. Nevertheless, the physical system deviates from its normal behavior, because the pumps are activated at some iterations later and this effect propagates even after the anomaly has ceased.

To complement the analysis of the physical data, we report an analysis of the network data in Fig. 8. The figure shows the histogram of the number of packets received by PLC1 during the selected

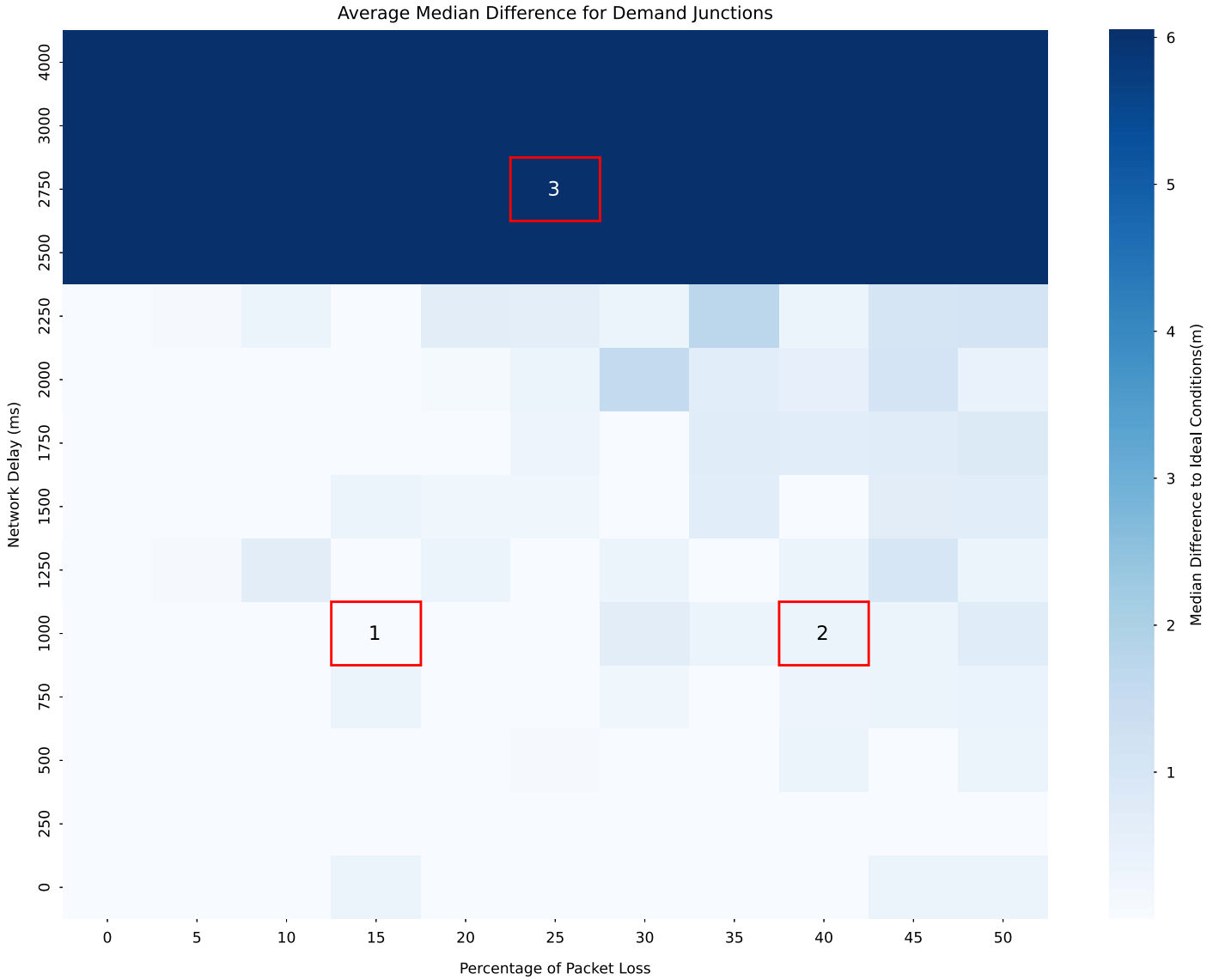

Fig. 6. Heat map of the distance in the Anytown system between normal operating conditions and network anomalies. Each point, or cell, illustrates a measure of disruption calculated by first taking the absolute value of the difference in pressure between normal operating conditions and network anomalies, and then averaging the vectors so-obtained across space (that is, across all junctions) and time. Each data point in the heat map was obtained by running a simulation of 1 week of the Anytown system, with a network anomaly on Wednesday, between 06:00 and 18:00. The network anomaly was represented by a combination of network delay in milliseconds and a percentage of packets lost. The three red boxes highlight specific anomalies illustrated in Fig. 7.

network anomalies. Anomaly 3 has the most obvious change in the number of packets of the *DATA* command. Recall that this type of packet has the tank readings for PLC1 to operate the pumps—so, without receiving these packets, PLC1 cannot properly operate the pumps. Anomalies 1 and 2 also show a drop in the number of *DATA* messages, albeit not dropping completely. Another interesting change in the number of packet messages is the behavior of the *R* command, which is sent by a network node when a connection is suddenly reset or finished. Here, we can see that during normal operating conditions there are no *R* command packets, but as the anomalies become harsher, the number of *R* packets increases.

Discussion and Conclusions

Ideally, numerical simulation models conceived for research on smart water networks should be able to represent with high fidelity not only the physical processes, but also the industrial environment that monitors and controls them. DHALSIM was conceived with such a concept in mind: its key distinguishing feature is the bi-directional coupling of EPANET with a high-fidelity network emulator (MiniCPS). Based on this feature, DHALSIM is the only model able to provide time series of both physical processes and network traffic (Table 2). The chief point to consider here is that all cyber components should not be seen as parts of a 'perfect' system that yields consistent performance over time. On the contrary, the communication process between sensors, PLCs, and SCADA is affected by many events—such as delays, noise, and packet drops—some of which can affect the physical processes and therefore the data received by SCADA, as shown in our work. By working with data generated by DHALSIM, we can thus better prepare and test the many analytics developed for smart water networks, such as those used for leakage detection (Farah and Shahrour 2017) or pump

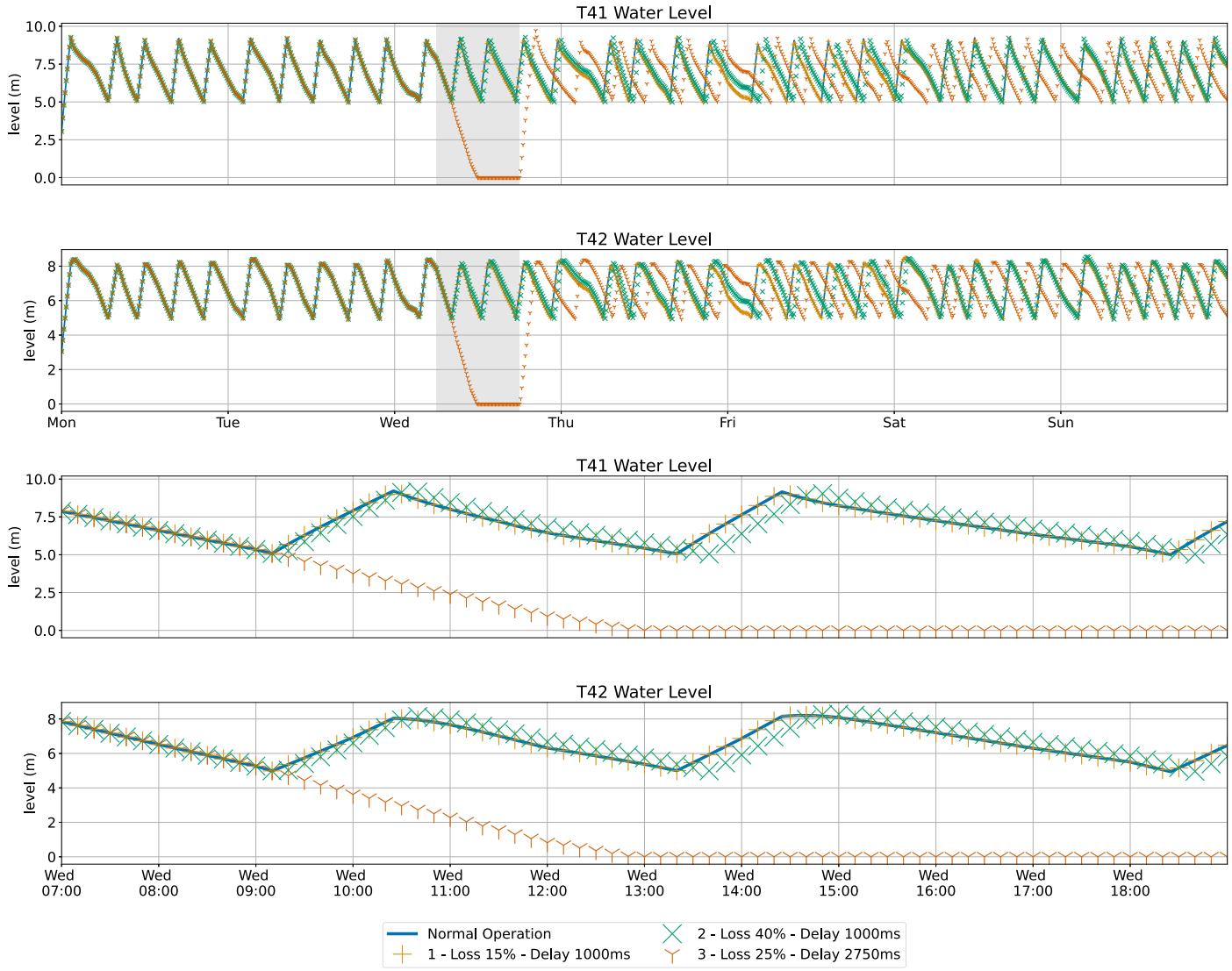

Fig. 7. Physical results of simulations with sensor noise and network anomalies. Water level in Tank T41 and T42 obtained during normal operating conditions (solid line) and when introducing packet loss and network delay. The first two panels report the results for the entire simulation period of one week, while the two bottom panels focus on Wednesday, 06:00 to 18:00, when network anomalies are simulated. These anomalies cause the pumps to activate at slightly different moments, generating a shift in the behavior of the water tank levels.

scheduling (Di Nardo et al. 2021). As we shall see in the companion paper (Murillo et al. 2022), DAHLSIM simulation of the full protocol stack also broadens the opportunities available for research in cyber-security (Tables 3 and 4). Although validating network traffic was out of the scope of this paper, we believe that further research in this field could help to develop better and more accurate experimentation platforms for CPS.

Naturally, the usability of DHALSIM would benefit from the availability of a well-calibrated EPANET model. While model calibration is indeed a difficult task, in our experience several water utilities (even for large cities) already employ sufficiently accurate EPANET models (Conejos Fuertes et al. 2020). Such utilities could already start testing DHALSIM. Even with sub-optimal EPANET models, the analysis of network traffic for a reasonable abstraction of the cyber-network would already help understand which anomalies could be expected in the cyber-layer due to cyber attacks or other disruptive events. Similarly, one could generate a vast amount of diverse data (e.g., different cyber attacks) with sub-optimal EPANET models to pre-train intrusion detection models based on Deep Learning (Tsiami and Makropoulos 2021); these models could be fine-tuned with (limited) real packet captures and SCADA historian data before deployment. As for the validation of a specific instance of DHALSIM, we believe that a reasonable approach should build on a quantitative comparison of emulated network traffic and hydraulic processes against field data. A comparison at the network level would have to check for statistical similarities between the emulated network traffic and the real traffic. We expect real traffic to be "faster" than emulated traffic (because emulated traffic speed is limited by computational resources capabilities). Nevertheless, the composition of traffic in relative terms (percentage of packets being TCP, distribution of types of packets) should be similar. A similar comparison should be repeated for the hydraulic model (Ostfeld et al. 2012). Ideally, these validation exercises should be first carried out on a physical tested (Ahmed et al. 2017) and then scaled-up to a real WDS. This is something we may focus on in the coming years.

The presence of a synchronization mechanism is an important point related to the development of DHALSIM—as well as other numerical models hinged on the bi-directional coupling of multiple modelling components. Without such a synchronization mechanism,

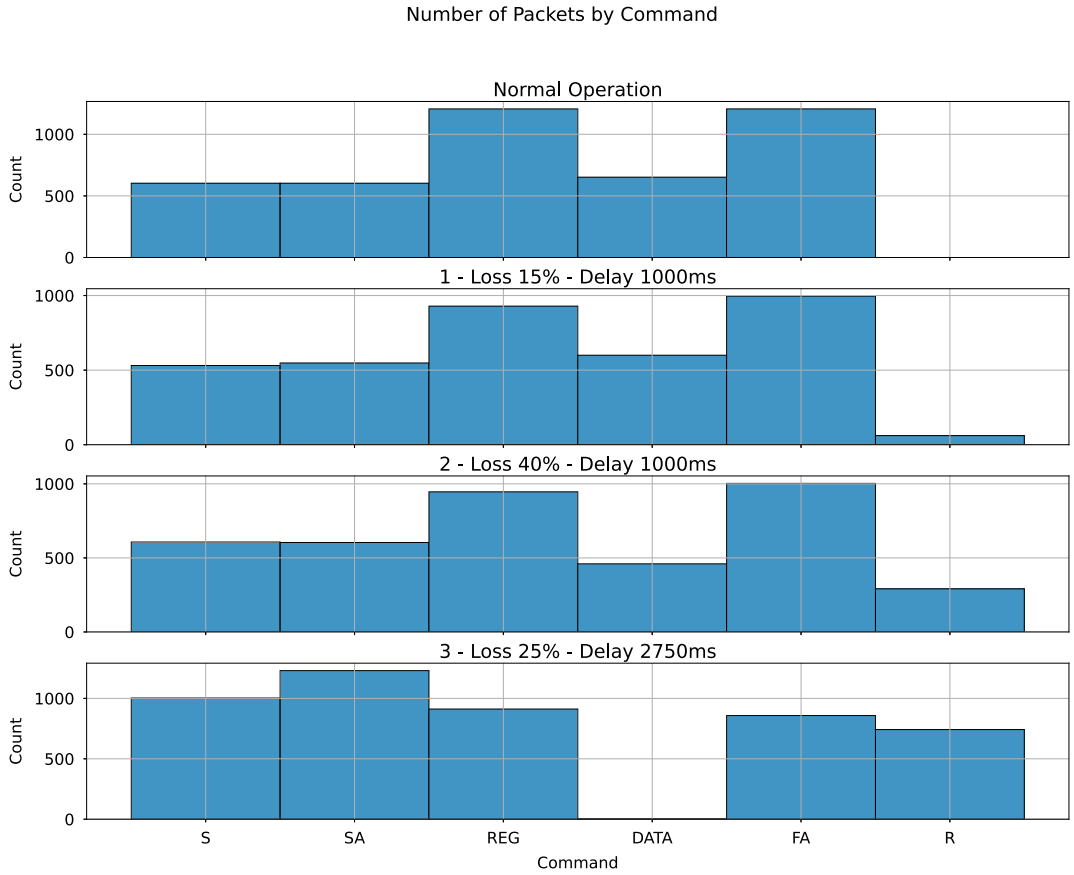

Fig. 8. Histogram of number of packets between normal operating conditions and network anomalies. The histogram shows the number of packets received by PLC1 with normal operating conditions and three network anomalies. As the network anomaly becomes more pronounced, the number of *R* (reset connection) command packets increases. This increase in the number of connections resets explains the deviations in the physical response of the system, because PLC1 does not receive updated readings of T41 and T42.

Table 2. Comparison of basic modeling features for three simulation models

Model feature	epanetCPA	Risknought	DHALSIM
Hydraulic engine	EPANET	EPANET	EPANET
Engine wrapper	MATLAB routines	WNTR (Python)	Epynet/WNTR (Python)
Process data	Yes	Yes	Yes
SCADA data	Yes	Yes	Yes
Control logic manipulation	Yes	Yes	Yes
Network emulation	No	No	Yes
Network data	No	No	Yes
Network performance manipulation	No	No	Yes

Note: Note that all models use EPANET as hydraulic engine, although the use of wrappers is very common. All models offer a representation of SCADA and control logic manipulation. However, DHALSIM is the only model that represents the communication processes happening within an industrial control system. Because of this feature, DHALSIM can produce network data (.pcap files) and offers functionalities for manipulating the performance of a communication network.

Table 3. Comparison of cyber-security modeling features for three simulation models

Cyber security feature	epanetCPA	Risknought	DHALSIM
Payload corruption	Yes	Yes	Yes
Full protocol stack corruption	No	No	Yes

Note: epanetCPA, Risknought, and DHALSIM offer payload corruption capabilities that allow to simulate attacks on sensors or hydraulic actuators. Note that DHALSIM is the only model able to corrupt the full network protocol stack. As shown in Murillo et al. (2022), this feature broadens the breadth of cyber-security analyses. For example, a user can analyse a full protocol stack to determine whether an anomaly is due to a cyber attack or a malfunctioning actuator.

Table 4. Comparison of basic extendability features for three simulation models

Extendability feature	epanetCPA	Risknought	DHALSIM
Third party tools integration	No	No	Yes
Programming language	MATLAB	Python	Python
Code available	Yes	No	Yes
Software license	MIT	NA	MIT

Note: Note that DHALSIM is able to directly integrate third party tools. This is made possible by the use of MiniCPS. Finally, note that all tools, with the exception of Risknought, are released under an open source license.

the PLCs may apply their control logic with outdated data, causing unexpected shifts in the control of physical actuators. In turn, this could lead to an inconsistent simulation of the physical processes, even when network events or cyber attacks are not present. Unlike the first version of DHALSIM introduced by Murillo et al. (2020),

the version presented here includes a simple, yet functional, synchronization mechanism that allows DHALSIM to generate the same physical results across multiple simulations. Yet, we believe this is a point warranting more research and development. Ensuring the scalability of such mechanisms across large domains or multiple virtual machines is, for example, an open challenge. This challenge becomes even greater if co-simulations environments are designed to interact in real-time with an actual physical system.

A final point worth discussing here is the relationship between DHALSIM and the growing field of digital twins for water distribution systems, broadly defined as virtual copies—or digital models—of a real system continuously fed with data to mimic the system's past, present, and future behavior (Alzamora et al. 2021). While the actual implementation of a digital twin may build on different tools (Valverde-Pérez et al. 2021), it appears that all digital twins currently available share some key components, such as a detailed hydraulic model and advanced analytics that keep the digital twin "anchored" to the state of the modelled system (USEPA 2015; Conejos Fuertes et al. 2020; Alzamora et al. 2021). Importantly, this view shows a lack of emphasis on the representation of other elements, namely the ICS equipment, which is actually seen as critical elements of a digital twin by other communities (Tao et al. 2018; Barricelli et al. 2019; Josifovska et al. 2019; Dietz and Pernul 2020). Yet, the results and arguments outlined above and the growing exposure of water utilities to cyber-physical attacks (Rasekh et al. 2016; Hassanzadeh et al. 2020) suggest that there is a need for digital twins that adequately represent all elements within a water distribution system—i.e., physical processes, industrial communication network, and ICS equipment. In this regard, DHALSIM offers a modelling platform on which digital twins of both physical and cyber layers could be developed.

Appendix. Glossary

These companion papers rely heavily on domain knowledge that might be unfamiliar to some of the readers. For this reason, a glossary has been included in this appendix to help them to understand some common terms in computer science.

Glossary

Cache: Memory component that stores data so that in the future can be retrieved faster.

Concurrent Processes: Multiple processes executing instructions simultaneously for better performance.

Container: Unit of software that runs isolated from other units, using light virtualization.

Ethernet: Set of wired link networking protocols fairly common in local area networks.

Gateway: Network node that allows data to flow from one network to another.

Guest Machine: A virtual machine running in a host machine.

Host Machine: A machine offering a virtualization platform to host virtual machines.

Instance: Example of data structure in which its attributes have determined values.

Network Interface: Interface used by a virtual or physical node to connect to a network.

Network Link: Physical or virtual link between two nodes.

Poll: A process in which one node continuously asks other nodes for updates regarding a set of variables.

Protocol: Set of rules, conventions, data structures, and algorithms that dictate how information is exchanged between two or more nodes.

Router: Networking device that forwards data between computer networks.

Session: Temporary information exchange between two or more network devices.

Switch: Network device that connects devices within a network.

Tag: Keyword assigned to a piece of information.

TCP connection: Session established with the TCP protocol to exchange information controlling possible packet loss and packet arrival order.

TCP Protocol: Communication protocol that controls the transmission of packets in a network. Offers mechanisms to ensure packets arrive to the destination.

Virtualization: Technology that enables multiple computing nodes to share physical resources, granting a degree of isolation between them.

Data Availability Statement

DHALSIM is available at https://github.com/afmurillo/DHALSIM. Some or all data, models, or code generated or used during the study are available in a repository or online in accordance with founder data retention policies. The dataset is available at https://zenodo.org/record/6528732.

Acknowledgments

This research is supported by Singapore's National Satellite Of Excellence, Design Science and Technology for Secure Critical Infrastructure (NSoE DeST-SCI) through the project "LEarning from Network and Process data to secure Water Distribution Systems (LENP-WDS)" (Award No. NSoE_DeST-SCI2019-0003) and by the Faculty of Civil Engineering and Geosciences of Delft University of Technology.

References

Ahmed, C. M., M. R. Gauthama Raman, and A. P. Mathur. 2020. "Challenges in machine learning based approaches for real-time anomaly detection in industrial control systems." In *Proc., 6th ACM on Cyber-Physical System Security Workshop, CPSS '20*, 23–29. New York: Association for Computing Machinery.

Ahmed, C. M., V. R. Palleti, and A. P. Mathur. 2017. "WADI: A water distribution testbed for research in the design of secure cyber physical systems." In *Proc., 3rd Int. Workshop on Cyber-Physical Systems for Smart Water Networks, CySWATER '17*, 25–28. New York: Association for Computing Machinery.

Alzamora, F., P. Conejos, M. Castro-Gama, and I. Vertommen. 2021. "Digital twins: A new paradigm for water supply and distribution networks." In *Proc., Hydrolink 2021-2 Artificial Intelligence*. Beijing: IAHR Secretariat.

Antonioli, D., and N. O. Tippenhauer. 2015. "MiniCPS: A toolkit for security research on CPS networks." In *Proc., 1st ACM Workshop on Cyber-Physical Systems-Security and/or PrivaCy, CPS-SPC '15*, 91–100. New York: Association for Computing Machinery.

Astarloa, A., M. Rodríguez, F. Duran, J. Jiménez, and J. Lázaro. 2020. "Synchronizing NTP referenced SCADA systems interconnected by high-availability networks." In *Proc., 2020 XXXV Conf. on Design of Circuits and Integrated Systems (DCIS)*, 1–6. New York: IEEE.

Barricelli, B. R., E. Casiraghi, and D. Fogli. 2019. "A survey on digital twin: Definitions, characteristics, applications, and design implications." *IEEE Access* 7 (Nov): 167653–167671. https://doi.org/10.1109/ACCESS.2019.2953499.

Bekerman, D., B. Shapira, L. Rokach, and A. Bar. 2015. "Unknown malware detection using network traffic classification." In *Proc., 2015 IEEE*

Conf. on Communications and Network Security (CNS), 134–142. New York: IEEE.

Bernieri, G., M. Conti, and F. Turrin. 2019. "KingFisher: An industrial security framework based on variational autoencoders." In *Proc., 1st Workshop on Machine Learning on Edge in Sensor Systems, SenSys-ML 2019*, 7–12. New York: Association for Computing Machinery. https://doi.org/10.1145/3362743.3362961.

Beshay, J. D., A. Francini, and R. Prakash. 2015. "On the fidelity of single-machine network emulation in Linux." In *Proc., 2015 IEEE 23rd Int. Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems*, 19–22. New York: IEEE.

Chertov, R., S. Fahmy, and N. Shroff. 2006. "Emulation versus simulation: A case study of TCP-targeted denial of service attacks." In *Proc., 2nd Int. Conf. on Testbeds and Research Infrastructures for the Development of Networks and Communities, 2006. TRIDENTCOM 2006*, 10. New York: IEEE.

Conejos Fuertes, P., F. Martinez Alzamora, M. Hervás Carot, and J. Alonso Campos. 2020. "Building and exploiting a digital twin for the management of drinking water distribution networks." *Urban Water J.* 17 (8): 704–713. https://doi.org/10.1080/1573062X.2020.1771382.

Conti, M., D. Donadel, and F. Turrin. 2021. "A survey on industrial control system testbeds and datasets for security research." *IEEE Commun. Surv. Tutorials* 23 (4): 2248–2294. https://doi.org/10.1109/COMST.2021.3094360.

Dietz, M., and G. Pernul. 2020. "Unleashing the digital twin's potential for ICS security." *IEEE Secur. Privacy* 18 (4): 20–27. https://doi.org/10.1109/MSEC.2019.2961650.

Di Nardo, A., D. L. Boccelli, M. Herrera, E. Creaco, A. Cominola, R. Sitzenfrei, and R. Taormina. 2021. "Smart urban water networks: Solutions, trends and challenges." *Water* 13 (4): 501. https://doi.org/10.3390/w13040501.

Eker, J., J. Janneck, E. Lee, J. Liu, X. Liu, J. Ludvig, S. Neuendorffer, S. Sachs, and Y. Xiong. 2003. "Taming heterogeneity: The Ptolemy approach." *Proc. IEEE* 91 (1): 127–144. https://doi.org/10.1109/JPROC.2002.805829.

Farah, E., and I. Shahrour. 2017. "Leakage detection using smart water system: Combination of water balance and automated minimum night flow." *Water Resour. Manage.* 31 (15): 4821–4833. https://doi.org/10.1007/s11269-017-1780-9.

Feamster, N., J. Rexford, and E. Zegura. 2014. "The road to SDN: An intellectual history of programmable networks." *ACM SIGCOMM Comput. Commun. Rev.* 44 (2): 87–98. https://doi.org/10.1145/2602204.2602219.

Galloway, B., and G. P. Hancke. 2013. "Introduction to industrial control networks." *IEEE Commun. Surv. Tutorials* 15 (2): 860–880. https://doi.org/10.1109/SURV.2012.071812.00124.

Gomes, C., C. Thule, D. Broman, P. G. Larsen, and H. Vangheluwe. 2018. "Co-simulation: A survey." *ACM Comput. Surv.* 51 (3): 1–33. https://doi.org/10.1145/3179993.

Handigol, N., B. Heller, V. Jeyakumar, B. Lantz, and N. McKeown. 2012. "Reproducible network experiments using container-based emulation." In *Proc., 8th Int. Conf. on Emerging Networking Experiments and Technologies, CoNEXT '12*, 253–264. New York: Association for Computing Machinery. https://doi.org/10.1145/2413176.2413206.

Hassanzadeh, A., A. Rasekh, S. Galelli, M. Aghashahi, R. Taormina, A. Ostfeld, and M. K. Banks. 2020. "A review of cybersecurity incidents in the water sector." *J. Environ. Eng.* 146 (5): 03120003. https://doi.org/10.1061/(ASCE)EE.1943-7870.0001686.

Hatchett, S., J. Uber, D. Boccelli, T. Haxton, A. Janke, A. Kramer, A. Matracia, and S. Panguluri. 2011. "Real-time distribution system modeling: Development, application, and insights." In *Proc., 11th Int. Conf. on Computing and Control for the Water Industry*. Exeter, UK: Centre for Water Systems, Univ. of Exeter.

Humayed, A., J. Lin, F. Li, and B. Luo. 2017. "Cyber-physical systems security: A survey." *IEEE Internet Things J.* 4 (6): 1802–1831. https://doi.org/10.1109/JIOT.2017.2703172.

Jolly, M. D., A. D. Lothes, L. S. Bryson, and L. Ormsbee. 2014. "Research database of water distribution system models." *J. Water Resour. Plann. Manage.* 140 (4): 410–416. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000352.

Josifovska, K., E. Yigitbas, and G. Engels. 2019. "Reference framework for digital twins within cyber-physical systems." In *Proc., SEsCPS '19*, 25–31. New York: IEEE.

Klise, K., R. Murray, and T. Haxton. 2018. "An overview of the water network tool for resilience (WNTR)." In *Proc., 1st Int. WDSA/CCWI Joint Conf.* Washington, DC: DOE.

Kobayashi, T. H., A. B. Batista, P. S. Brito, and P. S. Motta Pires. 2007. "Using a packet manipulation tool for security analysis of industrial network protocols." In *Proc., 2007 IEEE Conf. on Emerging Technologies and Factory Automation (EFTA 2007)*, 744–747. New York: IEEE.

Kuhr, T., T. Forster, T. Braun, and R. Gotzhein. 2013. "FERAL: Framework for simulator coupling on requirements and architecture level." In *Proc., 2013 11th ACM/IEEE Int. Conf. on Formal Methods and Models for Codesign (MEMOCODE 2013)*, 11–22. New York: IEEE.

Lantz, B., B. Heller, and N. McKeown. 2010. "A network in a laptop: Rapid prototyping for software-defined networks." In *Proc., 9th ACM SIGCOMM Workshop on Hot Topics in Networks, Hotnets-IX*. New York: Association for Computing Machinery.

Lantz, B., and B. O'Connor. 2015. "A mininet-based virtual testbed for distributed SDN development." *ACM SIGCOMM Comput. Commun. Rev.* 45 (4): 365–366. https://doi.org/10.1145/2829988.2790030.

Lu, X., Z. Lu, W. Wang, and J. Ma. 2011. "On network performance evaluation toward the smart grid: A case study of DNP3 over TCP/IP." In *Proc., 2011 IEEE Global Telecommunications Conf.—GLOBECOM 2011*, 1–6. New York: IEEE.

Makropoulos, C., and D. Savić. 2019. "Urban hydroinformatics: Past, present and future." *Water* 11 (10): 1959. https://doi.org/10.3390/w11101959.

Mala-Jetmarova, H., N. Sultanova, and D. Savic. 2017. "Lost in optimisation of water distribution systems? A literature review of system operation." *Environ. Modell. Software* 93 (Jul): 209–254. https://doi.org/10.1016/j.envsoft.2017.02.009.

Mala-Jetmarova, H., N. Sultanova, and D. Savic. 2018. "Lost in optimisation of water distribution systems? A literature review of system design." *Water* 10 (3): 307. https://doi.org/10.3390/w10030307.

Marchese, D., A. Jin, C. Fox-Lent, and I. Linkov. 2020. "Resilience for smart water systems." *J. Water Resour. Plann. Manage.* 146 (1): 02519002. https://doi.org/10.1061/(ASCE)WR.1943-5452.0001130.

Murillo, A., R. Taormina, N. Tippenhauer, and S. Galelli. 2020. "Co-simulating physical processes and network data for high-fidelity cybersecurity experiments." In *Proc., 6th Annual Industrial Control System Security (ICSS) Workshop*, 13–20. Silver Spring, MD: Applied Computer Security Associates.

Murillo, A., R. Taormina, N. O. Tippenhauer, and S. Galelli. 2022. "High-fidelity cyber and physical simulation of water distribution systems. II: Enabling cyber-physical attack localization." *J. Water Resour. Plann. Manage.* 149 (5): 04023010. https://doi.org/10.1061/JWRMD5.WRENG-5854.

Nikolopoulos, D., and C. Makropoulos. 2022. "Stikress-testing water distribution networks for cyber-physical attacks on water quality." *Urban Water J.* 19 (3): 256–270. https://doi.org/10.1080/1573062X.2021.1995446.

Nikolopoulos, D., G. Moraitis, D. Bouziotas, A. Lykou, G. Karavokiros, and C. Makropoulos. 2020. "Cyber-physical stress-testing platform for water distribution networks." *J. Environ. Eng.* 146 (7): 04020061. https://doi.org/10.1061/(ASCE)EE.1943-7870.0001722.

Ostfeld, A., et al. 2012. "Battle of the water calibration networks." *J. Water Resour. Plann. Manage.* 138 (5): 523–532. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000191.

Rajkumar, R. R., I. Lee, L. Sha, and J. Stankovic. 2010. "Cyber-physical systems: The next computing revolution." In *Proc., 47th Design Automation Conf., DAC '10*, 731–736. New York: Association for Computing Machinery. https://doi.org/10.1145/1837274.1837461.

Rasekh, A., A. Hassanzadeh, S. Mulchandani, S. Modi, and M. K. Banks. 2016. "Smart water networks and cyber security." *J. Water Resour. Plann. Manage.* 142 (7): 01816004. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000646.

Rossman, L. A. 2000. *EPANET 2: Users manual*. Cincinnati: Water Supply and Water Resources Division, National Risk Management Research Laboratory.

Shafiee, M. E., A. Rasekh, L. Sela, and A. Preis. 2020. "Streaming smart meter data integration to enable dynamic demand assignment for real-time hydraulic simulation." *J. Water Resour. Plann. Manage.* 146 (6): 06020008. https://doi.org/10.1061/(ASCE)WR.1943-5452.0001221.

Tanenbaum, A. S., and D. J. Wetherall. 2010. *Computer networks*. 5th ed. Hoboken, NJ: Prentice Hall.

Tao, F., H. Zhang, A. Liu, and A. Y. Nee. 2018. "Digital twin in industry: State-of-the-art." *IEEE Trans. Ind. Inf.* 15 (4): 2405–2415. https://doi.org/10.1109/TII.2018.2873186.

Taormina, R., S. Galelli, H. Douglas, N. O. Tippenhauer, E. Salomons, and A. Ostfeld. 2019. "A toolbox for assessing the impacts of cyber-physical attacks on water distribution systems." *Environ. Modell. Software* 112 (Feb): 46–51. https://doi.org/10.1016/j.envsoft.2018.11.008.

Taormina, R., S. Galelli, N. O. Tippenhauer, A. Ostfeld, and E. Salomons. 2016. "Assessing the effect of cyber-physical attacks on water distribution systems." In *Proc., World Environmental and Water Resources Congress 2016*, 436–442. Reston, VA: ASCE.

Taormina, R., S. Galelli, N. O. Tippenhauer, E. Salomons, and A. Ostfeld. 2017. "Characterizing cyber-physical attacks on water distribution systems." *J. Water Resour. Plann. Manage.* 143 (5): 04017009. https://doi.org/10.1061/(ASCE)WR.1943-5452.0000749.

Tsiami, L., and C. Makropoulos. 2021. "Cyber-physical attack detection in water distribution systems with temporal graph convolutional neural networks." *Water* 13 (9): 1247. https://doi.org/10.3390/w13091247.

Tuptuk, N., P. Hazell, J. Watson, and S. Hailes. 2021. "A systematic review of the state of cyber-security in water systems." *Water* 13 (1): 81. https://doi.org/10.3390/w13010081.

Urbina, D. I., J. A. Giraldo, N. O. Tippenhauer, and A. A. Cárdenas. 2016. "Attacking fieldbus communications in ICS: Applications to the SWaT testbed." In *Proc., Singapore Cyber-Security Conf. (SG-CRC) 2016*, 75–89. Amsterdam, Netherlands: IOS. https://doi.org/10.3233/978-1-61499-617-0-75.

USEPA. 2015. *Enhancements to the EPANET-RTX (real-time analytics) software libraries*. Technical Brief and Software EPA/600/S-14/441 1. Washington, DC: USEPA, Office of Research and Development.

Valverde-Pérez, B., B. Johnson, C. Wärff, D. Lumley, E. Torfs, I. Nopens, and L. Townley. 2021. *Digital water: Operational digital twins in the urban water sector: Case studies*. Rep. No. 1. London: International Water Association.