Enhancing industrial control system forensics using replication-based digital twins

Marietheres Dietz, Ludwig Englbrecht, Günther Pernul

Chapter 2

ENHANCING INDUSTRIAL CONTROL SYSTEM FORENSICS USING REPLICATION-BASED DIGITAL TWINS

Marietheres Dietz, Ludwig Englbrecht and Günther Pernul

Abstract Industrial control systems are increasingly targeted by cyber attacks. However, it is difficult to conduct forensic investigations of industrial control systems because taking them offline is often infeasible or expensive. An attractive option is to conduct a forensic investigation of a digital twin of an industrial control system. This chapter demonstrates how a forensic investigation can be performed using a replication-based digital twin. A digital twin also makes it possible to select the appropriate tools for evidence acquisition and analysis before interacting with the real system. The approach is evaluated using a prototype implementation.

Keywords: Digital forensics, industrial control systems, digital twins

1. Introduction

Industrial control systems have long life-spans. Since maintenance is performed only a few times a year, industrial control system firmware and software are updated very infrequently [19]. While safe operations are a priority for industrial control systems, adequate levels of security are generally lacking. As a result, industrial control systems are exposed to numerous threats.

When security is breached, an incident response is initiated to understand the situation, mitigate the effects, perform corrective actions and ensure safe operations. A digital forensic investigation provides the best insights into an incident and also assists in the prosecution of the perpetrators. Digital forensic readiness is essential to maximize the ability

to acquire useful evidence and minimize the costs of investigations [30]. An appropriate enterprise-wide maturity level is needed to implement digital forensic readiness for information technology assets [13]. However, an appropriate maturity level is even more difficult to attain by enterprises with operational technology assets such as industrial control systems.

Conducting digital forensic investigations of industrial control systems is challenging because the systems are required to operate continuously for safety and financial reasons. Since the systems cannot be stopped to acquire evidence and conduct forensic analyses, the only alternative is to reduce the shutdown time. This can be accomplished using digital twins of the real systems to identify where evidence resides in the real systems and to select the right tools for extracting evidence before the real systems are stopped. Digital twins replicate the dynamic behavior of their real counterparts. Unlike other state-of-the-art solutions, using digital twins enable industrial control systems to continue to operate while potential attacks are being investigated. Furthermore, unlike cyber ranges and testbeds, digital twins are well suited to digital forensics due to their fidelity, flexibility and two-way communications between the real systems and their digital twins. This chapter demonstrates how forensic investigations of industrial control systems can be performed using replication-based digital twins.

2. Background

This section provides an overview of digital twins, digital twin security and digital forensics.

2.1 Digital Twin

A digital twin is a controversial term with different meanings in different domains [23]. Nevertheless, it can be regarded as a virtual representation of a real object over its lifecycle. Although digital twins have been employed in several domains, including smart cities [14], healthcare [21] and product management [31], they are commonly deployed in the Industry 4.0 paradigm [23], which is the focus of this work.

According to Kritzinger et al. [20], a digital twin is distinguished from other virtual representations (e.g., digital models) by its data flow. A digital model is a manual flow from a real object to a digital object. In contrast, a digital twin has bidirectional automated data flows between the real and virtual worlds [4, 20]. Thus, the digital twin is able to gather state data from its physical counterpart. However, the twin usually contains other asset-relevant data such as specification data [4]. When

enhanced with semantics [26], this data can support various analyses, optimizations and simulations performed by the digital twin [1, 16].

2.2 Digital Twin Security

Several researchers have emphasized that digital twins must have adequate security [16, 26]. However, digital twins can also support industrial control system security [5, 8, 25]. Thus, digital twin security has two perspectives, securing digital twins and using digital twins to implement security. This work focuses on the second perspective and assumes that a digital twin has adequate security. The following security-centered definition of a digital twin is employed in this work [9].

Definition 1. A digital twin is a virtual double of a system during its lifecycle, providing sufficient fidelity for security measures via the consumption of real-time data when required.

Various digital twin modes exist to enable secure operations [5]. While a digital twin provides analytical and simulation capabilities, the replication mode, which supports the exact mirroring of a real system and its states, is relevant to this research [5]. The record and play functionality [10] is unique to the replication mode and is the essence of this work (Definition 1). While research has focused on digital twin security, little, if any, work has focused on using digital twins to support digital forensic investigations despite promising characteristics such as system state mirroring.

2.3 Digital Forensics

Digital forensics involves the identification, collection, preservation, validation, analysis, interpretation and presentation of digital evidence associated with an incident in a computer system [24]. The collection and analysis of digital evidence should be based on a comprehensive process model (see, e.g., [18]).

Internet of Things devices generate many traces during their operation that can be vital to digital forensic investigations. Clearly, there is a need for tools that can support digital forensic investigations of Internet of Things devices. Servida and Casey [27] discuss the challenges involved in examining Internet of Things devices. Although their work does not explicitly deal with industrial control systems, the three main challenges they present are relevant to this work. First, the computing power of the devices is very low and not suitable for performing complex tasks.

Second, most devices have limited embedded memory or an external SD card. Third, it is difficult to extract evidence due to device heterogeneity.

Digital forensic practitioners require considerable expertise, tools and time to completely and correctly reconstruct evidence given the large amounts of data to be processed. A promising approach is to use digital twins. A digital twin can be used to detect an attack. Additionally, it can provide crucial information and insights during digital forensic analysis.

Another challenge to conducting a digital forensic analysis is that the integrity of the data can be compromised during its recovery and analysis. A digital twin can assist in ensuring data integrity. The digital twin of an industrial control system can be examined and the digital forensic process and results verified before performing any actions on the real system.

3. Related Work

The application of digital twins to security and especially forensics has only recently drawn the attention of researchers. The concepts proposed by Eckhart et al. [7, 10] and Gehrmann and Gunnarsson [15] are closely related to this research.

In general, a digital twin is a high-fidelity representation of its real counterpart. Eckhart and Ekelhart [7] were the first to study industrial control system state replication using digital twins; their focus was on reflecting the states of the real system virtually. To avoid large bandwidth overhead, Eckhart and Ekelhart [7] proposed a passive approach that identifies stimuli that alter real-world system states and reproduces them in the digital world.

Gehrmann and Gunnarsson [15] demonstrated that an active approach is suitable for less complex digital twins with moderate synchronization frequencies that do not create overhead; examples are replications of a single industrial control system or an industrial plant with a few industrial control systems. They showed that synchronizing the states of a real system with a digital twin supports active replication. Gehrmann and Gunnarsson also specified security requirements, established a security architecture and implemented secure synchronization between the real object and its digital twin.

According to Eckhart et al. [10], the record and play (replay) mode is a special manifestation of replication using a digital twin. Generally, a digital twin would reflect the real system states at all times. Additionally, restoring the preceding state is enabled, instead of merely replicating the current state that would be lost as soon as the subsequent

state is replicated. The replay mode supports incident management, explicitly tracks infection histories [9] and provides novel functionality with regard to forensics [5]. Therefore, replay is a vital functionality provided by replication using digital twins, especially when applied to digital forensics.

Modern industrial control systems are exposed to security threats because they incorporate commodity hardware and software and operate in highly-interconnected environments. Researchers have attempted to enhance digital forensic capabilities by providing monitoring and logging functionality [3, 33].

The proposed research differs from the research described above in a key way. If an industrial control system is compromised, then its digital twin would exhibit the same malicious behavior as its real counterpart. While this is often considered a downside of replication [7, 15], the proposed research attempts to transform it to an advantage. Specifically, after replicating the exact states of the real system in its digital twin, forensic tools can be applied to conduct deep inspections of a security incident.

Note that this approach differs from traditional intrusion detection and security incident and event management (SIEM) research by focusing on deep inspection and resolution of incidents instead of mere detection. Indeed, the objective is to create a forensically-sound and replicable baseline for forensic analyses of industrial control systems.

The proposed approach advances the state-of-the-art in several respects. Artificial environments such as Mininet can be used to reproduce stimuli (or events) that change the states of industrial control systems or are responsible for the identified activities [7]. Such environments are well-suited to simulating systems and their events [8], but the fidelity of their replication is reduced [7]. Therefore, the proposed approach includes stimuli and events directly from a real system. It is also relevant to note that, while the digital-twin-based security framework of Gehrmann and Gunnarsson [15] considers security abstractly by suggesting a single security analysis component for multiple digital twins in a system, the proposed approach incorporates a separate security analysis module for each control system.

The proposed approach also advances previous work by providing a concrete definition and a proof-of-concept implementation of a digital twin environment for forensic (and/or security) analyses. Instead of investigating how digital twins can protect industrial control systems from external attacks [15], the focus is on digital forensics in the factory domain. The replication-based approach incorporates real communications between multiple control systems. The influences on the logical compo-

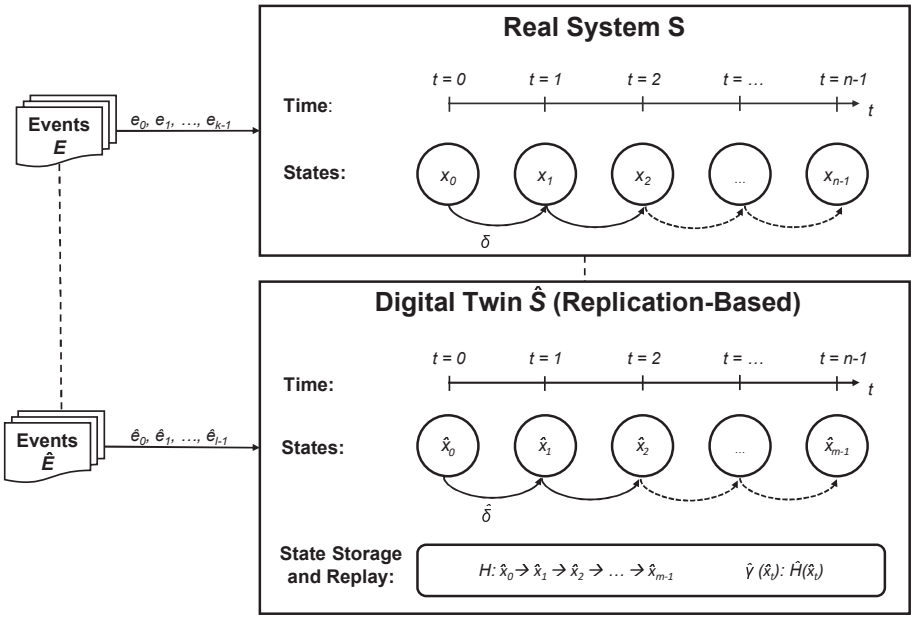

Figure 1. Replication-based digital twin with state storage and replay.

nents and filesystems of industrial control systems are also mimicked. Whereas other work relies on logging and monitoring mechanisms [3, 33], the proposed approach focuses on filesystem changes by making periodic recordings of content in digital twins. This enables the replaying of all the evidence generated by the replicated digital twins.

4. Replication Using Digital Twins

This section provides the theoretical foundations for the proposed approach using digital twins. Four theorems formalize the requirements for replication-based digital twins with replay capabilities.

4.1 Replication and Replay Theorems

Definition 1 above provides the security-centered definition of a digital twin [9]. Formal notions pertaining to state replication with digital twins are provided in [7, 15]. These notions and additional definitions create the basis for digital forensic analyses. Figure 1 presents the proposed replication-based digital twin with replay functionality that can be used for digital forensic analyses.

Sufficient fidelity of digital twins in the replication mode is required to support analyses. This concept is formalized by Theorems 1, 2 and 3.

Theorem 1 (Representation of States). The finite set of states $X = \{x_0, x_1, ..., x_{n-1}\}$ *of a real system is represented in its replication-based digital twin as* $\hat{X} = \{\hat{x}_0, \hat{x}_1, ..., \hat{x}_{m-1}\}$. *A high-fidelity digital twin is replicated as a subset of the real system corresponding to* $\hat{X} \subseteq X$ *where* $m \leq n$. *In an ideal digital twin,* $\hat{X} = X$.

Theorem 2 (Timely Orderliness). To replicate a real system accurately, the concept of time has to be considered. Let $x_t \in X_t$ *represent the real system at time t where the initial state is* x_0. *The digital twin replicates each state* $\hat{x}_t$ *in chronological order so that* $x_0 < x_1 < x_2 < ... < x_{n-1}$. *Time delays may occur between the real system states and digital twin states, but they do not affect digital forensics much because forensic investigations are typically conducted post mortem.*

In addition to having sufficient fidelity, a digital twin must be able to consume real-time data and replay it. This motivates Theorem 3.

Theorem 3 (Integration of Events.) If a system changes from one state to another, certain input data is required, which is referred to as events. Events might occur due to the inner workings of the system or due to its external environment that may not be covered by its digital twin. For example, commands from the system's program are internal events whereas network traffic from other systems are external events. Real events $E = \{e_0, e_1, ..., e_{k-1}\}$ *and the events replicated in the digital twin* $\hat{E} = \{\hat{e}_0, \hat{e}_1, ..., \hat{e}_{l-1}\}$ *express these inputs, where* $\hat{E} \subseteq E$ *and* $l \leq k$. *Furthermore, the transition function* δ *expresses the changes of states in the real system:* $\delta : X \cdot E \rightarrow X$, *i.e.,* $x_{t+1} = \delta(x_t, e_t)$. *Likewise,* $\hat{\delta}$ *expresses the changes of states in its digital twin. Events lead to state changes that in turn may leave traces such as new files or updates of internal values in the real system. As a result of replication, the same traces will be found in the digital twin.*

The highly-desirable replication-based replay functionality imposes additional requirements. This motivates a fourth theorem.

Theorem 4 (Accuracy in Replay). The replay function resets a digital twin to a starting state. Deviations from the previously-observed states of the digital twin should not occur when retrieving its historical events [9]. First, the transition function leading to a subsequent state x' *is repli-*

cated to achieve similar states in the digital twin: $\delta(x, e) = \hat{\delta}(\hat{x}, \hat{e}) \iff x' = \hat{x}'$. *Thus, starting with the initial state, a chain of historic states* $\hat{H} : \hat{x}_0 \mapsto \hat{x}_1 \mapsto \hat{x}_2 \mapsto ... \mapsto \hat{x}_n$ *can be constructed. This chain can be used to reset states and replay the subsequent states in chronological order. The replay function* $\hat{\gamma}(\hat{x}_t) : \hat{H}(\hat{x}_t)$ expresses a reset to state $\hat{x}_t$ and the traversing of the states in chronological order.

Finally, the real system is a deterministic system defined by tuples $S := (X, x_0, E, \delta)$. The digital twin is represented similarly by incorporating state storage and replay functionality $\hat{S} := (\hat{X}, \hat{x}_0, \hat{E}, \hat{\delta}, \hat{H}, \hat{\gamma})$.

4.2 Conceptual Framework

The framework collects data (e.g., network traffic) from a real system that is imported by a high-fidelity digital twin of the real system. The replication ensures that the states of the real system are mirrored by the digital twin. Thus, the digital twin would have digital evidence traces that mirror those in the real system, enabling the digital twin to be analyzed in a forensic investigation.

Forensic investigations, however, require the recording of the current replicated states as well previous states and their associated traces. Furthermore, the states should be accountable and in chronological order (specified in Theorem 2). Therefore, the replication-based digital twin framework also incorporates state storage and replay functionality.

Figure 2 shows the replication-based digital twin framework with state storage and replay functionality. Note that the events $\hat{E}$ in the framework may be external as well as internal.

The framework has four key building blocks: (i) data collection, (ii) digital twin replication, (iii) digital twin state storage and replay, and (iv) digital forensic analysis:

- **Data Collection:** Input data is required to replicate a real system with sufficient fidelity. The data collection component gathers the inputs (events $\hat{E}$) as specified in Theorem 3. The input data may be internal (static) or external (dynamic). Internal data typically can be obtained directly from the real system, such as program code that may alter the system state or commands that are sent in response to external events. They are static because they do not change often and do not exhibit streaming characteristics. In contrast, external data typically corresponds to events that affect the real system. They often occur outside the real system, but within its environment. External data can be characterized as mainly dynamic because it can emerge at any time and in a constant manner

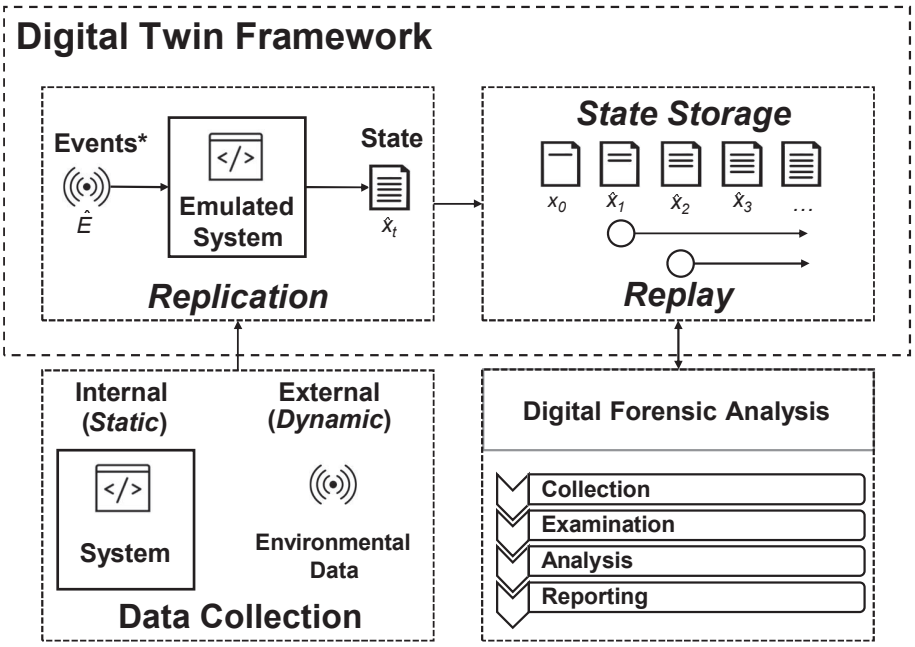

Figure 2. Digital twin framework.

(streaming characteristics). Examples of external data are network traffic from the real system's environment and sensor values.

- **Digital Twin Replication:** The collected data is used to replicate the real system as a digital twin. Internal data such as program code and system configurations are used to emulate the real system. It is important to choose the right technology for replication along with the system itself, desired degree of detail and levels of representation (e.g., network, software and operating system).

  While internal events automatically occur by integrating program code and software, external events have to be input to the system. This enables the replication of system behavior based on internal events and on external stimuli. The greater the amount of data integrated, the more accurate the replication, but a trade-off should be performed between replication fidelity and cost. According to Theorems 1, 2 and 3, the states of the real system are replicated in the desired manner. In each state, different traces are created based on the transition functions. For instance, a file could be created in state A while its content is changed in state B.

- **Digital Twin State Storage and Replay:** This component is vital to digital forensic investigations and elucidation. Stage storage keeps the various system states in chronological order ($\hat{H}$ in Theorem 4), ensuring that they remain accountable. The lateral movement of an attack can be elucidated in a step-by-step manner using system traces. It is critical that the actual content of the generated or modified system data are examined, as initiated by the recorded and replayed network traffic of the real system.

  Replay relies on state storage. It enables the replication of previous states and all their subsequent states (the transition function can be deduced by considering consecutive states). The replay functionality enables the replication to be reset to a desired state and to play all the succeeding states ($\hat{\gamma}(\hat{x}_t)$ in Theorem 4). A digital forensic practitioner can stop the replication at a state of interest, conduct a forensic analysis and continue.

- **Digital Forensic Analysis:** With state storage and replay, a forensic practitioner can gain valuable insights into previous states and processes. There are many ways to use these new opportunities in digital forensics. For example, a practitioner could reset the replication to a certain state using the replay functionality and use analysis tools in turn until a suitable tool is found. The exact replication of events and their storage in chronological order enhances the understanding of an attacker's tactics, techniques and procedures. With each consecutive state, the pattern of the attack might become clearer to the practitioner. An attack that deletes its traces can be investigated by leveraging the replication-based approach with state storage and replay according to the digital forensic process defined by Kent et al. [18].

5. Implementation and Evaluation

The implementation involved a real system comprising a windmill with a programmable logic controller (PLC). The real system and its digital twin had identical Unix operating systems, ran the OpenPLC programmable logic controller software and communicated using TCP/IP.

5.1 Implementation and Experimental Setup

The real system was replicated by passively capturing its network traffic using `tshark` and re-transmitting the traffic to the emulated system. The configuration supports forensic analyses of the real system during execution. This section provides details of the implementation.

The implementation incorporated four main components: (i) data collection, (ii) digital twin replication, (iii) digital twin state storage and replay, and (iv) digital forensic analysis:

- **Data Collection:** The real system and its digital twin employed Unix operating systems running OpenPLC software. OpenPLC supports the IEC 61131-3 standard [17], which defines the software architecture and programming languages for programmable logic controllers. Network traffic created during the operation of the real system was recorded as a PCAP file and re-transmitted to the digital twin.

- **Digital Twin Replication:** The real system was replicated by the digital twin, which executed in a virtual environment running the same Unix operating system and OpenPLC software. All the OpenPLC variables in the real system were refactored as "memory storage" in the digital twin. This facilitated the persistent storage feature of OpenPLC whereby values at various programmable logic controller addresses were saved to disk to provide insights into their changes during programmable logic controller operation.

  The open-source software Polymorph [29] was used to re-transmit network traffic. Polymorph translated the PCAP file data to templates to enable situation-aware interactions. It also facilitated dynamic integration of the components instead of the pure transmission of data. The templates were modifiable for subsequent re-transmission, which was vital because the digital twin had to respond correctly to commands issued in the real system. The digital twin operated (according to the transition function in Theorem 3) as closely as possible to the real system.

- **Digital Twin State Storage and Replay:** The storage and replay component was hosted on a virtual machine (VM), which created a storage snapshot of system state every minute. The snapshots enabled the entire virtual system to be re-created at any point in time. Since data was written and deleted during the execution of OpenPLC, the freed data area in the digital twin could be overwritten, which was problematic. Therefore, a modified version of stateless continuous data protection (CDP) software was employed. Continuous data protection is a technology that continuously captures and stores data changes, enabling data from any point in the past to be recovered [28, 34]. The software also enables the monitoring and restoration of all files generated during system execution.

The `sauvegardeEx` tool [12] was created to gain insights into the generation of files during OpenPLC operation. It is based on `sauvegarde`, an open-source, stateless implementation of continuous data protection [6].

Compared with traditional backup technologies, continuous data protection mechanisms improve the recovery point objective (RPO) metric [22]. The RPO metric defines the time between two successful backups and, thus, the maximum amount of data loss during a successful recovery. The RPO is zero for a system with fully synchronized protection. The RPO metric of zero provided by continuous data protection theoretically allows unlimited recovery points [28]. The `sauvegardeEX` tool implements continuous data protection at the block level (virtual machine snapshots) and at the file level. The virtual machine snapshots were also taken every minute to recover the running system, including volatile RAM memory. With more resources, snapshots could be taken more frequently.

The digital twin framework, which was equipped with the client version of `sauvegardeEx`, sent every file alteration along with the file content to the server. This enabled a specific file to be restored at any point in time and also addressed the problem of overwriting a freed storage area (due to file deletion or update). Instituting this mechanism during OpenPLC execution and replicating the environment via Polymorph ensured that all possible traces were recorded.

- **Forensic Analysis:** Continuous data protection was also exploited to generate data for digital forensic analysis. Continuous data protection technology has not been considered in the digital forensic context, but it is certainly important. In fact, the state storage and replay functionality supported by the modified continuous data protection mechanism enables different forensic tools to be used without the risk of compromising data in the real system or even rendering the data unusable. Indeed, the replication-based approach with state storage and replay completely supports the digital forensic process specified by Kent et al. [18].

5.2 Results and Evaluation

To evaluate the framework, network traffic between OpenPLC (master) and the control unit of the windmill (slave) was captured. The standard Modbus TCP protocol was used for communications. A Python-

based Modbus simulation tool `pyModbusTCP` was used to generate realistic sensor data. A `sensordata.py` script simulated the sensor data for the wind speed around the windmill. Eight registers in the Modbus slave device were relevant. The first four registers contained the current sensor data and the other four indicated the corresponding system status. The sensor data values ranged from one to ten. The sensor values were grouped into three system state categories, green, yellow and red:

- **Values 1-5:** System state is green (Statuscode 200).
- **Values 6-8:** System state is yellow (Statuscode 300).
- **Values 9-10:** System state is red (Statuscode 400).

Pseudorandom sensor values were written to the registers every ten seconds by the Python script. Pseudorandom numbers were used so that changes to the wind speed corresponded to ascending or descending patterns and the values did not vary too much. OpenPLC updated the system status every five seconds based on the sensor values. A slight delay always occurred before the new system status was stored in the registers.

Traffic between OpenPLC and the Modbus slave was transformed to the network templates by Polymorph. This mimicked the external behavior based on traffic content.

By applying Polymorph and `sauvegardeEx` to the replicated system running OpenPLC with persistent storage, all the states of the OpenPLC addresses and related file changes during execution were recorded. All the system artifacts were simultaneously recorded at the digital twin. This enabled the determination of the relationships between the changed states and file content at any point during execution. Since the framework was designed to acquire the actual file content of written files on the hard drive as well as volatile memory content, VirtualBox and its live snapshotting functionality were leveraged.

During the evaluation, 1,432 state changes were observed on the persistent data storage during a five-minute period. The periodic virtual machine snapshots enabled a retrospective analysis of the running system.

Several forensic tools were applied at three points in time during execution. Data used for the evaluation is available at [11]. Table 1 shows an excerpt of the analysis and suitable digital forensic tools. The tools presented in [32] were considered in the evaluation.

An important component of the evaluation was to analyze discrepancies between the real system and its digital twin. This was accomplished by comparing file-level evidence between the real system and its digital

Table 1. Excerpt of the recorded state changes and suitable digital forensic tools.

Timestamp	State Change	File Name	Suitable Tools
2020-09-11 09:16:59.123	File modification	persistent.file	CPLCD
2020-09-11 09:17:01.102	File modification	openplc.db	Bring2lite
2020-09-11 09:17:23.322	File modification	persistent.file	CPLCD
2020-09-11 09:18:00.000	VM snapshot	%/disk.vmdk	Autopsy, CPLCD
2020-09-11 09:18:01.202	File modification	persistent.file	CPLCD
2020-09-11 09:18:01.302	File modification	openplc.db	Bring2lite

twin using the approximate hashing function of Breitinger and Baier [2]. This hashing function was used instead of the SHA-256 hashing function because it provides measures of file similarity. Frequent comparisons of the recorded files helped identify and verify time-event correlations. Comparisons of the 1,432 recorded state changes yielded an average similarity of 98%.

6. Discussion

The proposed approach is easily implemented on architectures with open-source programmable logic controller software. All that is needed is knowledge about the real system and adequate recordings of network traffic.

Although a digital twin adequately replicates a real system, this research has omitted formal measurements of the similarity between them. In order for evidence from a digital twin to be admissible, it is vital their similarity be measured and documented. One approach is to use the synchronization function proposed by Gehrmann and Gunnarsson [15]. However, this mechanism can introduce time differences between the digital twin and its real counterpart. Specifically, system states caused by an attacker in the real system would manifest themselves earlier than in the digital twin.

An interesting possibility is to incorporate control theory in a digital twin. This would make the digital twin a better replication of the real system that would, in turn, contribute to the admissibility of the extracted evidence.

The proposed approach provides recordings of file content at various points in time (via `sauvegardeEx`) and system-wide snapshotting of the running programmable logic controller software (via VirtualBox). These features make it possible to detect and analyze RAM-based malware. However, a limitation is that the implementation employed Unix

and open-source OpenPLC software instead of industrial control system firmware. Although the underlying theory is sound, the open-source implementation would hinder its application in industrial environments.

The implementation of a digital twin for forensic investigations can be expensive. In addition to creating a digital twin and verifying its fidelity, it would be necessary to constantly modify the digital twin and verify that it keeps up with any and all changes made to the real system. This would require digital forensic professionals to have considerable industrial control system expertise, which would be an expensive proposition.

7. Conclusions

As attacks on critical infrastructure assets increase, it is imperative to develop digital forensic techniques targeted for industrial control systems. However, taking an industrial control system offline to conduct a digital forensic investigation is infeasible and expensive. An attractive alternative is to conduct a forensic investigation of a digital twin of an industrial control system. Implementing a digital twin with replication-based state storage and replay enables the acquisition and analysis of file-level evidence. Additionally, the digital twin could be used to select the appropriate forensic tools for evidence acquisition and analysis before interacting with the real system, thereby reducing system downtime when conducting the investigation.

Acknowledgement

This research was partly conducted for the ZIM SISSeC Project under Contract no. 16KN085725 from the German Federal Ministry of Economic Affairs and Energy.

References

[1] S. Boschert, C. Heinrich and R. Rosen, Next generation digital twin, *Proceedings of the Twelfth International Symposium on Tools and Methods of Competitive Engineering*, pp. 209–217, 2018.

[2] F. Breitinger and H. Baier, Similarity preserving hashing: Eligible properties and a new algorithm MRSH-v2, *Proceedings of the Fourth International Conference on Digital Forensics and Cyber Crime*, pp. 167–182, 2012.

[3] C. Chan, K. Chow, S. Yiu and K. Yau, Enhancing the security and forensic capabilities of programmable logic controllers, in *Advances in Digital Forensics XIV*, G. Peterson and S. Shenoi (Eds.), Springer, Cham, Switzerland, pp. 351–367, 2018.

[4] M. Dietz and G. Pernul, Digital twins: Empowering enterprises towards a system-of-systems approach, *Business and Information Systems Engineering*, vol. 62(2), pp. 179–184, 2020.

[5] M. Dietz and G. Pernul, Unleashing the digital twin's potential for ICS security, *IEEE Security and Privacy*, vol. 18(4), pp. 20–27, 2020.

[6] `dugpit`, `cdpfgl`: Continuous Data Protection for GNU/Linux, GitHub (`github.com/dupgit/sauvegarde`), 2021.

[7] M. Eckhart and A. Ekelhart, A specification-based state replication approach for digital twins, *Proceedings of the Workshop on Cyber-Physical Systems Security and Privacy*, pp. 36–47, 2018.

[8] M. Eckhart and A. Ekelhart, Towards security-aware virtual environments for digital twins, *Proceedings of the Fourth ACM Workshop on Cyber-Physical System Security*, pp. 61–72, 2018.

[9] M. Eckhart and A. Ekelhart, Digital twins for cyber-physical systems security: State of the art and outlook, in *Security and Quality in Cyber-Physical Systems Engineering*, S. Biffl, M. Eckhart, A. Lüder and E. Weippl (Eds.), Springer, Cham, Switzerland, pp. 383–412, 2019.

[10] M. Eckhart, A. Ekelhart and E. Weippl, Enhancing cyber situational awareness for cyber-physical systems through digital twins, *Proceedings of the Twenty-Fourth IEEE International Conference on Emerging Technologies and Factory Automation*, pp. 1222–1225, 2019.

[11] L. Englbrecht, DTDFEvaluation, GitHub (`github.com/LudwigEnglbrecht/DTDFEvaluation`), 2021.

[12] L. Englbrecht, `sauvegardeEX`, GitHub (`github.com/LudwigEnglbrecht/sauvegardeEX`), 2021.

[13] L. Englbrecht, S. Meier and G. Pernul, Towards a capability maturity model for digital forensic readiness, *Wireless Networks*, vol. 26(7), pp. 4895–4907, 2020.

[14] M. Farsi, A. Daneshkhah, A. Hosseinian-Far and H. Jahankhani (Eds.), *Digital Twin Technologies and Smart Cities*, Springer, Cham, Switzerland, 2020.

[15] C. Gehrmann and M. Gunnarsson, A digital twin based industrial automation and control system security architecture, *IEEE Transactions on Industrial Informatics*, vol. 16(1), pp. 669–680, 2020.

[16] M. Grieves and J. Vickers, Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems, in *Transdisciplinary Perspectives on Complex Systems*, F. Kahlen, S. Flumerfelt and A. Alves (Eds.), Springer, Cham, Switzerland, pp. 85–113, 2017.

[17] International Electrotechnical Commission, IEC 61131-3:2013 Programmable Controllers – Part 3: Programming Languages, Geneva, Switzerland, 2013.

[18] K. Kent, S. Chevalier, T. Grance and H. Dang, Guide to Integrating Forensic Techniques into Incident Response, NIST Special Publication 800-86, National Institute of Standards and Technology, Gaithersburg, Maryland, 2006.

[19] P. Kieseberg and E. Weippl, Security challenges in cyber-physical production systems, in *Software Quality: Methods and Tools for Better Software and Systems*, D. Winkler, S. Biffl and J. Bergsmann (Eds.), Springer, Cham, Switzerland, pp. 3–16, 2018.

[20] W. Kritzinger, M. Karner, G. Traar, J. Henjes and W. Sihn, Digital twins in manufacturing: A categorical literature review and classification, *IFAC-PapersOnLine*, vol. 51(11), pp. 1016–1022, 2018.

[21] Y. Liu, L. Zhang, Y. Yang, L. Zhou, L. Ren, F. Wang, R. Liu, Z. Pang and M. Deen, A novel cloud-based framework for elderly healthcare services using digital twins, *IEEE Access*, vol. 7, pp. 49088–49101, 2019.

[22] M. Lu and T. Chiueh, File versioning for block-level continuous data protection, *Proceedings of the Twenty-Ninth IEEE International Conference on Distributed Computing Systems*, pp. 327–334, 2009.

[23] E. Negri, L. Fumagalli and M. Macchi, A review of the roles of digital twins in CPS-based production systems, in *Value Based and Intelligent Asset Management: Mastering the Asset Management Transformation in Industrial Plants and Infrastructures*, A. Crespo Marquez, M. Macchi and A. Parlikad (Eds.), Springer, Cham, Switzerland, pp. 291–307, 2020.

[24] G. Palmer, A Road Map for Digital Forensic Research, DFRWS Technical Report, DTR-T001-01 Final, Air Force Research Laboratory, Rome, New York, 2001.

[25] J. Rubio, R. Roman and J. Lopez, Analysis of cybersecurity threats in Industry 4.0: The case of intrusion detection, *Proceedings of the International Conference on Critical Information Infrastructures Security*, pp. 119–130, 2017.

[26] G. Schroeder, C. Steinmetz, C. Pereira and D. Espindola, Digital twin data modeling with automationML and a communication methodology for data exchange, *IFAC-PapersOnLine*, vol. 49(30), pp. 12–17, 2016.

[27] F. Servida and E. Casey, IoT forensic challenges and opportunities for digital traces, *Digital Investigation*, vol. 28(S), pp. S22–S29, 2019.

[28] Y. Sheng, D. Wang, J. He and D. Ju, TH-CDP: An efficient block level continuous data protection system, *Proceedings of the International Conference on Networking, Architecture and Storage*, pp. 395–404, 2009.

[29] `shramos`, Polymorph (v2.0.5), GitHub (`github.com/shramos/polymorph`), 2020.

[30] J. Tan, Forensic readiness: Strategic thinking on incident response, presented at the *Second Annual CanSecWest Conference*, 2001.

[31] F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang and F. Sui, Digital twin driven product design, manufacturing and service with big data, *International Journal of Advanced Manufacturing Technology*, vol. 94(9), pp. 3563–3576, 2018.

[32] T. Wu, F. Breitinger and S. O'Shaughnessy, Digital forensic tools: Recent advances and enhancing the status quo, *Digital Investigation*, vol. 34, article no. 300999, 2020.

[33] K. Yau, K. Chow and S. Yiu, A forensic logging system for Siemens programmable logic controllers, in *Advances in Digital Forensics XIV*, G. Peterson and S. Shenoi (Eds.), Springer, Cham, Switzerland, pp. 331–349, 2018.

[34] X. Yu, Y. Tan, Z. Sun, J. Liu, C. Liang and Q. Zhang, A fault-tolerant and energy-efficient continuous data protection system, *Journal of Ambient Intelligence and Humanized Computing*, vol. 10(8), pp. 2945–2954, 2019.