A Recursive Watermark Method for Hard Real-Time Industrial Control System Cyber-Resilience Enhancement

Zhen Song, *Senior Member, IEEE*, Antun Skuric, and Kun Ji, *Member, IEEE*

***Abstract*— Cybersecurity is of vital importance to industrial control systems (ICSs), such as ship automation, manufacturing, building, and energy automation systems. Many control applications require hard real-time channels, where the delay and jitter are in the levels of milliseconds or less. To the best of our knowledge, no encryption algorithm is fast enough for hard real-time channels of existing industrial fieldbuses and, therefore, made mission-critical applications vulnerable to cyberattacks, e.g., delay and data injection attacks. In this article, we propose a novel recursive watermark (RWM) algorithm for hard real-time control system data integrity validation. Using a watermark key, a transmitter applies watermark noise to hard real-time signals and sends through the unencrypted hard real-time channel. The same key is transferred to the receiver by the encrypted nonreal-time channel. With the same key, the receiver can detect if the data have been modified by the attackers and take action to prevent catastrophic damages. We provide analysis and methods to design proper watermark keys to ensure reliable attack detection. We use a ship propulsion control system for the simulation-based case study, where our algorithm smoothly shuts down the system after attacks. We also evaluated the algorithm speed on a Siemens S7-1500 programmable logic controller (PLC). This hardware experiment demonstrated that the RWM algorithm takes about 2.8 μs to add or validate the watermark noise on one sample data point. As a comparison, common cryptic hashing algorithms can hardly process a small data set under 100 ms. The proposed RWM is about 32 to 1375 times faster than the standard approaches.**

***Note to Practitioners*—It is widely believed that the emerging Internet-of-Things (IoT) technologies will seamlessly connect countless smart devices, profoundly change the industry. Traditionally, field devices within the feedback control loops are isolated from the Internet by secure gateways. In the future, field devices will connect to the Internet in a more direct manner. To the best of our knowledge, no encryption algorithm is fast enough for hard real-time channels of existing industrial fieldbuses and, therefore, made mission-critical applications vulnerable to cyberattacks. We propose a novel recursive watermark (RWM) algorithm for hard real-time control system data integrity validation. This article serves industry practitioners in three ways. First, it is a timely caution to industrial IoT (IIoT) pilot users on the security challenges in real-time channels. Once a compromised edge device is connected to a field device, attackers have unlimited means to jeopardize valuable assets. In this article, we gave an example where attackers may damage shipboard assets by introducing millisecond-level delays. Second, since hard real-time encryption is not available, we propose a detour solution. With the proposed algorithm, even attackers may read the content in the real-time channel, and they cannot change it without being detected. We implemented the real-time RWM algorithms in structured control language (SCL) and tested on a Siemens S7-1500 programmable logic controller (PLC). Third, we provide theoretical analysis as design guidelines for practitioners to set up or customize the RWM algorithm per their specific applications.**

***Index Terms*— Cyber resilience, cyber security, delay attack, industrial control system (ICSs), Internet of Things (IoT), watermark.**

NOMENCLATURE

x	Vector.
x	Scalar.
X	Matrix.
$f(\ )$	Function.
$\mathcal{K}$	Set.
s_w	Variable (lower letter subscript).
T_S	A block in the system diagram (capital letters in the name and the subscript).
$\mathcal{N}(\mu, \sigma^2)$	Normal distribution.
A	Generic data integrity cyberattack event.
A_I	Data injection attack event.
A_R	Replay attack event.
A_D	Delay attack event.
$\mathbb{A}$	Set of all possible data integrity attacks, i.e., $\mathbb{A} \in \{A_I, A_S, A_D, \ldots\}$.
$\overline{A}$	Events without attacks.
k_A	Time instance when the attack happens.
$r[k]$	Reference signal (scalar or vector) at the kth time instance.
d	Discrete-time delay, where d is an integer.
$u[k]$	Control signal.
$y[k]$	Output signal.
$v[k]$	Sensor noise of normal distributions, where $v[k] \sim \mathcal{N}(0, \sigma_v^2)$.

Manuscript received February 28, 2019; revised November 12, 2019; accepted December 4, 2019. This article was recommended for publication by Associate Editor X. Li and Editor B. Vogel-Heuser upon evaluation of the reviewers' comments. *(Corresponding author: Zhen Song.)*

Zhen Song and Kun Ji are with Siemens Corporation, Princeton, NJ 08540 USA (e-mail: zhensong@ieee.org; ji.kun@siemens.com).

Antun Skuric is with the Department of Electric Machines, Drives and Automation (ZESA), University of Zagreb, 10000 Zagreb, Croatia (e-mail: antun.skuric@outlook.com).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2019.2963257

W	Sliding window length to calculate $g[k]$, $g_s[k]$, or $g_c[k]$.
$w[k]$	Actuator noise of normal distributions, where $w[k] \sim \mathcal{N}(0, \sigma_w^2)$.
$s[k]$	Raw sensor measurement. $s[k] = y[k] + v[k]$.
$n_s[k]$	Watermark noise.
$s_w[k]$	Watermarked sensor signal sent by the transmitter (T_S or T_C), where $s_w[k] = s[k] + n_s[k]$.
$s_h[k]$	Potentially hacked signal at the receiver side.
$s_r[k]$	Recovered sensor signal at the receiver side, i.e., if there is no attack, we should have $s_r[k] = s[k]$.
$g[k]$, $g_s[k]$, and $g_c[k]$	Generic detection function, the sensor channel detection function, and the control channel detection function, respectively, where k is the time instance; $g[k] \in \{g_s[k], g_c[k]\}$, and $g[k] \in \mathbb{R}$.
$g_d[k]$, $g_{\mathrm{ds}}[k]$, and $g_{\mathrm{dc}}[k]$	Boolean detection variable for the generic channel, the sensor channel, and the control channel, respectively.
τ	Threshold for the detection function, where $\tau > 0, \tau \in \mathbb{R}$.
D	Bound of the second-order derivative of the sensor or actuator noise.
d	Discrete-time delay introduced by attacks.
$\mathcal{K}$	Watermark key $\mathcal{K}$ is a set of configurable parameters for hashing functions. In one example, $\mathcal{K} = \{\alpha_1, \alpha_2, w, n_p, b, m, a_{m-1}, \ldots, a_0\}$, where α_1 and α_2 are the parameters for watermark noise generation; n_p is the RWM signal resetting duration; w is the window length for the watermark detector; and $b, m, a_{m-1}, \ldots, a_0$ are configuration parameters for pseudorandom number generation.
$h_n(k; \mathcal{K})$, $h_n(k)$	Natural number cryptic hashing function to map a positive integer number k to a real number, i.e., $h_n(k; \mathcal{K}) \in [-0.5, 0.5]$, $k \in \mathbb{N}$, $y \in [0, 1]$, $y \in \mathbb{R}$. Similar to h_r, we may denote it as h_n or $h_n(k)$ under the context when $\mathcal{K}$ is known.
$h_r(x; \mathcal{K})$, $h_r(x)$	Real number cryptic hashing function to map a real number x to a real number, i.e., $h_r(x; \mathcal{K}) \in [-0.5, 0.5]$, and $x \in \mathbb{R}$. When there is no ambiguity, we use $h_r(x)$ for simplicity.
$h_{\mathrm{rw}}(k-1; \mathcal{K})$	Short form for $h_r(s_h[k-1]; \mathcal{K})$.

I. Introduction

A. Motivation

RECENTLY, cybersecurity issues of industrial control systems (ICSs) have attracted much attention [1], [2]. An ICS is a set of industrial controllers and devices, such as a programmable logic controller (PLC), an input/output module (IO), and a human–machine interface (HMI). Today, ICSs are facing different cybersecurity challenges [3] compared with traditional information technology (IT) systems. For example, data integrity attack is of vital importance to the security of an ICS, where attackers manipulate the communication data of the timing of the data to jeopardize high-value targets. The Stuxnet worm issued a data injection attack [4], [5] on the PLCs and damaged over 1000 centrifuges of a nuclear facility. The Maroochy water breach [6] is another example, where the attacker intruded the ICS network and released one million liters of untreated sewage water into a storm-water drain over the course of three months and eventually polluted the local waterways.

In this article, we focus on the data integrity issues of the ICS hard real-time channels. To the best of our knowledge, no encryption algorithm or hardware is fast enough for the hard real-time channels of existing industrial fieldbuses and, therefore, made some critical control applications vulnerable to different data integrity attacks, e.g., replay and data injection attacks. In this article, hard real-time channels are those communication channels with delays and jitters in milliseconds or less, such as the real-time (RT) and isochronous RT (IRT) channels of the Profinet protocol [7]. The notion of the jitter is the range between the maximal and minimal delays. Many industry fieldbus protocols, such as EtherCAT, CAN bus, and Modbus, support hard real-time communications with different technical solutions and different performances. These concepts are illustrated in Fig. 1, where the delay in the standard TCP/UDP protocols is about 100 ms and the jitter range is about 100 ms or more. The Profinet RT channel has a stochastic delay of 10 ms and a jitter of 10 ms. The IRT channel of the Profinet is deterministic and, therefore, has the shortest communication jitters among the three, and the delay is under 1 ms. Keep in mind that the definition of real time in other domains may have different definitions. For instance, delays and jitters for "real-time" video stream can be in the ranges of seconds to hundreds of milliseconds [8], respectively. Delays and jitters in this level may introduce unstable issues of some critical control systems.

A Profinet-based ICS secure network topology is shown on the left-hand side of Fig. 2, where the field-level real-time communication among devices is through the unencrypted RT or IRT channels. The secure gateways can forward the real-time data within the RT and IRT channels to the TCP/IP packets for long-distance communications through the Internet. However, the real-time property is lost once the data passed the security gateways. Although equipped with special encryption chips, the SCALANCE gateway cannot encrypt or decrypt data in the RT or IRT channels. This is not a serious issue in the past several decades, because there seemed to be not enough motivations to connect field devices to the Internet in the

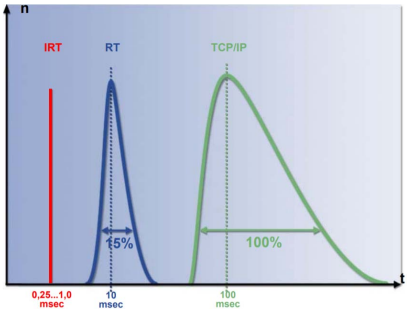


Fig. 1. Distribution of Profinet delays and jitters [7].

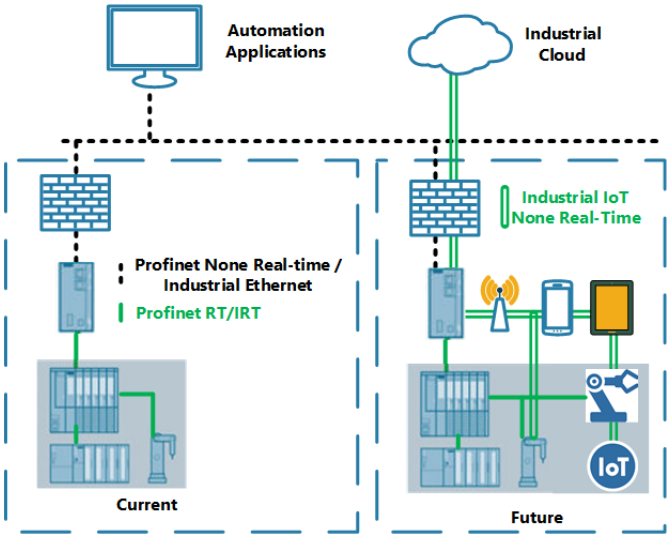


Fig. 2. Current and future ICS secure network topology.

past. With the emerging industrial Internet-of-Things (IIoT) technologies, there are unlimited new use cases to connect the field devices to the Cloud, which expose the unencrypted real-time channels to potential future attacks. As shown on the right-hand side of Fig. 2, operators may bring smart phones, tablets, and mobile devices to the field and connect to PLCs or HMIs directly. Robots and computer numeric control (CNC) machines may connect to the AI algorithms on the Cloud and continuously learn from each other. Smart sensors may be connected to the Cloud for big data analysis. If we do not get them carefully isolated, once a field device is compromised, hackers may read and write the unencrypted RT or IRT channels at will. Note that the network topology in Fig. 2 is applicable to generic industrial automation domains, including manufacturing, shipboard control system, energy automation, and building automation.

B. Potential Attacks on Shipboard Systems

Although the proposed recursive watermark (RWM) framework is applicable to ICS in general, we focus on the shipboard use case in this article. Shipboard control networks are not only high-value targets for cyberattacks but also sensitive to small delays or jitters [9]. Future Navy ships [9] and civilian ships will be electrical-driven [9]–[11] and involve agent-based distributed controls [12]–[14]. In our case study, we will demonstrate a case where millisecond-level delays

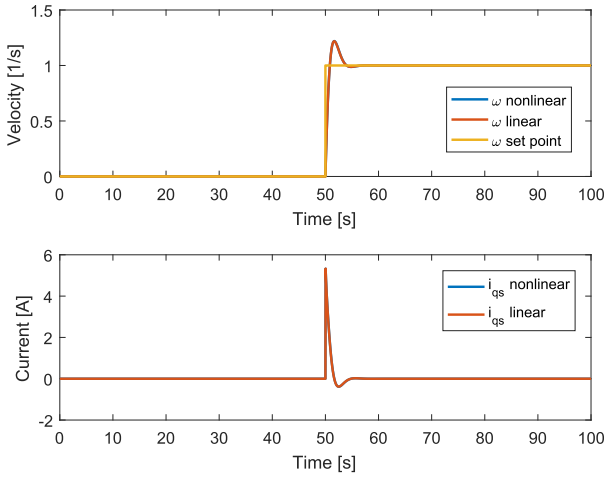


Fig. 3. Simulated shipboard response without delay.

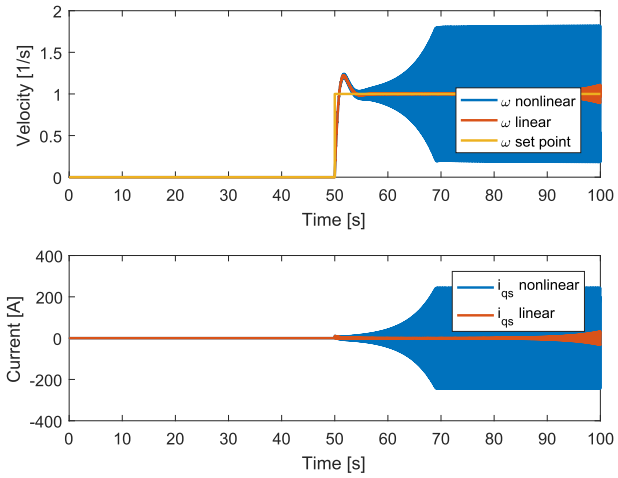


Fig. 4. Simulated shipboard response with a 12.8-ms delay of i_{qs}.

can unstable the ship-generator control loop and potentially damage the propulsion motor or other critical equipment. If this attack happens, a human operator cannot observe the abnormal signal, because the delay is too short. In shipboard systems, the dynamics fall within the range of milliseconds to seconds [15], [16], where small delays under human sensible level may have catastrophic damages. In order to illustrate the impacts of a potential delay attack, we simulate the effects using a shipboard power control system model described in Appendix A. Without delay attacks, the system step response is shown in Fig. 3, where the ω and i_{qs} are the angular speed and current of a shipboard electric propulsion motor, respectively. In our simulation-based study, we introduced small delays in the sensor signal channels of a shipboard system described in Appendix A. As shown in Fig. 4, a short 12.8-ms delay attack is sufficient to make the system unstable and potentially damage the propulsion motor.

C. Literature Review

Cyberattacks on ICSs can be classified in several groups [17], [18]:

1) denial-of-service (DoS) attacks [19], [20];
2) replay and delay attacks [21]–[23];

3) deception attacks [24];
4) control [25] and sensor [1] signal data injection attacks.

Except the DoS attacks, we consider other attacks as data integrity attacks. In replay attacks [22], [23], the attackers record and replay commands to trigger dangerous actions. A deception attacker can manipulate sensors and other physical components of the system, or inject malicious program logic [24]. Data injection attacks are performed by adding false data to the information channel with a purpose to impair the performance of the system [1], [2]. In this article, we focus on the data integrity attacks, because they raise unique challenges for many fieldbus-based ICSs. We do not distinguish *delay attacks* and *replay attacks*. Although the engineering-implementation methods of the attacks are different, the underline mathematical attack models [23] are the same, which we will be presented in Section II-B. One approach to protect against delay attack is based on the system identification theory [26], where the receiver side can estimate the delay and trigger emergency actions to enhance control system resilience [27]. This model-based approach requires knowledge on the system model structure [28] and demands sufficient excitation signals [26] to identify the parameters of nonlinear systems. We consider it a passive method, because it does not require adding control signals to the control system. Another approach is active protection, where we can add special signals for better intrusion detection [29]. The watermark methods belong to this approach.

Traditionally, digital watermark methods are used for hiding information in digital contents, such as photographs [30], audios [31], or videos [32]. The concept was introduced to control the system domain recently [5], [27], [29], [33]. Although using the same "watermark" analogy, the underline mathematical formulations and objectives of the digital content watermarks (video, audio, and image) and control system watermarks are significantly different. For example, it is desirable for a copyright watermark on a photograph to be robust against photograph editing, such that attackers cannot remove the copyright information. The watermark algorithms are often designed based on frequency domain analysis and compared against robustness metrics, such as peak signal-to-noise ratio (PNSR) and structural similarity measure (SSIM) [34]–[36]. For control systems, the ideal watermark should be very *fragile*, such that any tiny adjustment will destroy the watermark. In control systems, copying the sensor signal is not the major concern. Instead, the major threat is on modification of the signal, since it does not introduce a catastrophic result. In this regard, the objective of control system watermark is similar to cryptic hashing methods [37], such as MD5 or SHA algorithms. However, *standard hashing methods do not consider the timing*, which is of vital importance to control systems. Another unique challenge for control system watermark is due to its nature to involve dynamic system physical models. From the engineering perspective, modeling physical systems are expansive and effort-intensive. From the academia perspective, it is important to provide quantitative analysis, based on physics, to define the capability of the control system watermark methods.

To the best of our knowledge, all the existing control system watermark algorithms rely on physics models, mostly linear time-invariant (LTI) system state-space models. After adding different pseudorandom Gaussian noises [27], [38] or pseudorandom Bernoulli package dropout [39] on the top of the sensor or control noises, it is possible to detect the data integrity attacks with a Kalman filter and/or delay estimation techniques [27], [38]. After all, the watermark noise or packet dropout are not completely random. To add even more barriers to the attackers and improve detection performance, some hybrid methods combined packet dropout and watermark noises [39] together. Similarly, another work merged several watermark signals together [19], with different offsets and multiplexed in time, rather than one statistical zero mean noise. Another method to challenge attackers is to add pseudosystem dynamics in the watermark signals [33]. Some work focuses on cyber-resilience enhancement; therefore, the control system will try to operate after the attack by rejecting the attack signals as much as possible [27]. Also to the best of our knowledge, none of the existing work addressed the unencrypted hard real-time channels in industrial fieldbuses. We have not seen report on hardware experiments on mainstream industrial controllers. As a comparison, our proposed RWM method is more aligned with industrial practices. In order to avoid the intensive engineering efforts to build accurate dynamic models for industrial nonlinear systems (see Appendix A), we introduce internal dynamics through the recursive loop within the RWM algorithm. Since the internal dynamics is much faster than the standard plant dynamics, it is safe to allocate the RWM algorithm to different positions in the control system without knowing the detailed plant model. As explained in Section II-G, as far as the max second-order derivative of the plant output is less than a threshold, our RWM method is applicable. We employed the gradient $|\Delta^2 v|$ as a metric for the fragile level.

The rest of this article is organized as follows. Section II introduces the system block diagram and problem formulations. We then explain the watermark adding, removing, and detection algorithms. Section II-G provides detailed analysis on the detection algorithm. In Section III, we implemented the algorithms on Siemens PLC for performance testing. We also simulated a shipboard power system to validate the detection rate for common attacks. Finally, we conclude this article in Section IV. For readers who want to repeat our simulation, we provide a detailed shipboard system power system model in Appendix A.

II. Problem Formulation and Solution

A. Control System Communication Block Diagram

In this section, we first describe the system with attack only in the sensor signal channel, as shown in Fig. 5. Second, we introduce attacks in both the sensor and control signal channels, which is plotted in Fig. 6. Since our RWM algorithm does not rely on physical system dynamics, we can easily deploy the RWM algorithm to any one channel or all the channels.

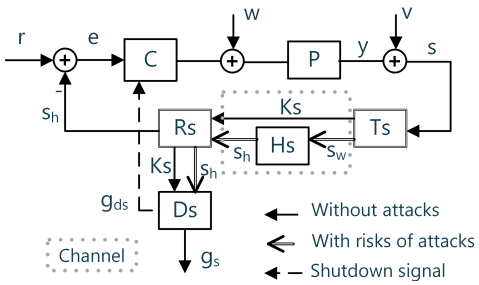


Fig. 5. Profinet ICS diagram with RWM in the sensor channel.

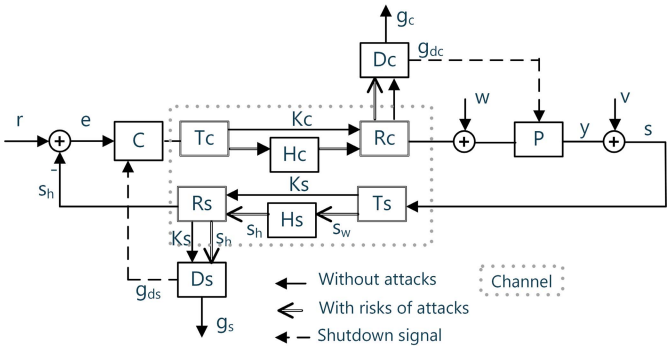


Fig. 6. Profinet ICS diagram with RWM in channels.

In Fig. 5, C is the controller, P is the plant, S is the sensors, H_S is the hacker, T_S is the transmitter in the sensor loop and R_S is the receiver in the sensor loop, and D_S is the detector. r, e, u, y, w, and v are the reference signal, error signal, control system input, output, actuator, and sensor noise, respectively. All these signals are time-variant, e.g., e is $e[t]$, and so on. The RWM key $\mathcal{K}_s$ is transferred from the sensor channel *transmitter*, T_S, to R_S by the encrypted nonreal-time channel. Once R_S receives the key, T_S can send watermarked data, s_w, to the *receiver*, R_S, by the real-time channels, i.e., RT or IRT channels. Attackers may issue data integrity attacks on s_w within the H_S block, and the output is s_h. Since the watermark noise is very small, R_S can feed the watermarked signal s_h directly the controller C. In order to check the data integrity, R_S also forwards $\mathcal{K}_s$ and s_h to the *detector* D_S to validate the data integrity, where some delay due to computation is not a concern, because it is out of the control loop. The float number g_s is the attack indicator, and g_{ds} is the Boolean variable to trigger the shutdown process for the control C. If any data integrity attack is detected by D_S, the whole controller will shut down smoothly.

Since the RWM algorithm is decoupled from the plant's dynamics, it is simple to embed another RWM block within the control signal channel, as shown in Fig. 6. The dotted-line box includes the control and sensor signal channels, where T_C, R_C, H_C, $\mathcal{K}_c$, D_C, g_c, g_{dc}, and w are the control channel transmitter, receiver, attacker, RWM key, detector, detector signal, detector decision signal, and actuator noise, respectively.

The name recursive is due to the unique *internal dynamics* design of the RWM algorithm, as shown in Fig. 7, where z^{-1} is the delay to introduce the internal state and internal dynamics of the RWM algorithm, k is the discrete-time instance, and y, v, and s_w are the system output, sensor noise, watermarked signal, respectively. The block in Fig. 7 is either T_s or T_c in Fig. 5 or 6. The motivation for internal dynamics is to make the RWM detector independent of plant dynamics; therefore, engineers can easily place the RWM algorithm in either the control channel or the sensor channel without the effort to model the physics of the plant accurately. It is a nontrivial task to model a real plant such as a ship power chain (as shown in Appendix A) or a manufacturing process. While some works [5] employ a Kalman filter to estimate the physical states of the plant, our proposed RWM detector estimates the internal state of the RWM algorithm. Since the sampling rate of the Profinet RT/IRT channels is very high, the induced delay of the RWM algorithm is short. Therefore, the internal state of the RWM algorithm changes much faster than typical physical variables in the plant. They are decoupled and placed at different positions on the system.

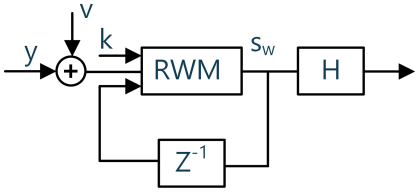


Fig. 7. RWM algorithm internal dynamics.

B. Attack Models

To issue data injection attacks, attackers must be able to read and write data. In generic data injection attacks, the attacker can inject any false data $u_a[k]$ to the channel.

Definition 1 (Data Injection Attack A_I): Given k_1 and k_2 the beginning and ending discrete-time instances of the attack; s_h is the received signal; s_w is the watermarked signal; and the data injection attack function f_I is defined as

$$s_h[k] = f_I(s_w[k]) = \begin{cases} u_a[k], & \text{if } k_1 < k < k_2 \\ s_w[k], & \text{else} \end{cases}$$

where $u_a[k]$ can be any signal defined by the attacker and $u_a[k] \neq s_w[k]$.

Now, we define shift attack, which is a subset of the data injection attack, i.e., $A_S \in A_I$. Instead of injecting the random signal, the attacker can shift the original signal by a value $h_a[k]$.

Definition 2 (Data Shift Attack A_S): Given k_1, k_2, s_h, and s_w in Definition 1, the data shift attack function f_S is defined as

$$s_h[k] = f_S(s_w[k]) = \begin{cases} s_w[k] + h_a[k], & \text{if } k_1 < k < k_2 \\ s_w[k], & \text{else} \end{cases}$$

where $h_a[k]$ is a small real value defined by the attacker.

Even if the communication channel is encrypted, an attacker can store the signal from a network device, such as a switch, and replay the same signal after certain delay. Some control systems are sensitive to even millisecond-level delays. The delay attack A_D is formulated as follows.

Definition 3 (Delay Attack A_D): Given k_1, k_2, s_h, and s_w in Definition 1 and d is the delay, the delay attack function

f_D is defined as

$$s_h[k] = f_D(s_w[k]) = \begin{cases} s_w[k-d], & \text{if } k_1 < k < k_2 \\ s_w[k], & \text{else.} \end{cases}$$

The delay d can be either a constant or a function of time.

In this article, we use $\overline{A}$ as the event without attack, i.e., when $s_h[k] = s_w[k]$. We use A to represent the attack event, where $A \in \{A_I, A_S, A_D, \ldots\}$. From the math definitions, A_I is the most generic set, i.e., $A_S \in A_I$ and $A_D \in A_I$. Of course, the software implementation of different attacks is different. Therefore, we discuss them separately in verbal discussions, yet often aggregate the cases to one form during algorithm discussions.

C. Hashing Functions

In Appendix B, we present a short tutorial on an existing pseudorandom integer sequence-generation function $h(k)$, which generates integers between 0 and $b-1$.

Definition 4 [Pseudorandom Number Generator h(k)]: Given a set of configuration parameters $\{b, m, a_0, a_1, \ldots, a_{m-1}\} \in \mathbb{Z}$, $h(k)$ is a pseudorandom number and $h(k) \in [0, b-1]$, where $k \in \mathbb{N}$.

Based on $h(k)$, we construct an integer hashing function $h_n(k)$ with a normalized real output.

Definition 5 [Integer Hashing $h_n(k)$]: Given a natural number k, h_n maps k to a real number between -0.5 and 0.5: $h_n(k) \in [-0.5, 0.5], k \in \mathbb{Z}$.

Similarly, we can construct a hashing function $h_r(x)$, which maps the real number x to another pseudorandom number between -0.5 and 0.5.

Definition 6 [Real Hashing $h_r(x)$]: Given a real number x, h_r maps x to a real number between -0.5 and 0.5: $h_r(x) \in [-0.5, 0.5], x \in \mathbb{R}$.

We can easily construct h_n and h_r functions based on the h function, so we do not present the details due to the limited space. In Appendix A, we provide a brief summary on the h function. The pseudorandom number-generation process cannot be easily reversed [40], and we found it fast enough for hard real-time control purposes. There are many potential choices for h_n and h_r functions, and they should meet the following criterion of the RWM algorithm.

1) *Not Reversible:* The ideal hashing functions are cryptic functions, i.e., one cannot derive k from the value of $h_n(k)$.
2) *Fast:* The forward mapping calculation must be fast enough to meet the RT performance requirement.

D. Recursive Watermark-Generation Algorithm

While the concept of watermark for control system has been discussed before [21], the proposed RWM method introduced two or more channels: one nonreal-time encrypted channel and one or more real-time yet unencrypted channels. As aforementioned, mainstream industry fieldbuses, such as Profinet, offer these multi-channel settings. The RWM-generation algorithm resides at the sender side with two communication channels connected to the receiver side. Different from the watermark approaches, which use one public channel only, the proposed RWM uses the encrypted channel to transfer the watermark *key* of the unencrypted channels. With this advantage, one novel feature of the proposed RWM method is the ability to identify the exact time of the attack event due to the new recursive watermarking algorithm.

Algorithm 1 Offline Watermark-Generation Algorithm

Data: Input Data is $\mathcal{K}$, $s[k]$, $s_w[k-1]$, s_{max}, s_{min}
Result: Output is $s_w[k]$
Offline: during the initialization phase
From $\mathcal{K}$, generate two lookup table T_1 and T_2, where T_1 has n_n elements and $T1[k] = h_n(k; \mathcal{K}), k \in [0, n_n]$; T_2 has n_r elements and $T2[k] = h_r(k; \mathcal{K}), k \in [0, n_p]$.
Online: Given $s[k]$
$n = k \bmod n_p$
if $n = 0$ **then**
  $s_w[k] = s[k]$
end
else
  $h_n(k) = T_1[n]$
  $i = \left\lfloor \dfrac{s_w[k-1] - s_{min}}{s_{max} - s_{min}} \right\rfloor$
  $h_{rw}(k-1) = T_2[i]$
  $s_w[k] = s[k] + \alpha_1\ h_n[n] + \alpha_2\ h_{rw}(k-1)$
end
return $s_w[k]$;

The motivation of the watermark-generation algorithm is to add the watermark noise on the original signal. With the same set of parameters that the sender used, i.e., the key, the receiver can easily validate the self-consistency. The attacker does not have the key and cannot validate the self-consistency. If the attacker modifies the watermarked signal at certain time, the self-consistency breaks since that time. Thanks to the recursive feature, any attack on the watermarked signal at one time instance will propagate to the following samples.

There are unlimited methods to define the RWM function h_{rw}. For example, it can be a high-order recursive function as $h_{\text{rw}} = 0.7\ s_w[k-1] + 0.2\ s_w[k-2] + 0.1\ s_w[k-3]$. In this article, we only discuss the first-order RWM due to its simplicity, where $h_{\text{rw}} = h_r(s_w[k-1])$.

Definition 7 (First-Order RWM Signal): Given the key $\mathcal{K}$ a set of constant parameters, the first-order watermarked signal $s_w[k]$ is defined as

$$s_w[k] := \begin{cases} s[k], & \text{if } k \bmod n_p = 0 \\ s[k] + \alpha_1 h_n(k; \mathcal{K}) + \alpha_2 h_r(s_w[k-1]; \mathcal{K}), & \text{else} \\ \{\alpha_1, \alpha_2\} \geq 0. \end{cases}$$

For simplicity, it is also denoted as

$$s_w[k] := s[k] + \alpha_1 h_n(k) + \alpha_2 h_{\text{rw}}(k-1). \tag{1}$$

For presentation purpose, in (1), we refer $h_r(s_w[k-1])$ as $h_{\text{rw}}(k-1)$. Moreover, we refer the difference between $s_w[k]$ and $s[k]$ as the watermark noise $n_s[k]$. For the first-order RWM, we have $n_s[k] := \alpha_1\ h_n(k) + \alpha_2\ h_{\text{rw}}(k-1)$. The definition is illustrated in Fig. 8, where h_n and h_{rw} are reset to 0 after every n_p samples, when $s[k] = s_w[k]$.

Algorithm 2 Online Watermark-Generation Algorithm

Data: Input data is $\mathcal{K}$, $s[k]$, $s_w[k-1]$,s_{max}, s_{min}
Result: Output is $s_w[k]$
Online: T_1, T_2 are arrays. Given $s[k]$
$n = k \bmod n_p$
if $n = 0$ **then**
  $s_w[k] = s[k]$
end
else
  $T_1[k] = h_n(k; \mathcal{K})$
  $h_n(k) = T_1[k]$
  $i = \left\lfloor \dfrac{s_w[k-1] - s_{min}}{s_{max} - s_{min}} \right\rfloor$
  $T_2[i] = h_r(i; \mathcal{K})$
  $h_{rw}(k-1) = T_2[i]$
  $s_w[k] = s[k] + \alpha_1\ h_n[n] + \alpha_2\ h_{rw}(k-1)$
end
return $s_w[k]$

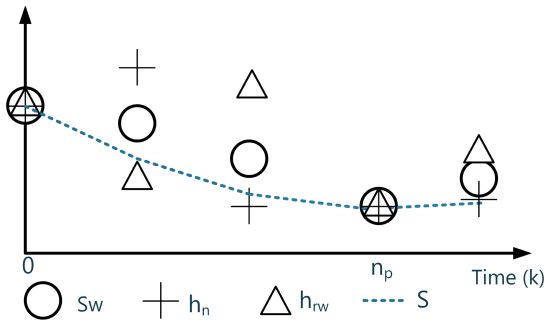


Fig. 8. RWM-generation method.

For the engineering implementation, we have the choice to calculate the pseudorandom numbers either offline or online. The first method requires more memory yet demands less computations. The second method is opposite. Hereby, we define the first and second methods in Algorithms 1 and 2, respectively. In the algorithms, the mod function is the modulus operation and $\lfloor x \rfloor$ is the floor operator, i.e., the largest integer no larger than x.

E. Watermark-Removing Algorithm

Since the transmitter intentionally added noises of very small magnitude, it is acceptable to feed the watermarked signal to the actuator directly. However, if users do not want the small performance degradation due to the additional noise, there is an option to remove the watermark, which can be conducted by the D_S block in Fig. 5. As shown in Fig. 8, $s_w[k] = s[k]$ and n_p is a periodic reset event, when the receiver can start the watermark-removing process.

Definition 8 (Recovered Signal for the First-Order RWM Signal):

$$s_r[k] = \begin{cases} s_h[k], & \text{if } k \bmod n_p = 0 \\ s_h[k] - \alpha_1 h_n[k](n; \mathcal{K}) - \alpha_2 h_r(s_h[k-1]; \mathcal{K}), & \text{else} \\ \alpha_1 + \alpha_2 = 1, \quad \{\alpha_1, \alpha_2\} \in [0, 1]. \end{cases}$$

The signals s_r and s_h are the recovered and received signals, respectively. The process is written in Algorithm 3.

Algorithm 3 Watermark-Removal Algorithm

Data: $\mathcal{K}$, $s_h[0], s_h[1], \cdots, s_h[k]$, time k
Result: $s_r[k]$
$n = k \bmod n_p$ **if** $n = 0$ **then**
  ▶ *It is the reset event*
  $s_r[k] = s_h[k]$
else
  ▶ *Not the reset event*
  $s_r[k] = s_h[k] - \alpha_1\ h_n(n; \mathcal{K}) - \alpha_2 h_r(s_h[k-1]; \mathcal{K})$
end
return $s_r[k]$

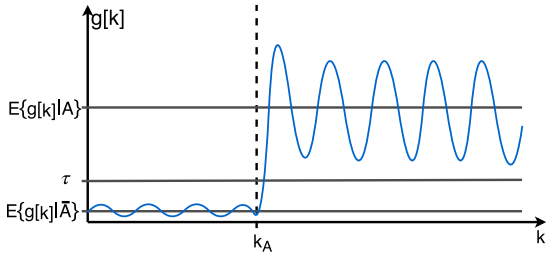


Fig. 9. Notations for χ^2 detector outputs.

F. Watermark-Validation Algorithm

The receiver validates whether the received signal has the proper watermark signal or not. If yes, the receiver believes there is no cyberattack. If not, the receiver will raise an alarm and trigger the shutdown process. In order to validate the watermark signal, the receiver needs to compare whether the noise in the watermarked signal follows the distribution of the watermarked signal based on the expected pseudorandom noise. Since the sender already told the receiver key parameters, $\mathcal{K}$ of the watermark noise, the receiver can employ the χ^2 test to check if the received watermarked signal $s_w[k]$ follows the expected.

The objective of the detection algorithm is illustrated in Fig. 9, where x-axis is the time and y-axis is the detector signal, and $g[k]$ can be either $g_s[k]$ or $g_c[k]$ in Fig. 5 or 6. Before the attack, i.e., $\bar{A}$, the detector signal $g[k]$ is smaller than that after the attack. Due to the unavoidable sensor or actuator noises, v or w, $g[k]$ is not completely 0 before the attack. With properly designed watermark signal, we can set up a threshold τ, such that $\mathbb{E}\{g_k|\overline{A}\} < \tau < \mathbb{E}\{g_k|A\}$, where $\mathbb{E}\{g_k|\overline{A}\}$ and $\mathbb{E}\{g_k|A\}$ are the expectations of $g[k]$ without and with attacks, respectively. The scenario is shown in Fig. 9.

The receiver validates the received signal, $s_h[k]$, by using the χ^2 test to check if the estimated noise $\hat{v}[k]$ is still subject to the known normal distribution. The χ^2 test is widely used for system malfunction analysis [41] and validation [19], [21]

$$e[k] = s_r[k] - \hat{s}_r[k|k-1]$$

where $e[k]$ is the forecast error based on $s_r[k-1]$ and the vector $\hat{s}[k|k-1]$ is the forecast on $s_r[k]$

$$\begin{aligned} \hat{s}_r[k|k-1] &\approx s_r[k-1] + \Delta s_r[k-1] \\ &= 2s_r[k-1] - s_r[k-2] \\ e[k] &= s_r[k] - 2s_r[k-1] + s_r[k-2]. \end{aligned} \tag{2}$$

Algorithm 4 Watermark-Validation Algorithm

Data: $\mathcal{K}$, $s_r[0], s_r[1], \cdots, s_r[k]$
Result: $g[k]$, $g_d[k]$
$g[k] = 0$
for $i = k - W + 1$, $i \leq k$ **do**
  $e[i] = s_r[i] - 2s_r[i-1] + s_r[i-2]$
  $g[i] = g[i] + e[i]\sigma_v^{-1}e[i]$ ▶ *Calculation χ^2 detector*
  $i = i + 1$
end
if $g[k] \geq \tau$ **then**
  $g_d[k] = 1$; ▶ *Shut down control*
else
  $g_d[k] = 0$; ▶ *Continue*
end

In the detector D_S, a null hypothesis test is conducted, assuming that there is no attack, $\overline{A}$; therefore, $s_r[k] = s_w[k]$ and $s_r[k] = s[k] = y[k]+v[k]$. Since $v[k]$ is subject to normal distribution $v[k] \sim \mathcal{N}(0, \sigma_v^2)$, we can apply the χ^2 test with W sample points

$$g[k] := \frac{1}{W} \sum_{i=k-W+1}^{k} e[i]\sigma_v^{-1}e[i]. \tag{3}$$

We have $\mathbb{E}\{g_k\} = \lim_{W\to+\infty} g[k]$.

If $g[k]$ is not bigger than a threshold τ, we have good confidence that there is no attack. If $g[k]$ is too big, the detector will send the shutdown signal to the controller. We have $g_d[k]$ the Boolean shutdown signal defined as

$$g_d[k] = \begin{cases} 1, & g[k] \geq \tau \\ 0, & g[k] < \tau. \end{cases}$$

G. Analysis

We need to prove two fundamental properties for the sensor channel RWM algorithm.

1) Without attack and ignoring the sensor noise v, the original signal can be recovered, e.g., given the RWM key, $\mathcal{K}$, we can recover the sensor signal s from the received sensor signal s_r.
2) When the absolute value of the second derivative of the sensor noise is no larger than a value D, it is possible to find a threshold τ to discriminate the attack and no-attack event.

Due to the limited space, we will not prove the similar properties of the control channel. We can easily replace sensor noise v by the actuator noise w, and the proof is immediate. Property 1 shows that the RWM algorithm has no information loss when there is no sensor noise. In engineering practices, system noises (sensor and actuator noises) are unavoidable; therefore, property 2 gives a practical guideline on the capability of the RWM algorithm. As aforementioned, we intentionally introduce fast internal dynamics of the RWM algorithm to be decoupled from the physical system dynamics. The sensor noises depending on hardware may have a high-frequency component and overwhelm the watermark noise n_s. For engineering practices, we need to quantify a metric to measure the noise and ensure that it will not jeopardize the watermark noise; otherwise, the $\mathbb{E}\{g_k|A\}$ and $\mathbb{E}\{g_k|\overline{A}\}$ in Fig. 9 are not significantly different.

1) Recover Sensor Signal:

Theorem 1: Given $s_w[k]$ defined in Definition 7 and $s_r[k]$ defined in Definition 8, when there is no attack, we can recover the sensor signal, i.e., $s_r[k] = s[k]$.

Proof: Under the no-attack condition, i.e., $\overline{A}$, $s_h[k] = s_w[k]$. At the time $k = 0$, based on the definitions, it is trivial to have

$$s_r[0] = s_h[0] = s_w[0] = s[0].$$

When $k > 0, k < n_p$, since $k \bmod n_p \neq 0$, we have

$$\begin{aligned} s_h[1] &= s_w[1] = s[1] + \alpha_1\ h_n(1;\mathcal{K}) + \alpha_2\ h_r(s[0];\mathcal{K}) \\ s_r[1] &= s_h[1] - \alpha_1\ h_n(1;\mathcal{K}) - \alpha_2\ h_r(s[0];\mathcal{K}) \\ s_r[1] &= s[1] \\ &\vdots \\ s_r[k] &= s[k], k < n_p. \end{aligned}$$

When $k = n_p$, the watermark signal is reset. Therefore

$$\begin{aligned} s_r[n_p] &= s_h[n_p] = s_w[n_p] = s[n_p] \\ s_r[n_p] &= s[n_p]. \end{aligned}$$

Similarly, it is easy to show that

$$\begin{aligned} s_r[n_p+1] &= s_h[n_p+1] - \alpha_1\ h_n(1;\mathcal{K}) - \alpha_2\ h_r(s_h[n_p];\mathcal{K}) \\ s_r[n_p+1] &= s_h[n_p+1] - \alpha_1\ h_n(1;\mathcal{K}) - \alpha_2\ h_r(s[n_p];\mathcal{K}) \\ s_r[n_p+1] &= s[n_p+1] + \alpha_1\ h_n(1;\mathcal{K}) + \alpha_2\ h_r(s[n_p];\mathcal{K}) \\ &\quad -\alpha_1\ h_n(1;\mathcal{K}) - \alpha_2\ h_r(s[n_p];\mathcal{K}) \\ s_r[n_p+1] &= s[n_p+1] \\ &\vdots \\ s_r[n_p+k] &= s[n_p+k],\ k < n_p. \end{aligned}$$

In summary, when there is no attack $\overline{A}$, we have $s_r[k] = s[r]$ for $k \geq 0$. □

2) Expected Value Without Attack: We will derive the expectations of detection $\mathbb{E}\{g_k\}$ when there is no attack and then compare those expectations with the attacks. If they are significantly different, the detector can distinguish them giving sufficient number of samples.

Theorem 2: Given sensor noise $v \sim \mathcal{N}(0, \sigma_v^2)$, D is the bound of the max absolute value of the second-order derivative of the sensor signal, i.e., $|\Delta^2\ y| \leq D$, then the expectation of the χ^2 testing is g_k and $\mathbb{E}\{g_k|\overline{A}\} = 6\sigma_v$.

Proof: As proved in Theorem 1, without attack $s_r[k] = s[k]$ and $s[k] = y[k] + v[k]$ by definition, the watermarking residue and the forecast error can be written as

$$\begin{aligned} e[k] &= y[k] - 2y[k-1] + y[k] + v[k] - 2v[k-1] + v[k-2] \\ e[k] &= \Delta^2\ y[k] + \Delta^2\ v[k]. \end{aligned}$$

System noise par of the detector $\Delta^2\ v[k]$ follows $\Delta^2\ v[k] \sim \mathcal{N}(0, 6\sigma_v^2)$, where Δ^2 is the second-order discrete derivation operator and is defined as

$$\Delta^2\ y[k] = y[k] - 2y[k-1] + y[k-2] = y''[k] \cdot T_s^2.$$

Here, T_s is a sampling time of the channel and $y''[k]$ is the second-order gradient for the continuous-time system. When the no-attack scenario is considered, the expectation of the $g[k]$ is as follows:

$$\begin{aligned}\mathbb{E}\{g_k|\overline{A}\} &= \frac{1}{\sigma_\upsilon}\mathbb{E}\{e^2[k]\} \\ &= \frac{1}{\sigma_\upsilon}(\mathbb{E}\{[\Delta^2\ y[k]]^2\} + \mathbb{E}\{[\Delta^2\ \upsilon[k]]^2\})\end{aligned}$$

where signals y and υ are independent. The expectation of the second derivation of the signal y naturally tends to be zero: $\mathbb{E}\{\Delta^2\ y[k]\} = 0$.

With the assumption that the system noise υ is white, different time instances of the noise $\upsilon[k], \upsilon[k-1], \upsilon[k-2]$ are independent as well and their cumulative expectation becomes

$$\begin{aligned}\mathbb{E}\{[\Delta^2\ \upsilon[k]]^2\} &= \mathbb{E}\{\upsilon^2[k]\} + 4\mathbb{E}\{\upsilon^2[k-1]\} + \mathbb{E}\{\upsilon^2[k-2]\} \\ &= 6\sigma_\upsilon^2.\end{aligned}$$

Therefore, the expectation of the no-attack χ^2 detector is

$$\mathbb{E}\{g_k|\overline{A}\} = \frac{1}{\sigma_\upsilon}\left(6\sigma_\upsilon^2\right) = 6\sigma_\upsilon.$$

□

Comments: This theorem quantifies the acceptable sensor noise of the RWM algorithm. While the magnitude of noise is important to the robustness metrics of the digital content watermarks [34]–[36], both frequency and magnitude are related to the noise-rejection capability of the RWM algorithm. Too many high-frequency components in the sensor noise lead to larger $\Delta^2\ \upsilon[k]$, which demands a larger threshold for the χ^2 detector. Since $\Delta^2\ \upsilon[k] \sim \mathcal{N}(0, 6\sigma_\upsilon^2)$, the value of the noise component $\Delta^2\ \upsilon[k]$ of the detector g_k follows the Gaussian normal distribution, and its maximal value V can be determined by satisfying the equation:

$$\Pr(|\Delta^2\ \upsilon[k]| < V\ ) = \Phi(V) - \Phi(-V) = \gamma$$

where Pr() is the probability function of the Gaussian distribution and $\Phi()$ is its cumulative distribution function. The desired probability is γ with a recommended value of 0.95. The scalar V is the maximum value of $\Delta^2\ \upsilon[k]$, which can be calculated from the sensor measurement for a given γ. Finally, with the calculated V and known limit D, the threshold value τ can be set as

$$\tau \geq \max\{g_k|\overline{A}\} = \frac{1}{\sigma_\upsilon}(D+V)^2.$$

3) Expected Value Under Data-Injection Attacks: If the attacker injects signal $u_a[k]$, as shown in Definition 1, we can estimate the expected value of the χ^2 testing as well.

Theorem 3: Under the data injection attack in Definition 1, the expected χ^2 testing result is

$$\mathbb{E}\{g_k|A_I\} = \frac{1}{\sigma_\upsilon}\mathbb{E}\{[\Delta^2\ u_a]^2\} + \frac{1}{\sigma_\upsilon}(6\mu_n + 6\mu_{\text{rw}})$$

where $\mu_n = \alpha_1^2\mathbb{E}\{h_n^2(k)\} and \mu_{\text{rw}} = \alpha_2^2\mathbb{E}\{h_{\text{rw}}^2(k)\}$.

Proof: Let us start from the simple case when $k < n_p$ and the attack event happens before n_p. Based on Definitions 8 and 1, the recovered signal under A_I is

$$\begin{aligned}s_h[k] &= u_a[k] \\ s_r[k] &= u_a[k] - \alpha_1 h_n(k) - \alpha_2 h_{\text{rw}}(u_a[k-1]) \\ e[k] &= \Delta^2 s_r[k].\end{aligned} \tag{4}$$

The expectation of the second derivation of the u_a is unknown, because it is introduced from the attacker. On the other hand, h_n and h_{rw} have well-known distributions and variances, and their expectations can easily be calculated

$$\begin{aligned}\mathbb{E}\{e^2[k]\} &= \mathbb{E}\{[\Delta^2\ u_a[k]]^2\} + \alpha_1^2\mathbb{E}\{[\Delta^2\ h_n(k)]^2\} \\ &\quad + \alpha_2^2\mathbb{E}\{[\Delta^2\ h_{\text{rw}}(u_a[k-1])]^2\} \\ &= \mathbb{E}\{[\Delta^2\ u_a[k]]^2\} + 6\mu_n + 6\mu_{\text{rw}}.\end{aligned}$$

Therefore, the expectation of the χ^2 detector during the data injection attack is

$$\mathbb{E}\{g_k|A_I\} = \frac{1}{\sigma_\upsilon}\mathbb{E}\{[\Delta^2\ u_a]^2\} + \frac{1}{\sigma_\upsilon}(6\mu_n + 6\mu_{\text{rw}}).$$

□

4) Expected Value Under Delay Attacks: Due to the limited space, we study one delay attack, where the delay is of one discrete-time instance, $d = 1$. Other scenarios can be analyzed with the same technique.

Theorem 4: Given delay attacks in Definition 3 and the delay is 1

$$\mathbb{E}\{g_k|A_D\} = \frac{1}{\sigma_\upsilon}(6\sigma_\upsilon^2 + 20\mu_n).$$

Proof: In the case of one sample delay attack, the received signal becomes

$$\begin{aligned}s_h[k] &= s_w[k-1] \\ s_r[k] &= s_w[k-1] - \alpha_1 h_n(k) - \alpha_2 h_{\text{rw}}(s_w[k-2]) \\ e[k] &= s_r[k-1] - 2s_r[k-2] + s_r[k-3] \\ &\quad + \alpha_1(-h_n[k] + 3h_n[k-1] - 3h_n[k-2] + h_n[k-3]).\end{aligned}$$

Therefore, the expectation of the watermarking residue $e[k]$ is as follows:

$$\begin{aligned}\mathbb{E}\{e^2[k]\} &= \mathbb{E}\{[\Delta^2\ y[k-1]]^2\} + \mathbb{E}\{[\Delta^2\ \upsilon[k-1]]^2\} \\ &\quad + \alpha_1^2\mathbb{E}\{[-h_r(k) + 3h_r(k-1) \\ &\qquad - 3h_r(k-2) + h_r(k-3)]^2\}.\end{aligned}$$

Since random noises h_n are independent of different discrete-time samples k and previously discussed $\mathbb{E}\{\Delta^2\ y[k]\} = 0$, the χ^2 detector expectation is as follows:

$$\mathbb{E}\{g_k|A_D\} = \frac{1}{\sigma_\upsilon}\left(6\sigma_\upsilon^2 + 20\mu_n\right).$$

□

III. Experiments

A. Simulation Experiment

In the simulation experiment, we demonstrated the performance of the RWM algorithm based on an attack toward a sinusoid signal. Then, we applied the algorithm to a shipboard

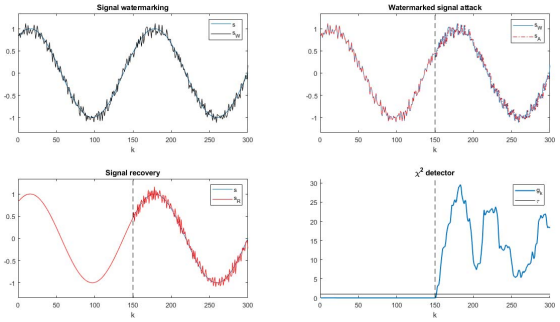


Fig. 10. Watermark authentication for the two-sample delay attack, window-size $w = 10$, variance $v = 1$, and noise distribution uniform with magnitude 0.1.

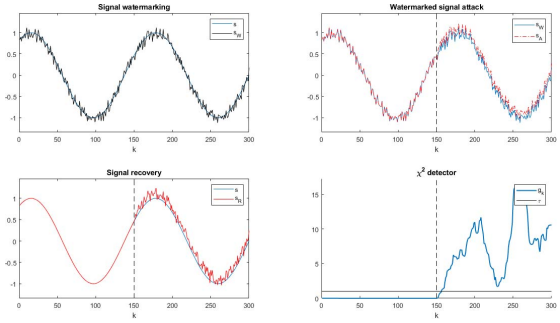


Fig. 11. Watermark authentication for data injection attack, window-size $w = 10$, variance $v = 1$, and noise distribution uniform with magnitude 0.1.

power system data integrity validation. The detailed physics model of the shipboard system is described in Appendix A. Without protection, an attacker may damage the generator using delay attacks. We used the RWM algorithm to detect the attacks in time and shut down the system smoothly.

1) Sinusoid Signal: Using simple sinusoid signals as examples, we compare the effects of watermark overlay, detection, and removing algorithms. On the top left of Figs. 10 and 11, there are comparisons between the raw sensor signal $s[k]$ and the watermarked signal $s_w[k]$ without an attack. The noise looks random, but it contains hidden information that is not observable to our eyes. The top-right plots of Figs. 10 and 11 are the comparisons of the watermarked signal before and after the delay attack and the data injection attack, respectively. The watermarked signal after the one sample delay looks very similar to the signal before the attack in Fig. 10. After the watermark being removed, the recovered signal shows abnormal yet small noise after the event of the attack, which is shown at the bottom left of Fig. 10. The χ^2 test identified the broken consistency of the after-attack signal at the bottom right of Figs. 10 and 11. We can easily identify the attack time with a threshold using the χ^2 test.

2) Shipboard System: The MATLAB Simulink model of the shipboard system is shown in Fig. 12, where we assume that the attacker can manipulate the sensor measurement on ω: the rotation speed, T_e: the torque, and λ_{dr}: the motor flux. When the system is working properly, the step response is shown in Fig. 3. As mentioned in Appendix A, the system is sensitive to small delays not observable by humans. Under the delay attack, the system is unstable, as shown in Fig. 4. With the proposed RWM algorithm, we can quickly detect

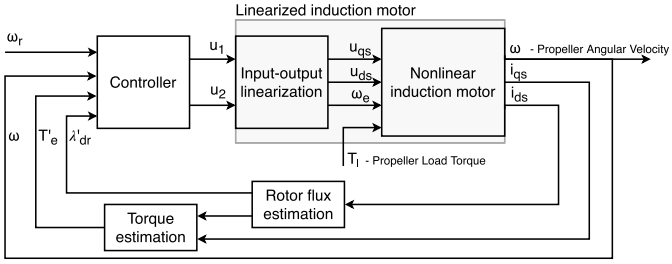


Fig. 12. MATLAB simulation model of the shipboard control system.

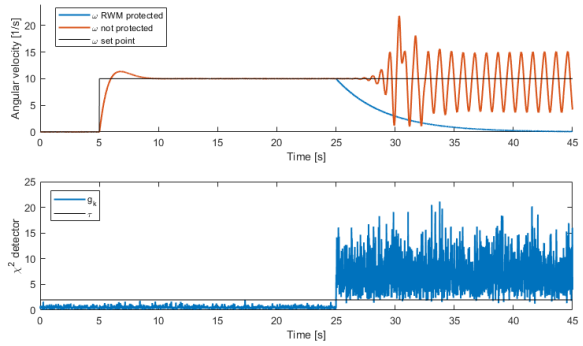


Fig. 13. Shipboard system response with a delay attack at the 25th second.

the delay and data injection attacks, as shown in Fig. 13 and Fig. 14, respectively. The χ^2 test within the watermark validator identified the attack immediately and given the system enough time to smoothly shut down the motor and safely reduce the ω, as shown by the blue curves in Fig. 13. Without the RWM protection, the ω signal became unstable within 1–5 s, which may damage the motor or other assets.

B. Hardware Experiment

The key motivation of the RWM algorithm is to provide hard real-time data integrity checking on resource-limited embedded devices. It is important to validate the speed of the algorithm on mainstream ICS, e.g., PLCs. From the RWM algorithm pseudocode in Section II-D, it is obvious that the complexity of the algorithm is rather low: once the pseudorandom numbers are generated, the complexity is $\mathcal{O}(n)$.

We evaluated the performance on Siemens S7-1500 PLC, as shown in Fig. 15. The S7-1500 PLC is a high-end PLC in the Simatic PLC product family, and it is widely used in industrial automations, such as ship control, factory automation, energy automation, and transportation systems. We implemented the RWM algorithms in structured control language (SCL) [42], which is one of the IEC 61131 standard PLC languages.

We implemented Algorithms 1 and 2 and evaluated their performance on the S7-1500 PLC. Algorithm 1 is faster yet requires more memory, while Algorithm 2 requires less memory but a bit more computation time. For both cases, the hashing table has 10^5 entries. We evaluated their performance in three categories.

1) Pseudorandom Number Generation: Both the offline and online watermark-generation algorithms (Algorithms 1 and 2) are based on cryptic pseudorandom number generations. Many industrial automation systems have limited random

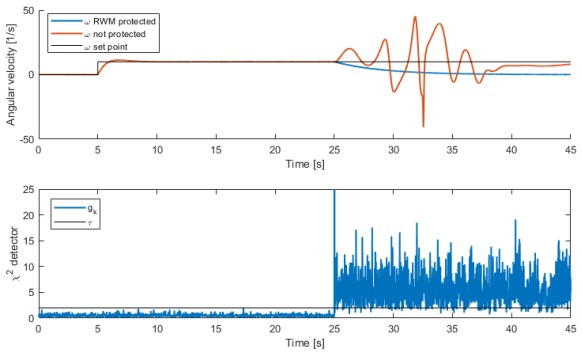


Fig. 14. Shipboard system response to inject attack at the 25th second.

TABLE I

10^5 ELEMENT LOOKUP TABLE CREATION TIME IN ms

Base b	5	5	5	5	2	5	31	53
Degree m	4	5	6	9	4	4	4	4
min[ms]	13	14	14	17	11	13	13	13
max[ms]	24	24	20	31	18	21	21	21
average[ms]	16	18	18	20	16	17	16	16

TABLE II

χ^2 DETECTOR ALGORITHM RESULTS IN ms/10^3 ITERATIONS

Window size W in time	5	10	20	50
min[ms]	2	5	11	28
max[ms]	9	13	22	42
average[ms]	5	9	14	32
avg per operation[μs]	5	9	14	32

number-generation features, e.g., the default random number-generation function in the SCL language can only output pseudorandom numbers within a fixed range from −32 768 to +32 767. Since we need a fast algorithm with a configurable range of the output, we implemented a matured pseudorandom sequence-generation algorithm [40] in the SCL. More details of the algorithm are presented in Appendix III-B. In Table I, we list the computation time to generate 10^5 random natural numbers that are no larger than $b^m - 1$. For better accuracy, we averaged the time values after about 100 experiments. The speed is fast enough for hard-real-time applications.

2) Watermark Validation: The performance of the χ^2 detector has been evaluated using different window sizes W. As shown in Table II, all the elapsed time values are expressed in ms/10^3 iterations of the detector. The average values are calculated after 100 experiments.

3) Hashing Algorithm: Hashing algorithm performance is compared for both hashing functions h_r and h_n. Both hashing functions use the lookup tables in memory and calculate the table index in relation to the hashing input. Illustrated in Table III, the results are expressed in ms/10^6 iterations per experiment and averaged after 100 experiments.

Now, we compare the performance of the RWM algorithm with two standard hashing algorithms, i.e., MD5 and SHA256. The experiment performed hashing operation for 10^5 float numbers. The experiments were conducted in two separate cases:

1) 10^5 consequent hashing operations;
2) 10^3 hashing operations for 100 bundled numbers.

TABLE III

HASHING PERFORMANCE IN ms/10^6 ITERATIONS

Hashing function	h_r	h_n
min[ms]	6	1
max[ms]	12	13
average[ms]	8	8

TABLE IV

HASHING PERFORMANCE FOR DIFFERENT METHODS

Experiment		RWM	MD5	SHA256
Case 1	Time[ms]	16	12500	22000
	Per Num [μs]	0.16	125	220
	Speedup Ratio	-	781	1375
Case 2	Time[ms]	16	500	950
	Per Num [μs]	0.16	5	9.5
	Speedup Ratio	-	32	60

Fig. 15. Shipboard resilient control experimental system.

All the values are averaged after five experiments. The speedup ratio in Table IV is calculated by dividing the computation time of the alternative methods (MD5 and SHA256) by the time of the RWM algorithm.

IV. CONCLUSION AND FUTURE WORKS

Today, there are no encryption or data integrity protections on the real-time channels in the fieldbus protocols against the data integrity attacks. As the emerging IIoT trend is connecting more field devices directly to the Internet or external networks, it is important to protect the real-time channels without compromising the control system performance. In this article, we proposed a set of RWM algorithms for hard real-time data integrity validation on resource-limited embedded systems to protect Profinet, a standard fieldbus protocol, from attacks at the real-time channels. Tested on a mainstream PLC, i.e., Siemens S7-1500, the algorithm is 32–1375 times faster than the standard approach. In a shipboard power system case study, we demonstrated that the algorithm can detect fatal delay and data injection attacks. In order to limit the

TABLE V
INDUCTION MOTOR PARAMETERS

Parameter	Symbol	Value	Unit
Rated power	P	400	hp
Pole pairs	p	4	-
Moment of inertia	J	4.1	kgm^2
Stator resistance	r_s	7.6	$m\Omega$
Rotor resistance	r_r	4.6	$m\Omega$
Stator inductance	L_{ls}	0.15	mH
Rotor inductance	L_{lr}	0.15	mH
Mutual inductance	L_M	5.2	mH

engineering efforts, the proposed algorithm is decoupled from the plant dynamics and can be easily applied to different systems without the system dynamics model. In the future, we plan to investigate high-order RWM systems.

APPENDIX A
SHIPBOARD POWER SYSTEM

For readers' convenience, we briefly present the simulation model used in this article. The simulations in this article are conducted based on a simplified all-electric propulsion ship model [43]. We developed an MATLAB Simulation model, i.e., Fig. 12 with parameters in Table V. In the system, there are two control variables, u_1 and u_2, and four outputs ω, λ_{dr}, i_{ds}, and i_{qs}. Given the input u_1, T_e the electrical torque, and T_l the load torque, the angular velocity ω of an induction motor [44] can be calculated from a decoupled [45] model

$$T_e = \frac{\frac{3}{2}\frac{p}{2}\frac{L_M}{L_M+L_{lr}}}{sL_1 + r_s}u_1 \tag{5}$$

$$L_1 = L_{ls} + \frac{L_M L_{lr}}{L_M + L_{lr}} \tag{6}$$

$$\omega = \frac{1}{sJ}\frac{p}{2}(T_e - T_l) = K_\omega \frac{1}{s}(T_e - T_l). \tag{7}$$

The rotor flux λ_{dr} is controlled by u_2, where

$$\lambda_{\text{dr}} = \frac{1}{s\frac{L_M+L_{ls}}{L_M} + \frac{r_s}{L_M}}u_2. \tag{8}$$

The stator current i_{ds} and i_{qs} are controlled by u_1 and u_2

$$i_{\text{ds}} = \frac{\lambda_{\text{dr}}}{L_M} \tag{9}$$

$$i_{\text{qs}} = \frac{1}{\frac{3}{2}\frac{p}{2}\frac{L_M}{L_{lr}+L_M}}\frac{T_e}{\lambda_{\text{dr}}}. \tag{10}$$

The currents i_{ds} and i_{qs} can be measured, but rotor flux λ_{dr} has to be estimated, e.g., using the following equation:

$$\hat{\lambda}_{\text{dr}} = \frac{L_M}{s\frac{L_{lr}}{r_r} + 1}i_{\text{ds}} \tag{11}$$

where $\hat{\lambda}_{\text{dr}}$ is the estimated rotor flux.

This system is designed using the cascade of PI controllers, with parameters shown in Table VI. Figs. 16 and 17 show the control system diagram, where circled values (ω, i_{qs}, i_{ds}) represent sensor measurement and values with $'$ sign are the estimated states (T_e', λ_{dr}').

Table VI shows the parameters of the PI controllers used in this article. PI_T and PI_ω are the controllers in the torque T_e and the shaft velocity ω cascades, and PI_λ is controller in the flux loop.

TABLE VI
PI CONTROLLERS PARAMETERS

Controller	P value	I value
PI_λ	1	0.1
PI_T	1	1
PI_ω	10	10

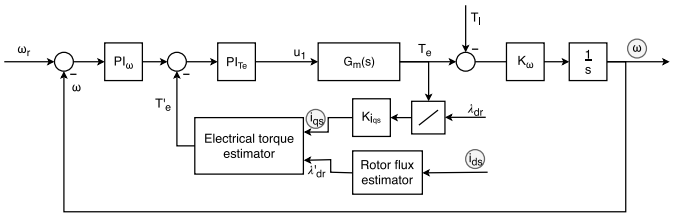


Fig. 16. Block diagram mechanical subsystem closed-loop control.

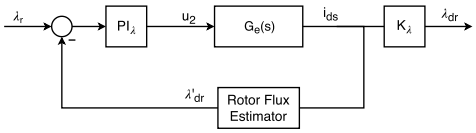


Fig. 17. Block diagram electrical subsystem closed-loop control.

APPENDIX B
PSEUDORANDOM NUMBER GENERATION

As mentioned in Section III-B, the cyberresilience of the RWM algorithm is based on a pseudorandom number generator, which derives a sequence of white noises that attackers cannot guess. Out of numerous candidates, we used a matured algorithm [40]. The algorithm can quickly generate white noise sequence of length n with autocorrelation functions of either 1 or $-1/n$. If $R[i]$ is the autocorrelation function at the ith position, we have $R[0] = 1$, $R[i] = -1/n, i \in [1, n-1]$. We need to set up base b and degree m for the algorithm, where the sequence length n is defined by m with $n = b^m - 1$. The base b used by the mod-b operator is defined as

$$x \oplus_b y = (x + y) \bmod b.$$

In this article, we use $h(x)$ to generate pseudorandom numbers for the proposed RWM algorithm

$$h(x) = x^m \oplus_b a_{m-1}x^{m-1} \oplus_b \ldots \oplus_b a_1 x \oplus_b a_0$$

where a_i are the configurable parameters such that $\{a_{m-1}, \ldots, a_1, a_0\} \in [0,\ b-1]$.

ACKNOWLEDGMENT

The authors would like to thank Office of Naval Research (ONR), Naval Research Laboratory (NRL), and the Comprehensive Cyber Industry control system Resilient Security (CCIRS)/REsilient CONtrol system (RECON) Team for their supports. They would also like to thank Dr. A. Canedo, Dr. M. Kang, J. Luo, Dr. R. Craven, Dr. S. Mertoguno, M. Veldink, Dr. P. Kumar, and A. Varró for the in-depth discussions.

REFERENCES

[1] S. Mclaughlin *et al.*, "The cybersecurity landscape in industrial control systems," *Proc. IEEE*, vol. 104, no. 5, pp. 1039–1057, May 2016, doi: 10.1109/JPROC.2015.2512235.

[2] T. H. Morris and W. Gao, "Industrial control system cyber attacks," in *Proc. 1st Int. Symp. ICS SCADA Cyber Secur. Res. (ICS-CSR)*, Leicester, U.K., Sep. 2013, pp. 22–29. [Online]. Available: https://ewic.bcs.org/content/ConWebDoc/51165

[3] A. A. Cardenas, S. Amin, and S. Sastry, "Secure control: Towards survivable cyber-physical systems," in *Proc. IEEE 28th Int. Conf. Distrib. Comput. Syst. Workshops (ICDCS)*, Jun. 2008, pp. 495–500, doi: 10.1109/ICDCS.Workshops.2008.40.

[4] N. Falliere, L. O. Murchu, and E. Chien. (2011). W32. Stuxnet Dossier, Security Response 5.6. Symantec Corp. [Online]. Available: https://preview.tinyurl.com/cpvzv2a

[5] Y. Mo, S. Weerakkody, and B. Sinopoli, "Physical authentication of control systems: Designing watermarked control inputs to detect counterfeit sensor outputs," *IEEE Control Syst.*, vol. 35, no. 1, pp. 93–109, Feb. 2015, doi: 10.1109/MCS.2014.2364724.

[6] J. Slay and M. Miller, "Lessons learned from the Maroochy water breach," in *Proc. Crit. Infrastruct. Protection (IFIP)*, vol. 253, E. Goetz and S. Shenoi, Eds. Boston, MA, USA: Springer, 2007, ch. 6, pp. 73–82, doi: 10.1007/978-0-387-75462-8_6.

[7] Siemens. *PROFINET Real-Time Communication*. [Online]. Available: http://www.profibus.org.pl/index.php?option=com_docman&task=doc_view&gid=28

[8] F. Liu and H. Koenig, "A survey of video encryption algorithms," *Comput. Secur.*, vol. 29, no. 1, pp. 3–15, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404809000698

[9] T. V. Vu, D. Gonsoulin, F. Diaz, C. S. Edrington, and T. El-Mezyani, "Predictive control for energy management in ship power systems under high-power ramp rate loads," *IEEE Trans. Energy Convers.*, vol. 32, no. 2, pp. 788–797, Jun. 2017.

[10] S. Jothibasu and S. Santoso, "New electric shipboard topologies for high resiliency," *IEEE Trans. Power Syst.*, vol. 33, no. 3, pp. 2975–2983, May 2018.

[11] R. A. Dougal, "Design tools for electric ship systems," in *Proc. IEEE Electr. Ship Technol. Symp.*, 2005, pp. 8–11, doi: 10.1109/ESTS.2005.1524645.

[12] K. Huang, D. A. Cartes, and S. K. Srivastava, "A multiagent-based algorithm for ring-structured shipboard power system reconfiguration," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 37, no. 5, pp. 1016–1021, Sep. 2007, doi: 10.1109/TSMCC.2007.900643.

[13] K. P. Logan, "Intelligent diagnostic requirements of future all-electric ship integrated power system," *IEEE Trans. Ind. Appl.*, vol. 43, no. 1, pp. 139–149, Jan. 2007, doi: 10.1109/TIA.2006.886993.

[14] K.L. Butler-Purry, "Multi-agent technology for self-healing shipboard power systems," in *Proc. 13th Int. Conf., Intell. Syst. Appl. Power Syst.*, 2005, doi: 10.1109/ISAP.2005.1599264.

[15] N. Doerry, "Naval power systems: Integrated power systems for the continuity of the electrical power supply," *IEEE Electrific. Mag.*, vol. 3, no. 2, pp. 12–21, Jun. 2015.

[16] J. V. Amy, Jr., and N. Doerry, "Design considerations for a reference MVDC power system," in *Proc. SNAME Maritime Conv.*, 2016.

[17] L. Zhang and H. Zhang, "A survey on security and privacy in emerging sensor networks: From viewpoint of close-loop," *Sensors*, vol. 16, no. 4, p. 443, Apr. 2016. [Online]. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4850957/

[18] T. Morris and W. Gao, "Classifications of industrial control system cyber attacks," in *Proc. 1st Int. Symp. ICS SCADA Cyber Secur. Res.*, Leicester, U.K., Sep. 2013.

[19] J. Rubio-Hernán, Cicco, and J. García-Alfaro, "Revisiting a watermark-based detection scheme to handle cyber-physical attacks," in *Proc. 11th Int. Conf. Availability, Rel. Secur. (ARES)*, Aug. 2016, pp. 21–28, doi: 10.1109/ARES.2016.2.

[20] Z. Yang, P. Cheng, and J. Chen, "Learning-based jamming attack against low-duty-cycle networks," *IEEE Trans. Dependable Secure Comput.*, vol. 14, no. 6, pp. 650–663, Nov. 2017, doi: 10.1109/TDSC.2015.2501288.

[21] Y. Mo and B. Sinopoli, "Secure control against replay attacks," in *Proc. 47th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2009, pp. 911–918, doi: 10.1109/ALLERTON.2009.5394956.

[22] C. Fang, Y. Qi, P. Cheng, and W. X. Zheng, "Cost-effective watermark based detector for replay attacks on cyber-physical systems," in *Proc. 11th Asian Control Conf. (ASCC)*, Dec. 2017, pp. 940–945, doi: 10.1109/ASCC.2017.8287297.

[23] A. Teixeira, D. Pérez, H. Sandberg, and K. H. Johansson, "Attack models and scenarios for networked control systems," in *Proc. ACM 1st Int. Conf. High Confidence Netw. Syst. (HiCoNS)*, New York, NY, USA, 2012, pp. 55–64, doi: 10.1145/2185505.2185515.

[24] H. Zhang, P. Cheng, J. Wu, L. Shi, and J. Chen, "Online deception attack against remote state estimation," *IFAC Proc. Volumes*, vol. 47, no. 3, pp. 128–133, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667016416046

[25] Y. Mo and B. Sinopoli, "Integrity attacks on cyber-physical systems," in *Proc. ACM 1st Int. Conf. High Confidence Netw. Syst. (HiCoNS)*, New York, NY, USA, 2012, pp. 47–54, doi: 10.1145/2185505.2185514.

[26] L. Ljung, *System Identification*, L. Ljung, Ed., 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1999. [Online]. Available: http://portal.acm.org/citation.cfm?id=293154

[27] A. Sargolzaei, K. K. Yen, M. N. Abdelghani, S. Sargolzaei, and B. Carbunar, "Resilient design of networked control systems under time delay switch attacks, application in smart grid," *IEEE Access*, vol. 5, pp. 15901–15912, 2017, doi: 10.1109/ACCESS.2017.2731780.

[28] S. Bjorklund and L. Ljung, "A review of time-delay estimation techniques," in *Proc. 42nd IEEE Int. Conf. Decis. Control*, vol. 3, Dec. 2003, doi: 10.1109/CDC.2003.1272997.

[29] P. R. K. B. Satchidanandan, "Dynamic watermarking: Active defense of networked cyber physical systems," *Proc. IEEE*, vol. 105, no. 2, pp. 219–240, Feb. 2017, doi: 10.1109/JPROC.2016.2575064.

[30] V. M. Potdar, S. Han, and E. Chang, "A survey of digital image watermarking techniques," in *Proc. 3rd IEEE Int. Conf. Ind. Inform. (INDIN)*, Aug. 2005, pp. 709–716.

[31] J. Bajpai and A. Kaur, "A literature survey—Various audio watermarking techniques and their challenges," in *Proc. 6th Int. Conf.-Cloud System Big Data Eng. (Confluence)*, Jan. 2016, pp. 451–457.

[32] M. Asikuzzaman and M. R. Pickering, "An overview of digital video watermarking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 9, pp. 2131–2153, Sep. 2018.

[33] J. Rubio-Hernan, L. De Cicco, and J. Garcia-Alfaro, "On the use of watermark-based schemes to detect cyber-physical attacks," *EURASIP J. Inf. Secur.*, vol. 2017, no. 1, p. 8, Jun. 2017, doi: 10.1186/s13635-017-0060-9.

[34] A. Mishra, A. Rajpal, and R. Bala, "Bi-directional extreme learning machine for semi-blind watermarking of compressed images," *J. Inf. Secur. Appl.*, vol. 38, pp. 71–84, Feb. 2018. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2214212617303162

[35] A. R. A. M. R. Bala, "A novel fuzzy frame selection based watermarking scheme for MPEG-4 videos using bi-directional extreme learning machine," *Appl. Soft Comput. J.*, pp. 603–620.

[36] R. G. A. M. S. Jain, "A semi-blind HVS based image watermarking scheme using elliptic curve cryptography," *Multimedia Tools Appl.*, vol. 77, no. 15, p. 19235–19260, Aug. 2018, doi: 10.1007/s11042-017-5351-0.

[37] S. Bakhtiari, R. Safavi-Naini, and J. Pieprzyk. (1995). *Cryptographic Hash Functions: A Survey*. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.56.8428

[38] Y. Mo, R. Chabukswar, and B. Sinopoli, "Detecting integrity attacks on SCADA systems," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 4, pp. 1396–1407, Jul. 2014, doi: 10.1109/TCST.2013.2280899.

[39] S. Weerakkody, O. Ozel, and B. Sinopoli, "A Bernoulli–Gaussian physical watermark for detecting integrity attacks in control systems," in *Proc. 55th Annu. Allerton Conf. Commun., Control Comput. (Allerton)*, Oct. 2017, pp. 966–973.

[40] F. J. MacWilliams and N. J. A. Sloane, "Pseudo-random sequences and arrays," *Proc. IEEE*, vol. 64, no. 12, pp. 1715–1729, Dec. 1976, doi: 10.1109/PROC.1976.10411.

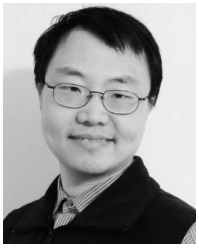
[41] R. K. Mehra and J. Peschon, "Correspondence item: An innovativeapproach to fault detection and diagnosis in dynamic systems," *Automatica*, vol. 7, no. 5, pp. 637–640, Sep. 1971, doi: 10.1016/0005-1098(71)90028-8.

[42] Siemens. *Programming Guideline for S7-1200/S7-1500*. Accessed: Feb. 27, 2019. [Online]. Available: https://preview.tinyurl.com/http-www1-siemens-cz-ad-curr

[43] B. Zahedi and L. E. Norum, "Modeling and simulation of all-electric ships with low-voltage DC hybrid power systems," *IEEE Trans. Power Electron.*, vol. 28, no. 10, pp. 4525–4537, Oct. 2013.

[44] P. C. Krause and C. H. Thomas, "Simulation of symmetrical induction machinery," *IEEE Trans. Power App. Syst.*, vol. PAS-84, no. 11, pp. 1038–1053, Nov. 1965.

[45] G.-S. Kim, I.-J. Ha, and M.-S. Ko, "Control of induction motors for both high dynamic performance and high power efficiency," *IEEE Trans. Ind. Electron.*, vol. 39, no. 4, pp. 323–333, Aug. 1992.



Zhen Song (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from Utah State University, Logan, UT, USA, in 2007.

From 2006 to 2018, he was with Siemens Corporate Technology, Princeton, NJ, USA, where he has been a Principal Investigator of government research projects sponsored by Department of Defense (DOD), DOE, Naval Research Laboratory (NRL), Office of Naval Research (ONR), and New York State. He developed the sensor system of ODIS robot, which has been deployed to Iraq and Afghanistan to protect U.S. soldiers from car bombs. He is also a Senior Data Scientist with Siemens Smart Infrastructure, Austin, TX, USA.

Antun Skuric received the B.S.E.E. and M.S.E.E. degrees from the Faculty of Electrical Engineering and Computing, University of Zagreb, Zagreb, Croatia, in 2014 and 2017, respectively.

He is currently a Research Associate with the Mechatronics Laboratory, Department of Electric Machines, Drives and Automation (ZESA), University of Zagreb. He is also co-founder of a start-up, GuitarFriend, University of Zagreb. His main research interest is optimal control applications on mechatronic systems.

Kun Ji (Member, IEEE) received the Ph.D. degree in mechanical engineering from Texas A&M University, College Station, TX, USA, in 2006.

Since then, he has been with Siemens Corporate Technology, Princeton, NJ, USA. His research interests focus on resilient control and distributed sensor networks, grid automation and industrial automation systems, building control and energy efficiency, and cyber-physical systems.