Proc. of International Conference on Artificial Intelligence, Computer, Data Sciences and Applications (ACDSA 2025)
7-9 August 2025, Antalya-Türkiye

Towards Quantifying Information Leakage of Timing Attacks in Cyber-Physical Systems

Sepehr Amir-Mohammadian
Dept. of Computer Science
University of the Pacific
Stockton, CA, USA
samirmohammadian@pacific.edu

***Abstract*—Timing side-channel attacks exploit variations in execution time to infer confidential information from a system's behavior. In this paper, we develop a formal framework for analyzing timing side-channel leakage in cyber-physical systems by modeling their digital twins. We rely on a process-algebraic calculus for digital twins to express timing-sensitive behaviors and define its semantics using probabilistic Markov models. This enables rigorous quantification of information leakage based on execution timing variations. By leveraging entropy-based analysis, we measure an attacker's ability to infer secret states from observable execution delays.**

***Index Terms*—Cyber-physical systems, Cybersecurity, Digital twins, Timing attacks.**

I. Introduction

Cyber-Physical Systems (CPSs) integrate computational intelligence with physical processes, enabling real-time decision-making in domains such as autonomous systems, industrial automation, and smart grids [1]. These systems rely on embedded software to process sensor data and control actuators, ensuring adaptability and resilience in dynamic environments [2], [3]. Digital Twins (DTs) have further enhanced CPS capabilities by providing synchronized virtual models for system monitoring, predictive analytics, and operational optimization [4], [5].

Side-channel attacks are increasingly prevalent in CPSs, exploiting unintended information leakage from system behavior, such as execution timing, power consumption, electromagnetic emissions, or acoustic signals, e.g., [6]. While well studied in cryptography [7] and cloud computing [8], their impact on DT-driven CPSs remains underexplored. In particular, *timing side-channel attacks* leverage variations in execution time [9], sensor polling intervals [10], or communication latencies [11] to infer system states. For example, execution delays in a predictive maintenance CPS can reveal component degradation [12], while response time variations in an industrial CPS may expose operational loads or safety margins [13].

The continuous state space of CPS models complicates direct information leakage quantification. To address this, we focus on DTs as discrete approximations, enabling a more tractable analysis of information flow. To formalize information leakage, we consider a DT with a secret input h and a public output l. An attacker, with full knowledge of the DT's cyber component, executes the DT and observes time passage as well as the value of l. Leakage is quantified using the notion of *knowledge uncertainty*, which is the degree of ambiguity an observer has about a hidden variable. The amount of leaked information is the difference between knowledge uncertainties in the initial and final states [14]. Prior to execution, the attacker's uncertainty about h reflects her prior knowledge; observing outputs and timing during execution reduces this uncertainty, revealing information about h.

This paper develops a formal framework for analyzing timing side-channel leakage in DT-driven CPSs. Using a process-algebraic calculus of DTs, we model time-sensitive DT behaviors and demonstrate how execution timing variations serve as an attack vector. We further employ Markovian processes to model probabilistic execution paths and apply entropy-based quantification techniques to measure information leakage. Our approach provides a structured methodology for identifying and modeling timing side-channels in DT-driven CPSs through systematically i) handling CPS concurrency and nondeterminism, ii) modeling execution timing precisely, and iii) quantifying timing leakage using entropy measures.

For this work, we consider DTs as *simulations* of CPSs, emphasizing their analytical role over real-time synchronization. This assumption aligns with our study's scope, enabling the use of DTs as simplified models for analyzing and quantifying information leakage.

The rest of the paper is organized as follows. Section II introduces a process-algebraic model of DTs with labeled transition semantics. Section III presents the Markovian framework for leakage analysis. Section IV applies this framework to DT specifications and illustrates leakage quantification with an example. Section V reviews related work, and Section VI concludes the paper.

II. Process-Algebraic Model of Digital Twins

CPSs consist of a physical plant, a cyber controller, sensors, and actuators. Sensors collect data from the plant and send it to the cyber controller, which then adjusts actuators to influence the plant. Together, sensors, actuators, and the physical plant define the environment.

Inspired by CCPS [15], we define a calculus for DTs with cyber and physical components operating over floating-point values, where $\mathbb{R}_{\mathtt{s}}$ denotes the set of `s`-bit floats.

Let $x, h, l \in \mathcal{X}$ be state variables, $c \in \mathcal{C}$ be communication channels, $a \in \mathcal{A}$ be actuator names, and $s \in \mathcal{S}$ be sensor names. Values $v \in \mathcal{V}$ include floating-point numbers of size

$\mathbb{s}$. Given a name set $\mathcal{N}$, the function space $\mathbb{R}_{\mathbb{s}}^{\mathcal{N}} = \{\xi | \xi : \mathcal{N} \to \mathbb{R}_{\mathbb{s}}\}$ represents mappings from names to floating points. The function $\xi[n \mapsto v]$ updates n to v, keeping all other mappings unchanged. We define $\xi \leq \xi'$ if $\forall x \in \mathcal{N}, \xi(x) \leq \xi'(x)$.

A DT $E \bowtie P$ consists of a physical environment E and a cyber process P, formally defined in Definitions 2.1 and 2.2.

Definition 2.1 (Physical Environment: E): Let $\hat{\mathcal{X}} \subseteq \mathcal{X}$, $\hat{\mathcal{A}} \subseteq \mathcal{A}$, and $\hat{\mathcal{S}} \subseteq \mathcal{S}$. Then, the physical environment E is defined as $(\xi_{\mathrm{x}}, \xi_{\mathrm{u}}, \xi_{\mathrm{w}}, evol, \xi_{\mathrm{e}}, meas, inv)$, where: (i) $\xi_{\mathrm{x}} \in \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{X}}}$ is the state function, i.e., it maps state variables to floating point numbers. (ii) $\xi_{\mathrm{u}} \in \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{A}}}$ is the actuator function, i.e., it maps actuator names to floats. (iii) $\xi_{\mathrm{w}} \in \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{X}}}$ is the evolution uncertainty function, i.e., it maps a state variables to a floating point number denoting the maximum distance between its value and its representation in the model. (iv) $evol : \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{X}}} \times \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{A}}} \times \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{X}}} \to \mathcal{P}(\mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{X}}})$ is the evolution map that receives a state function, an actuator function, and the uncertainty function and returns a (finite) set of next possible states. (v) $\xi_{\mathrm{e}} \in \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{S}}}$ is the sensor error function (measurement error function), i.e., it maps each sensor name to a floating point number that denotes the maximum distance between its measured value and its actual value in the model. (vi) $meas : \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{X}}} \times \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{S}}} \to \mathcal{P}(\mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{S}}})$ is the measurement map, that receives a state function and an error function and returns a set of possible measurement functions. (vii) $inv : \mathbb{R}_{\mathbb{s}}^{\hat{\mathcal{X}}} \to \{\top, \bot\}$ is the invariable function. If $inv(\xi_{\mathrm{x}}) = \bot$, then the DT's E is in deadlock. If $inv(\xi_{\mathrm{x}}) = \top$ then the evolution of E can continue. We assume monotonic evolution with respect to uncertainty, i.e., for $\xi_{\mathrm{w}} \leq \xi'_{\mathrm{w}}$, it holds that $evol(\xi_{\mathrm{x}}, \xi_{\mathrm{u}}, \xi_{\mathrm{w}}) \subseteq evol(\xi_{\mathrm{x}}, \xi_{\mathrm{u}}, \xi'_{\mathrm{w}})$.

Definition 2.2 (Syntax of Processes: P): The syntax of processes is defined as $P, Q ::= \mathsf{nil} \mid \mathsf{idle}.P \mid P \parallel Q \mid \lfloor \pi.P \rfloor Q \mid [b]\{P\}, \{Q\} \mid P \backslash c \mid X \mid \mathsf{rec}\ X.P$.

nil is the terminal process. $\mathsf{idle}.P$ delays execution by one time unit before continuing as P. $P \parallel Q$ is the parallel execution of P and Q. $\lfloor \pi.P \rfloor Q$ is a prefix with timeout, where π represents: (i) $\mathsf{snd}\ c\langle v \rangle$, sending value v on channel c, (ii) $\mathsf{rcv}\ c(x)$, receiving a value on channel c and binding it to x, (iii) $\mathsf{read}\ s(x)$, reading from sensor s, (iv) $\mathsf{write}\ a\langle v \rangle$, writing v to actuator a. If π occurs within one time unit, execution proceeds as P; otherwise, it continues as Q. $P \backslash c$ restricts communication on channel c. $[b]\{P\}, \{Q\}$ executes P if boolean b is true; otherwise, executes Q. $\mathsf{rec}\ X.P$ defines recursion, ensuring X is time-guarded in P. For syntactic convenience, we define $\pi.P \triangleq \mathsf{rec} X. \lfloor \pi.P \rfloor X$, ensuring that π precedes P after potentially multiple time units.

To ensure proper integration, we impose a well-formedness condition: DT $E \bowtie P$ is well-formed if: (i) every sensor s in P has a corresponding $\xi_{\mathrm{e}}(s)$ in E, and (ii) every actuator a in P has a corresponding $\xi_{\mathrm{u}}(a)$ in E.

Semantics. Semantics is defined by the labeled transition system (LTS) of DTs. In this regard, we first introduce auxiliary functions on environments to simplify its expression. Let $E = (\xi_{\mathrm{x}}, \xi_{\mathrm{u}}, \xi_{\mathrm{w}}, evol, \xi_{\mathrm{e}}, meas, inv)$. We define: (i) $read\_sensor(E, s) = \{\xi(s) \mid \xi \in meas(\xi_{\mathrm{x}}, \xi_{\mathrm{e}})\}$: the set of possible sensor s measurements at state ξ_{x} with error ξ_{e}. (ii) $update\_act(E, a, v) = (\xi_{\mathrm{x}}, \xi_{\mathrm{u}}[a \mapsto v], \xi_{\mathrm{w}}, evol, \xi_{\mathrm{e}}, meas, inv)$: the environment E with actuator a updated to value v. (iii) $next(E) = \{(\xi, \xi_{\mathrm{u}}, \xi_{\mathrm{w}}, evol, \xi_{\mathrm{e}}, meas, inv) \mid \xi \in evol(\xi_{\mathrm{x}}, \xi_{\mathrm{u}}, \xi_{\mathrm{w}})\}$: the set of possible environments after state evolution. (iv) $inv(E) = inv(\xi_{\mathrm{x}})$: an abuse of notation lifting inv to environments.

We now define the LTS of processes and the entire system. The LTS for processes follows CCPS, with a slight modification: we introduce a set $\mathcal{M}$ containing all mappings from names to values (a memory). While $\mathcal{M}$ is not central to DT semantics, it becomes essential for studying information flow. The configurations consist of pairs $(P, \mathcal{M})$. Figure 1 defines the LTS of processes (rules P1-P14), where $\lambda \in \{\mathsf{idle}, \tau, \bar{c}v, cv, a!v, s?v\}$ represents transition labels. P6 propagates untimed actions over parallel components, while P14 enforces mandatory communication when enabled by P5.

The LTS of DTs is defined over configurations of the form $(E \bowtie P, \mathcal{M})$. Transitions are given by $E \bowtie P, \mathcal{M} \xrightarrow{\alpha} E' \bowtie P', \mathcal{M}'$ as shown in Figure 1 (rules D1-D6), where $\alpha \in \{\mathsf{idle}, \tau, cv, \bar{c}v\}$ represents DT activities: (i) τ: Non-observable activities, including sensor readings, actuator updates, and internal process communication. (ii) $cv, \bar{c}v$: Observable channel transmissions with external processes. (iii) idle: Time progression, evolving the environment. All DT transition rules assume $inv(E)$ as a precondition, but omitted for brevity.

The DT calculus defines three constructs that interact with time and may introduce timing side channels: (i) explicit delay execution, $\mathsf{idle}.P$, (ii) timeout on communication, $\lfloor \pi.P \rfloor Q$, and (iii) recursive process execution, $\mathsf{rec}\ X.P$.

The DT calculus preserves the semantics of CCPS, ensuring that properties like time wellness and bisimulation remain consistent. However, as these aspects are not central to information leakage quantification in this work, we do not cover them here.

III. Basics of Information Flow Quantification

This section introduces key concepts and notations for the proposed Quantitative Information Flow (QIF) approach. While Salehi et al. [16] provide a detailed treatment, we include relevant material for a self-contained exposition.

A probability distribution is a function $Pr : \mathbb{X} \to [0, 1]$ over a set $\mathbb{X}$, satisfying $\sum_{x \in \mathbb{X}} Pr(x) = 1$. The set of all such distributions is denoted as $\mathcal{D}(\mathbb{X}) = \{Pr \mid Pr : \mathbb{X} \to [0, 1]\}$. Given an equivalence relation R on a set S, the quotient space S/R is defined as usual. For a random variable X over a finite set $\mathbb{X}$, its vulnerability is $Vul(X) = \max_{x \in \mathbb{X}} Pr(X = x)$, which is the maximum chance of accurately guessing the value of a variable in a single try. Renyi's min-entropy is then: $H_\infty(X) = -\log_2 Vul(X)$.

Markov Decision Processes (MDPs) offer a formal foundation for analyzing information flow in DTs. However, their inherent nondeterminism limits their suitability for QIF analysis. To address this, our model resolves nondeterminism by employing probabilistic schedulers, which induce Markov Chains (MCs)—structures that serve as an appropriate formal basis for studying QIF in DTs.

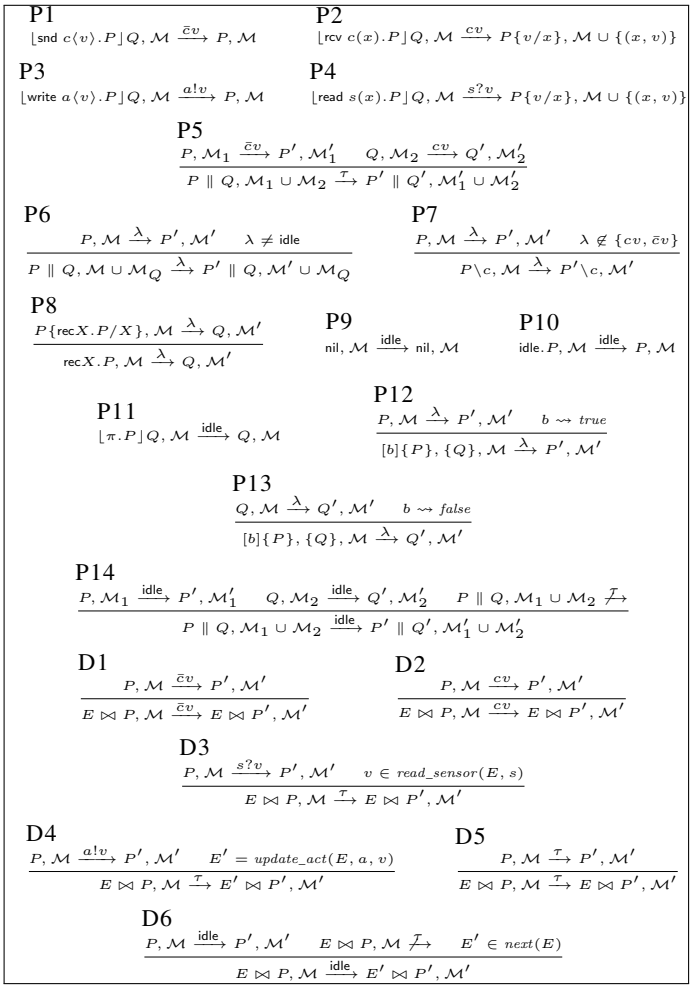

Fig. 1: LTS of 1) processes: $P, \mathcal{M} \xrightarrow{\lambda} P', \mathcal{M}'$, and 2) DTs: $E \bowtie P, \mathcal{M} \xrightarrow{\alpha} E' \bowtie P', \mathcal{M}'$.

Definition 3.1 (Markov Decision Process (MDP)): $M = (S, Act, \mathbf{P}, \zeta, AP, V)$ is an MDP where: (i) S is the set of states. (ii) Act is the set of actions. (iii) $\mathbf{P} : S \to Act \to S \to [0,1]$ is the transition probability such that $\forall s \in S, \alpha \in Act, \Sigma_{s' \in S} \mathbf{P}(s)(\alpha)(s') \in \{0,1\}$ i.e., action α is either enabled in a state or not. (iv) $\zeta : S \to [0,1]$ is the initial state probability distribution, with $\Sigma_{s \in S} \zeta(s) = 1$. (v) AP is the set of atomic propositions. (vi) $V : S \to AP$ is the labeling function.

The set of enabled actions at state s is $Act(s)$, while $Init(M)$ denotes the set of initial states. An execution path σ is a sequence of states connected by transitions, with $PathFrags(s, s')$ representing fragments between states s and s'. A trace is a sequence of labels. Multiple paths can yield the same trace.

Definition 3.2 (Markov Chain (MC)): $M = (S, \mathbf{P}, \zeta, AP, V)$ is a discrete time MC, where $\mathbf{P} : S \times S \to [0,1]$ is the transition probability between states, such that $\forall s \in S, \Sigma_{s' \in S} \mathbf{P}(s, s') = 1$. Other components of MC are defined identical to Definition 3.1.

The probability of an execution path $\sigma = (s_0, \ldots, s_n)$ is (i) $\zeta(s_0)$, if $n = 0$, and (ii) $\zeta(s_0) \cdot \Pi_{0 \le i \le n-1} Pr(s_i, s_{i+1})$, if $n \neq 0$. Then, reachability probability of state s is $Pr(s) = \Sigma_{\sigma \in PathFrags(s_0, s) \wedge s_0 \in Init(M)} Pr(\sigma)$.

For QIF, we truncate execution paths at a fixed length to ensure bounded analysis. The resulting states at this length are called final states. The truncated Markovian models of DTs form directed acyclic graphs (DAGs), where initial states act as roots and final states as leaves.

A probabilistic scheduler resolves MDP nondeterminism by assigning execution probabilities to subprocesses, determining their order. Since DT concurrency is nondeterministic, the scheduler controls execution flow. We use memoryless probabilistic schedulers, which assign action probabilities based only on the current state, inducing an MC from the MDP.

Definition 3.3 (Memoryless probabilistic scheduler (MPS)): Let $M = (S, Act, \mathbf{P}, \zeta, AP, V)$ be an MDP. $\delta : S \to \mathcal{D}(Act)$ is an MPS such that $\delta(s) \in \mathcal{D}(Act(s))$ for all $s \in S$.

Definition 3.4 (Induced MC of an MDP): Let $M = (S, Act, \mathbf{P}, \zeta, AP, V)$ be an MDP and $\delta : S \to \mathcal{D}(Act)$ be an MPS. The MC of M induced by δ is $M_\delta = (S, \mathbf{P}_\delta, \zeta, AP, V)$, where $\mathbf{P}_\delta(s, s') = \Sigma_{\alpha \in Act(s)} \delta(s)(\alpha) \cdot \mathbf{P}(s)(\alpha)(s')$.

IV. Computing Information Leakage in DTs

In this section, we define the semantics of DTs using Markovian models, specifying their MDP representation. Let $M^{E \bowtie P} = (S, Act, \mathbf{P}, \zeta, AP, V)$, where: (i) S is the set of LTS configurations of DTs, where each state $s \in S$ takes the form $(E \bowtie P, \mathcal{M})$. We assume predefined variables $l, h \in \mathcal{X}$ stored in $\mathcal{M}$, representing public and secret data from the attacker's perspective. When multiple public or secret components exist, l and h are treated as concatenated records. Let Val_l and Val_h denote their respective domains. (ii) $Act = \{\varepsilon, \tau, \mathsf{idle}, cv, \bar{c}v\}$ is the set of actions, derived from LTS transitions. Since different evaluation contexts can yield the same activity, actions may transition nondeterministically to multiple states. (iii) $\mathbf{P}$ defines state transition probabilities under specific actions. Evolution uncertainty ξ_{w} and sensor errors ξ_{e} introduce probabilistic transitions, where sensor errors follow probability distributions [17]. Evolution uncertainty models probable system behaviors, distinct from nondeterminism, which governs subprocess selection. (iv) The initial state probability is given by $\zeta(s) = Pr(h = \bar{h})$, where $s = (E \bowtie P, \mathcal{M})$, $E = (\xi_{\mathrm{x}}, \cdots)$, and $\xi_{\mathrm{x}}(h) = \bar{h}$. (v) AP is instantiated as $Val_l \times \mathbb{N}$, where $\mathbb{N}$ represents timestamps as natural numbers counting time units. (vi) The labeling function $V(E \bowtie P, \mathcal{M}) \in Val_l \times \mathbb{N}$ maps a state to a public value l and its associated timestamp in the execution path.

According to Definition 3.4, MPS δ resolves nondeterminism in MDP $^{E \bowtie P}$ and induces MC $M_\delta^{E \bowtie P}$.

Example 4.1: Consider a DT with floating-point precision fixed to one digit. The initial state $s_0 = E_0 \bowtie P_0, \mathcal{M}_0$ consists of: (i) $E_0 = (\xi_{\mathrm{x}}, \xi_{\mathrm{u}}, \xi_{\mathrm{w}}, evol, \xi_{\mathrm{e}}, meas, inv)$, where h and x are environmental variables with $\xi_{\mathrm{x}}(x) = 0.0$ and evolution uncertainty $\xi_{\mathrm{w}}(x) = 0.1$. The system has a single sensor s with $\xi_{\mathrm{e}}(s) = 0.5$ and a single actuator a with

$\xi_{\mathrm{u}}(a) = 0.2$. Evolution is defined as $evol(\xi_{\mathrm{x}}, \xi_{\mathrm{u}}, \xi_{\mathrm{w}}) = \{\xi \mid \xi(x) = \xi_{\mathrm{x}}(x) + \xi_{\mathrm{u}}(a) + \gamma, \gamma \in \{-0.1, 0.0, 0.1\}\}$. Sensor measurement follows $meas(\xi_{\mathrm{x}}, \xi_{\mathrm{e}}) = \{\xi \mid \xi(s) = \xi_{\mathrm{x}}(h) + \gamma, \gamma \in \{-0.5, -0.4, \ldots, 0.5\}\}$. The system remains valid as long as $inv(\xi_{\mathrm{x}}) = (\xi_{\mathrm{x}}(x) < 1.0)$. (ii) $P_0 = \mathsf{read}\ s(h).([h > 0.4]\{\mathsf{idle.idle}.P\}\{\mathsf{idle}.P\})\backslash c$, where $P = \mathsf{snd}\ c\langle 1\rangle.\mathsf{nil} \parallel \mathsf{rcv}\ c(l).\mathsf{nil}$. (iii) $\mathcal{M}_0 = \{(l, 0.0)\}$.

The sequence of states in a path typically follows this pattern: First, the sensor reads h, potentially introducing a measurement error. Based on the measured value, the process takes either 1 or 2 time units to assign 1 to l. For simplicity, both evolution uncertainty and sensor error are assumed to follow a uniform distribution. We fix h at either 0.0 or 1.0, focusing solely on confidentiality. These two cases are analyzed below.

Case 1: $\xi_{\mathrm{x}}(h) = 0.0$**.** ***Step 1:*** From s_0 eleven states $s_{1,1}, s_{1,2}, \ldots, s_{1,11}$ are accessible, each with action τ and probability of transition $1/11$ such that for $i \in \{1, 2, \ldots, 11\}$, $s_{1,i} = (E_0 \bowtie P_1, \mathcal{M}_{1,i})$, where P_1 is defined as $([h > 0.4]\{\mathsf{idle.idle}.P\}\{\mathsf{idle}.P\})\backslash c$ and $\mathcal{M}_{1,1} = \{(l, 0.0), (h, -0.5)\}, \mathcal{M}_{1,2} = \{(l, 0.0), (h, -0.4)\}, \ldots, \mathcal{M}_{1,11} = \{(l, 0.0), (h, 0.5)\}$. ***Step 2:*** From each $s_{1,i}$ ($i \in \{1, 2, \ldots, 11\}$), three states $s_{2,i,j}$ ($j \in \{1, 2, 3\}$) are accessible with action idle and probability 1/3, such that $s_{2,i,j} = (E_{2,i,j} \bowtie P_{2,i,j}, \mathcal{M}_{2,i,j})$, where $\mathcal{M}_{2,i,j} = \mathcal{M}_{1,i}$, and (i) in $E_{2,i,1}$ we have $\xi_{\mathrm{x}}(x) = 0.1$, (ii) in $E_{2,i,2}$ we have $\xi_{\mathrm{x}}(x) = 0.2$, and (iii) in $E_{2,i,3}$ we have $\xi_{\mathrm{x}}(x) = 0.3$. While $P_{2,11,j} = \mathsf{idle}.P\backslash c$ ($j \in \{1, 2, 3\}$), we have $P_{2,i,j} = P\backslash c$, for $i \in \{1, \ldots 10\}$ and $j \in \{1, 2, 3\}$. ***Step 3:*** (1) Each state $s_{2,i,j}$ ($i \in \{1, \ldots, 10\}$ and $j \in \{1, 2, 3\}$) transitions to state $s_{3,j,i} = (E_{3,i,j} \bowtie P_{3,i,j}, \mathcal{M}_{3,i,j})$ with action τ and probability 1, where $P_{3,i,j} = \mathsf{nil} \parallel nil$, $E_{3,i,j} = E_{2,i,j}$, and $\mathcal{M}_{3,1,j} = \{(l, 1.0), (h, -0.5)\}, \mathcal{M}_{3,2,j} = \{(l, 1.0), (h, -0.4)\}, \ldots, \mathcal{M}_{3,10,j} = \{(l, 0.0), (h, 0.4)\}$. (2) Each state $s_{2,11,j}$ ($j \in \{1, 2, 3\}$) transitions to three states $s_{3,11,j}$, $s_{3,11,j+1}$ and $s_{3,11,j+2}$ with action idle and probability 1/3, where for $k \in \{1, \ldots, 5\}$, $P_{3,11,k} = P\backslash c$, $\mathcal{M}_{3,11,k} = \mathcal{M}_{1,11} = \{(l, 0.0), (h, 0.5)\}$, and (i) in $E_{3,11,1}$ we have $\xi_{\mathrm{x}}(x) = 0.2$, (ii) in $E_{3,11,2}$ we have $\xi_{\mathrm{x}}(x) = 0.3$, (iii) in $E_{3,11,3}$ we have $\xi_{\mathrm{x}}(x) = 0.4$, (iv) in $E_{3,11,4}$ we have $\xi_{\mathrm{x}}(x) = 0.5$, and (v) in $E_{3,11,5}$ we have $\xi_{\mathrm{x}}(x) = 0.6$. ***Step 4:*** (1) Each state $s_{3,i,j}$ ($i \in \{1, \ldots, 10\}$ and $j \in \{1, 2, 3\}$) transitions to three states $s_{4,i,j}$, $s_{4,i,j+1}$ and $s_{4,i,j+2}$ with action idle and probability 1/3, where for $k \in \{1, \ldots, 5\}$, $P_{4,i,k}$ and $\mathcal{M}_{4,i,k}$ are the same as the process and memory of the predecessor state, and i) in $E_{4,i,1}$ we have $\xi_{\mathrm{x}}(x) = 0.2$, (ii) in $E_{4,i,2}$ we have $\xi_{\mathrm{x}}(x) = 0.3$, (iii) in $E_{4,i,3}$ we have $\xi_{\mathrm{x}}(x) = 0.4$, (iv) in $E_{4,i,4}$ we have $\xi_{\mathrm{x}}(x) = 0.5$, and (v) in $E_{4,i,5}$ we have $\xi_{\mathrm{x}}(x) = 0.6$. (2) Each state $s_{3,11,k}$ ($k \in \{1, \ldots, 5\}$) transitions to state $s_{4,11,k}$ with action τ and probability 1, where $P_{4,11,k} = \mathsf{nil} \parallel nil$, $E_{4,11,k} = E_{3,11,k}$, and $\mathcal{M}_{4,11,k} = \{(l, 1.0), (h, 0.5)\}$.

The first DAG in Figure 2 illustrates the resulting MDP. Note that to reduce visual clutter, the transition label idle is replaced with i where necessary in this figure. The DAG contains 135 states and 99 distinct paths from initial to final states. States s_0 and $s_{1,i}$ ($i \in \{1, \ldots, 11\}$) have label $(l = 0, 0)$, whereas states $s_{2,i,j}$ for $i \in \{1, \ldots, 11\}$ and $j \in \{1, 2, 3\}$ have label $(l = 0, 1)$, states $s_{3,i,j}$ for $i \in \{1, \ldots, 10\}$ and $j \in \{1, 2, 3\}$ have label $(l = 1, 1)$, and states $s_{3,11,k}$ for $k \in \{1, \ldots, 5\}$ have label $(l = 0, 2)$. States $s_{4,i,k}$ for $i \in \{1, \ldots, 11\}$ and $k \in \{1, \ldots, 5\}$ have label $(l = 1, 2)$.

Case 2: $\xi_{\mathrm{x}}(h) = 1.0$**.** In this case, the DAG structure is the same, given as the second DAG in Figure 2, with the following differences in state details: For for $i \in \{1, \ldots, 11\}$, $j \in \{1, 2, 3\}$, and $k \in \{1, \ldots, 5\}$: (i) $\mathcal{M}'_{1,1} = \{(l, 0.0), (h, 0.5)\}, \mathcal{M}'_{1,2} = \{(l, 0.0), (h, 0.6)\}, \ldots, \mathcal{M}'_{1,11} = \{(l, 0.0), (h, 1.5)\}$. (ii) $P'_{2,i,j} = \mathsf{idle}.P\backslash c$. (iii) Each state $s'_{2,i,j}$ transitions to three states $s'_{3,i,j}$, $s'_{3,i,j+1}$ and $s'_{3,i,j+2}$ with action idle and probability 1/3, where $P_{3,i,k} = P\backslash c$, $\mathcal{M}_{3,i,k} = \mathcal{M}_{1,i}$, and (i) in $E'_{3,i,1}$ we have $\xi_{\mathrm{x}}(x) = 0.2$, (ii) in $E'_{3,i,2}$ we have $\xi_{\mathrm{x}}(x) = 0.3$, (iii) in $E'_{3,i,3}$ we have $\xi_{\mathrm{x}}(x) = 0.4$, (iv) in $E'_{3,i,4}$ we have $\xi_{\mathrm{x}}(x) = 0.5$, and (v) in $E'_{3,i,5}$ we have $\xi_{\mathrm{x}}(x) = 0.6$. (iv) Each state $s'_{3,i,k}$ transitions to state $s'_{4,i,k}$ with action τ and probability 1, where $P'_{4,i,k} = \mathsf{nil} \parallel nil$, $E'_{4,i,k} = E'_{3,i,k}$, and $\mathcal{M}'_{4,1,k} = \{(l, 1.0), (h, 0.5)\}, \mathcal{M}'_{4,2,k} = \{(l, 1.0), (h, 0.6)\}, \ldots, \mathcal{M}'_{4,11,k} = \{(l, 0.0), (h, 1.5)\}$. (v) States s'_0 and $s'_{1,i}$ have label $(l = 0, 0)$, states $s'_{2,i,j}$ have label $(l = 0, 1)$, states $s'_{3,i,k}$ have label $(l = 0, 2)$, and states $s'_{4,i,k}$ have label $(l = 1, 2)$.

In this example, only one action is available in each state to transition to the next state, meaning the scheduler does not affect the structure of MC DAGs. Thus, $M_\delta^{E \bowtie P}$ is structurally identical to $M^{E \bowtie P}$, differing only in that each transition in $M_\delta^{E \bowtie P}$ is associated solely by its transition probability.

A. Attacker's View of the DT

To assess the attacker's ability to infer h, we define her view of the DT. The attacker can distinguish final from non-final states based on time progression but cannot differentiate between execution paths yielding identical traces. For instance, in the induced MC of Example 4.1, the attacker perceives only two traces $\langle (l = 0, 0), (l = 0, 0), (l = 0, 1), (l = 1, 1), (l = 1, 2)\rangle$ and $\langle (l = 0, 0), (l = 0, 0), (l = 0, 1), (l = 0, 2), (l = 1, 2)\rangle$, both spanning two time units, despite the presence of 198 distinct execution paths. As a result, final states sharing the same public values, timestamps, and trace remain indistinguishable. Although the attacker does not directly observe h, she can infer its possible values probabilistically in each final state. This reasoning is formalized through the *back-bisimulation* equivalence relation [16], denoted by $\sim_b$.

Definition 4.2: Let $M_\delta^{E \bowtie P}$ be an MC. A back-bisimulation for $M_\delta^{E \bowtie P}$ is a binary relation R on S such that for all $(s_1, s_2) \in R$: (i) $V(s_1) = V(s_2)$, (ii) every predecessor s'_1 of s_1 has a corresponding predecessor s'_2 of s_2 with $(s'_1, s'_2) \in R$, and (iii) every predecessor s'_2 of s_2 has a corresponding predecessor s'_1 of s_1 with $(s'_1, s'_2) \in R$. States s_1 and s_2 are back-bisimilar, denoted $s_1 \sim_b s_2$, if a back-bisimulation R exists with $(s_1, s_2) \in R$.

In Definition 4.2, condition (i) ensures that states s_1 and s_2 share the same public values. Condition (ii) requires that every incoming transition to s_1 has a corresponding transition

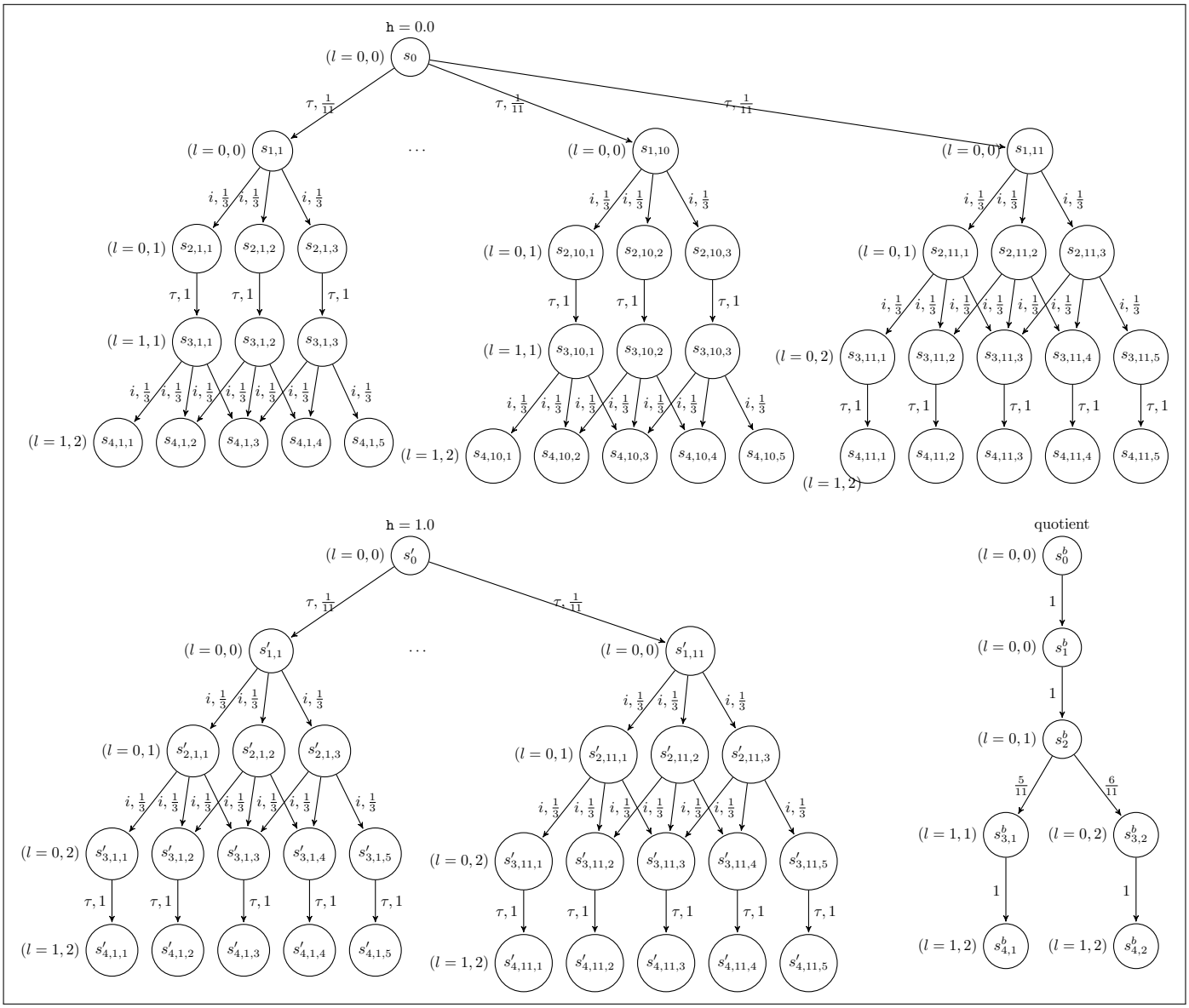

Fig. 2: Markovian model of example DT: 1) MDP in case $h = 0.0$, 2) MDP in case $h = 1.0$, and 3) back-bisimulation quotient (attacker's view).

to s_2, while condition (iii) ensures the reverse. Consequently, $\sim_b$ forms an equivalence relation, partitioning the MC state space into equivalence classes. The back-bisimulation quotient of $M_\delta^{E \bowtie P}$, denoted $M_\delta^{E \bowtie P} / \sim_b$, is defined accordingly.

Definition 4.3: For MC $M_\delta^{E \bowtie P} = (S, \mathbf{P}_\delta, \zeta, Val_l, V)$ and relation $\sim_b$, the back-bisimulation quotient is defined as $M_\delta^{E \bowtie P} / \sim_b = (S/\sim_b, \mathbf{P}'_\delta, s_0^b, Val_l, V, Pr(h))$ where: (i) $S/\sim_b$ is the quotient space of S under $\sim_b$, (ii) $\mathbf{P}'_\delta : (S/\sim_b) \times (S/\sim_b) \to [0,1]$ such that for all s^b and t^b, $\mathbf{P}'_\delta(s^b, t^b) = (\Sigma_{s \in s^b, t \in t^b} Pr(s) \cdot \mathbf{P}_\delta(s,t)) / Pr(s^b)$, where $Pr(s)$ and $Pr(s^b)$ are the probabilities of reaching to s and s^b in $M_\delta^{E \bowtie P}$ and $M_\delta^{E \bowtie P} / \sim_b$ respectively, (iii) $s_0^b = Init(M)$, (iv) $V([s]_{\sim_b}) = V(s)$, and (v) $Pr(h)$ maps s^b to the probability distribution of h in s^b, $Pr(h_{s^b})$, such that $Pr(h_{s^b} = \bar{h}) = (\Sigma_{s \in s^b, s=(E \bowtie P, \mathcal{M}), \mathcal{M}=(\xi_{\mathrm{x}}, \cdots), \xi_{\mathrm{x}}(h)=\bar{h}} Pr(s)) / Pr(s^b)$.

Example 4.4: Having defined the induced MC $M_\delta^{E \bowtie P}$ of the DT in Example 4.1, the back-bisimulation quotient $M_\delta^{E \bowtie P} / \sim_b$ is given as the third DAG in Figure 2. The label in s_0^b and s_1^b is $(l = 0, 0)$, in s_2^b the label is $(l = 0, 1)$, in $s_{3,1}^b$ the label is $(l = 1, 1)$, in $s_{3,2}^b$ the label is $(l = 0, 2)$, and in states $s_{4,1}^b$ and $s_{4,2}^b$ the label is $(l = 1, 2)$. The probability of transition from s_2^b to $s_{3,1}^b$ is $5/11$, and to $s_{3,2}^b$ is $6/11$. Each state is an equivalence class, containing back-bisimilar states of $M_\delta^{E \bowtie P}$, given in the following: (i) $s_0^b = \{s_0, s'_0\}$, (ii) $s_1^b = \{s_{1,i}, s'_{1,i} \mid i \in \{1, \ldots, 11\}\}$, (iii) $s_2^b = \{s_{2,i,j}, s'_{2,i,j} \mid i \in \{1, \ldots 11\}, j \in \{1,2,3\}\}$, (iv) $s_{3,1}^b = \{s_{3,i,j} \mid i \in \{1, \ldots 10\}, j \in \{1,2,3\}\}$, (v) $s_{3,2}^b = \{s'_{3,i,k} \mid i \in \{1, \ldots 11\}, k \in \{1, \ldots, 5\}\} \cup \{s_{3,11,k} \mid k \in \{1, \ldots, 5\}\}$,

(vi) $s^b_{4,1} = \{s_{4,i,k} \mid i \in \{1,\dots 10\}, k \in \{1,\dots,5\}\}$, and $s^b_{4,2} = \{s'_{4,i,k} \mid i \in \{1,\dots 11\}, k \in \{1,\dots,5\}\} \cup \{s_{4,11,k} \mid k \in \{1,\dots,5\}\}$.

$Pr(h)$ for each state is defined as follows: (i) $Pr(h_{s^b_0}) = Pr(h_{s^b_1}) = Pr(h_{s^b_2}) = \{0 \mapsto 1/2, 1 \mapsto 1/2\}$, (ii) $Pr(h_{s^b_{3,1}}) = Pr(h_{s^b_{4,1}}) = \{0 \mapsto 1, 1 \mapsto 0\}$, and (iii) $Pr(h_{s^b_{3,2}}) = Pr(h_{s^b_{4,2}}) = \{0 \mapsto 1/12, 1 \mapsto 11/12\}$.

B. Information Leakage Quantification

In each state of $M^{E \bowtie P}_\delta / \sim_b$, the attacker's uncertainty about h is captured by $Pr(h)$. As execution progresses, the scheduler's choice of (sub)processes and the attacker's observations of public values and timestamps cause $Pr(h)$ to evolve along each trace, transforming the initial distribution of h into posterior distributions in the final states, denoted by $final(M^{E \bowtie P}_\delta / \sim_b)$. For example, in Example 4.1, this length is set to 2 time units.

h's uncertainty in state s^b, given $Pr(h_{s^b})$, is measured by $H_\infty(h_{s^b})$. The initial uncertainty is $H_\infty(h_{s^b_0})$, while the remaining uncertainty in final states is given by the expected uncertainty $\sum_{s^b_f \in final(M^{E \bowtie P}_\delta / \sim_b)} Pr(s^b_f) \cdot H_\infty(h_{s^b_f})$. The leakage of the DT $E \bowtie P$ under scheduler δ is then $\mathcal{L}(E \bowtie P) = H_\infty(h_{s^b_0}) - \sum_{s^b_f \in final(M^{E \bowtie P}_\delta / \sim_b)} Pr(s^b_f) \cdot H_\infty(h_{s^b_f})$.

Example 4.5: In the DT of Example 4.1 with the backbisimulation quotient in Figure 2, the leaked information amount is computed as 0.93 bits:

$$\begin{aligned}&\mathcal{L}(E \bowtie P) = \\ &\quad H_\infty(h_{s^b_0}) - (Pr(s^b_{4,1}) \cdot H_\infty(h_{s^b_{4,1}}) + Pr(s^b_{4,2}) \cdot H_\infty(h_{s^b_{4,2}})) \\ &\quad = 1 - (5/11 \cdot 0 + 6/11 \cdot (-\log_2(11/12))) = 0.93.\end{aligned}$$

V. Related Work

Several works have explored information leakage in cyber-physical systems (CPSs) and digital twins (DTs) through formal and statistical methods. Feng et al. [18] analyzed smart grid CPSs from an information flow perspective, showing how physical observations can unintentionally reveal cyber information, and proposed a method to quantify such leakage. Building on this, Feng et al. [19] developed a framework combining algorithmic quantification and statistical analysis to measure information leakage in CPSs, with a focus on smart grid contexts. Zhang et al. [20] investigated side-channel-based model extraction attacks on intelligent CPSs, presenting a framework that exploits side-channel data to reconstruct internal models. More recently, Huang et al. [21] used a temporal logic variant to formally model DTs and provide statistical guarantees on information leakage during interactions between physical systems and their digital counterparts.

VI. Conclusion

This paper introduced a formal framework for analyzing and quantifying timing side-channel leakage in CPSs through their DTs. Using the algebraic calculus of DTs, we model timing-sensitive behaviors and demonstrated how execution timing variations serve as an attack vector. By leveraging probabilistic Markov models and entropy-based quantification, we systematically evaluated information leakage and attacker inference capabilities. Our findings highlight the critical role of execution timing in revealing sensitive system states, emphasizing the need for robust mitigation strategies.

Future work includes extending our framework to capture other side-channel vectors, refining mitigation techniques, and applying our analysis to real-world DT implementations.

References

[1] J. Shi, J. Wan, H. Yan, and H. Suo, "A survey of cyber-physical systems," in *WCSP*. IEEE, 2011, pp. 1–6.

[2] P. Sharma and J. Gillanders, "Cybersecurity and forensics in connected autonomous vehicles: A review of the state-of-the-art," *IEEE Access*, vol. 10, pp. 108 979–108 996, 2022.

[3] K. Zhang, Y. Shi, S. Karnouskos, T. Sauter, H. Fang, and A. W. Colombo, "Advancements in industrial cyber-physical systems: An overview and perspectives," *IEEE T IND INFORM*, vol. 19, no. 1, pp. 716–729, 2022.

[4] L. M. Oliveira, R. Dias, C. M. Rebello, M. A. Martins, A. E. Rodrigues, A. M. Ribeiro, and I. B. Nogueira, "Artificial intelligence and cyber-physical systems: A review and perspectives for the future in the chemical industry," *AI*, vol. 2, no. 3, p. 27, 2021.

[5] S. Zeb, A. Mahmood, S. A. Hassan, M. J. Piran, M. Gidlund, and M. Guizani, "Industrial digital twins at the nexus of nextg wireless networks and computational intelligence: A survey," *J. Netw. Comput. Appl.*, vol. 200, p. 103309, 2022.

[6] S. R. Chhetri, A. Canedo, and M. A. A. Faruque, "Confidentiality breach through acoustic side-channel in cyber-physical additive manufacturing systems," *ACM TCPS*, vol. 2, no. 1, pp. 1–25, 2017.

[7] P. C. Kocher, "Timing attacks on implementations of diffie-hellman, rsa, dss, and other systems," in *CRYPTO'96*. Springer, 1996, pp. 104–113.

[8] S. Zander, G. Armitage, and P. Branch, "Covert channels and countermeasures in computer network protocols," *IEEE Commun. Mag.*, vol. 45, no. 12, pp. 136–142, 2007.

[9] Y. Jin, P. Qiu, C. Wang, Y. Yang, D. Wang, and G. Qu, "Timing the transient execution: A new side-channel attack on intel cpus," *arXiv preprint arXiv:2304.10877*, 2023.

[10] C. Shepherd, J. Kalbantner, B. Semal, and K. Markantonakis, "A side-channel analysis of sensor multiplexing for covert channels and application fingerprinting on mobile devices," *arXiv preprint arXiv:2110.06363*, 2021.

[11] M. Tan, J. Wan, Z. Zhou, and Z. Li, "Invisible probe: Timing attacks with pcie congestion side-channel," in *IEEE S&P*. IEEE, 2021, pp. 322–338.

[12] WorkTrek, "The ultimate guide to predictive maintenance," https://rb.gy/6366t2, 2025, accessed on 2025-05-08.

[13] S. Mubeen, E. Lisova, and A. Vulgarakis Feljan, "Timing predictability and security in safety-critical industrial cyber-physical systems: A position paper," *Applied Sciences*, vol. 10, no. 9, p. 3125, 2020.

[14] G. Smith, "On the foundations of quantitative information flow," in *FoSSaCS*. Springer, 2009, pp. 288–302.

[15] R. Lanotte and M. Merro, "A calculus of cyber-physical systems," in *LATA*. Springer, 2017, pp. 115–127.

[16] K. Salehi, A. A. Noroozi, S. Amir-Mohammadian, and M. Mohagheghi, "An automated quantitative information flow analysis for concurrent programs," in *QEST*. Springer, 2022, pp. 43–63.

[17] P. Harris, P. F. Østergaard, S. Tabandeh, H. Söderblom, G. Kok, M. van Dijk, Y. Luo, J. Pearce, D. Tucker, A. P. Vedurmudi *et al.*, "Measurement uncertainty evaluation for sensor network metrology," *Metrology*, vol. 5, no. 1, p. 3, 2025.

[18] L. Feng and B. McMillin, "Quantification of information flow in a smart grid," in *IEEE COMPSACW*. IEEE, 2014, pp. 140–145.

[19] ——, "Information flow quantification framework for cyber physical system with constrained resources," in *IEEE COMPSAC*, vol. 2. IEEE, 2015, pp. 50–59.

[20] Q. Pan, J. Wu, X. Lin, and J. Li, "Side-channel analysis-based model extraction on intelligent cps: An information theory perspective," in *IEEE iThings and GreenCom and CPSCom and SmartData and Cybermatics*. IEEE, 2021, pp. 254–261.

[21] L. Huang, L. R. Varshney, and K. E. Willcox, "Formal verification of digital twins with TLA and information leakage control," *arXiv preprint arXiv:2411.18798*, 2024.