Formal Robustness for Cyber-Physical Systems under Timed Attacks

Jian Xiang
UNC Charlotte
Charlotte, NC 28223, US
jian.xiang@charlotte.edu

Simone Tini
University of Insubria
Como, 22100, Italy
simone.tini@uninsubria.it

Ruggero Lanotte
University of Insubria
Como, 22100, Italy
ruggero.lanotte@uninsubria.it

Massimo Merro
University of Verona
Verona, 37134, Italy
massimo.merro@univr.it

***Abstract*—Cyber-physical systems are increasingly deployed in safety-critical applications, making their robustness under adversarial conditions a critical concern. Among the diverse range of threats, timed attacks, i.e., attacks triggered at particular timing, pose a unique challenge due to their ability to disrupt system behaviors in subtle and complex ways. In this paper, we propose a formal framework for quantitative analysis of the robustness of system's safety against timed attacks on cyber-physical systems modeled via the formalism of *hybrid programs* and *differential dynamic logic*. We introduce a series of timing related properties to characterize the robustness of safety against timed attacks, and develop a system of reasoning techniques, with a focus on the timing of dynamics, to establish these properties. We showcase the reasoning techniques with a case study on a water tank system with non-trivial dynamics.**

I. Introduction

Cyber-Physical Systems (CPSs) are complex systems that integrate controllers and physical dynamics, playing a crucial role in many safety-critical applications, such as autonomous vehicles, power grids, and industrial automation.

The hybrid nature of CPSs exposes them to a wide range of *cyber-physical attacks* [1], [2], i.e., security breaches in cyberspace that adversely affect the physical process of the CPS under attack. In particular, *timed attacks* [3] leverage precise timing to induce subtle but significant disruptions, such as destabilizing control loops, misaligning schedules, or triggering unsafe behaviors. For example, timed GPS spoofing can mislead autonomous vehicles at highway merge, and coordinated load manipulation can destabilize power grids. The challenge is further amplified by the difficulty of detecting timed attacks, as their subtle and unpredictable nature allows them to mimic legitimate timing behaviors, often evading standard intrusion detection systems (IDSs).

While great progress has been made in verifying CPS safety properties using formal methods [4], [5], [6], existing approaches often focus on binary safety guarantees [7]. However, we believe that *quantitative safety* [8], [6], [9] is much more interesting in the setting of CPSs because it allows system engineers to evaluate the safety margins under timed attacks, such as how much the duration of a timed attack can affect a system's safety, how long the system can sustain the impact of timed attack without failing, and how much the system can tolerate unsafety if it eventually recovers to safety.

In this paper, we investigate the impact on the safety of CPSs when exposed to *timed physics-based attacks* tampering with sensor measurements and/or actuator commands. We introduce a formal framework that integrates quantitative safety notions with the ability to reason about the impact a timed attack has on a system's safety. Our threat model assumes that the timed attacks target the controller, and system engineers have detected and classified the timed attacks. Upon detection they try to estimate robustness, impact, and tolerance in the compromised system. We first introduce a notion of quantitative safety that incorporates timing, based on the minimum distance to the unsafe region from a system's reachable states within a time duration. Using this notion, we define three *timed properties*: (1) *timed robustness*, which quantifies how much a system's safety would be impacted by a timed attack, if it stays safe; (2) *timed impact*, which quantifies how unsafe the system would become; and (3) *timed tolerance*, which quantifies how much the system can tolerate unsafety without violating a system threshold.

To establish these timed properties, our approach reduces their analyses to reasoning about the relationship between the compromised system and the genuine system. Such a relational reasoning offers a significant advantage in analyzing the effectiveness of timed attacks: it avoids the need for a complete safety analysis of the compromised system, which may require an exhaustive exploration of its behaviors. Instead, relational reasoning allows us to focus on (1) the differences caused by the attacks and (2) the duration when the attacks make impact. Such an approach can enable more efficient analysis, which is particularly useful considering that CPSs often have large state space.

Inspired by the notion of behavioral distance [10], we formalize such a relationship as *timed simulation distance*: a *forall-exists* relational property that intuitively quantifies the difference in the behavior of the compromised system and the genuine system in terms of safety, in a specific time interval. In particular, our timed distance provides lower bounds on the safety of the compromised system w.r.t. the genuine one.

In the paper, we work within the formalism of *hybrid programs* and *differential dynamic logic* (dL) [11], [12], [13]. Hybrid programs are a formalism for modeling systems that have both continuous and discrete behaviors. They can express continuous evolution (as differential equations) as well as

discrete transitions. dL is the logic of hybrid programs, which is used to specify and verify safety properties. The formalism models hybrid systems with continuous and discrete dynamics in a unified formal language, thus providing a distinct advantage over other formalisms for relational reasoning. It can directly encode relational properties, providing more intuitive support for relational reasoning. Moreover, programming language based techniques for relational reasoning can be adapted to the setting of dL.

Using this advantage, we encode our timed simulation distances as forall-exists relational formulas in dL, inspired by the techniques of self-composition [14], [15]. Existing proof rules and axioms developed for dL can help reason with this encoding, but they lack support for timing and relational reasoning, especially, for dynamics. To address the challenges, we propose a framework that extends dL's capabilities to reason about relational properties and timing.

The main contributions of this paper are the following:

- A clean threat model of timed attacks in the setting of the differential dynamic logic (Section III).
- Quantitative timed notions of safety, robustness, attack impact, and attack tolerance for cyber-physical systems under timed attack (Section IV).
- To reason about above timed properties, we develop ad-hoc timed simulation distances which can be encoded and proven via untimed simulation distances, which are essentially forall-exists properties (Section V).
- A sound proof system for reasoning about general forall-exists properties (Section VI-A), and proof techniques to reason with the timing of dynamics (Section VI-B).
- A non-trivial case study, modeling a water tank system, to show the usefulness of the proposed proof techniques to reason with CPSs under timed attacks (Section VII).

We introduce preliminaries in Section II. Section VIII discusses related work, and Section IX concludes.

II. Preliminaries

A. Hybrid Programs and Differential Dynamic Logic

Hybrid programs [12], [13] are a formalism for modeling systems that have continuous and discrete behaviors. Fig. 1 gives the syntax and semantics for hybrid programs. The semantics of a program P is expressed as a transition relation $[\![P]\!]$ between states. For a set of variables $\mathbb{V}$, a *state* is a map $\omega : \mathbb{V} \mapsto \mathbb{R}$ assigning a real value to each variable. The set of all states is denoted by Sta. Thus, when $(\omega, \nu) \in [\![P]\!]$ then there is an execution of P that starts in ω and ends in ν.

Variables are real-valued and can be deterministically assigned ($x := \theta$, where θ is a real-valued term) or nondeterministically assigned ($x := *$). A hybrid program $x' = \theta \,\&\phi$ expresses the continuous evolution of variables: given the current value of variable x, the system follows the differential equation $x' = \theta$ for some (nondeterministically chosen) amount of time so long as the formula ϕ, the *evolution domain constraint*, holds for all of that time. Note that x can be a vector of variables and then θ is a vector of terms of the same dimension. Hybrid programs also include the operations of Kleene algebra with tests [16]: testing formulas, sequential composition, nondeterministic choice, and nondeterministic repetition.

Syntax

$\theta, \delta ::= x \mid c \mid \theta \oplus \delta$

$P, Q ::= x := \theta \mid x := * \mid x' = \theta \,\&\phi \mid ?\phi \mid P;Q \mid P \cup Q \mid P^*$

$\phi, \psi ::= \top \mid \theta \sim \delta \mid \neg\phi \mid \phi \wedge \psi \mid \forall x.\ \phi \mid [P]\phi$

Term semantics

$\omega[\![x]\!] = \omega(x)$

$\omega[\![c]\!] = c$

$\omega[\![\theta \oplus \delta]\!] = \omega[\![\theta]\!] \oplus \omega[\![\delta]\!]$ where $\oplus$ denotes arithmetics

Program semantics

$[\![x := \theta]\!] = \{(\omega,\nu) \mid \nu(x) = \omega[\![\theta]\!]$ and for all other variables $y \neq x$, $\nu(y) = \omega(y)\}$

$[\![x := *]\!] = \{(\omega,\nu) \mid \nu(y) = \omega(y)$ for all variables $y \neq x\}$

$[\![x' = \theta \,\&\phi]\!] = \{(\omega,\nu) \mid$ exists solution $\varphi : [0,r] \mapsto$ Sta of $x' = \theta$ with $\varphi(0) = \omega$ and $\varphi(r) = \nu$, and $\varphi(t) \in [\![\phi]\!]$ for all $t \in [0,r]\}$

$[\![?\phi]\!] = \{(\omega,\omega) \mid \omega \in [\![\phi]\!]\}$

$[\![P;Q]\!] = \{(\omega,\nu) \mid \exists \mu, (\omega,\mu) \in [\![P]\!]$ and $(\mu,\nu) \in [\![Q]\!]\}$

$[\![P \cup Q]\!] = [\![P]\!] \cup [\![Q]\!]$

$[\![P^*]\!] = [\![P]\!]^*$ the transitive, reflexive closure of $[\![P]\!]$

Formula semantics

$[\![\top]\!] =$ Sta

$[\![\theta \sim \delta]\!] = \{\ \omega \mid \omega[\![\theta]\!] \sim \omega[\![\delta]\!]\ \}$, $\sim$ denotes comparisons

$[\![\neg\phi]\!] =$ Sta $\setminus [\![\phi]\!]$

$[\![\phi \wedge \psi]\!] = [\![\phi]\!] \cap [\![\psi]\!]$

$[\![\forall x.\ \phi]\!] = [\![x := *]\!\phi]$

$[\![[P]\phi]\!] = \{\ \omega \mid \forall \nu$ if $(\omega,\nu) \in [\![P]\!]$ then $\nu \in [\![\phi]\!]\}$

Fig. 1. Syntax and semantics of hybrid programs and dL formulas

Differential dynamic logic (dL) [11], [12], [13] is the dynamic logic of hybrid programs. Fig. 1 also gives the semantics for dL formulas: $[\![\phi]\!]$ denotes the set of states satisfying formula ϕ. In addition to the standard logical connectives of first-order logic, dL includes primitive propositions that allow for comparisons of real-valued terms (which may include derivatives) and the *modality of necessity* $[P]\phi$, which holds in a state if and only if after any possible execution of hybrid program P, formula ϕ holds. The modality of necessity can be used to encode the *modality of existence*, i.e., $\langle P\rangle\phi = \neg[P]\neg\phi$.

The common (Boolean) version of *Safety properties* of a system are often defined as follows:

Definition 1 (Safety [7])**.** A hybrid program P is *safe for a postcondition ϕ_{post}, assuming a precondition ϕ_{pre}*, written $\text{SAFE}(P, \phi_{pre}, \phi_{post})$, if the formula $\phi_{pre} \rightarrow [P]\phi_{post}$ holds.

When modeling CPSs in dL, in order to explicitly represent the cyber component that controls a physical plant, the hybrid program P often has the form $(ctrl; plant)^*$, where $ctrl$ models logical actions of the controller and does not contain continuous terms; and $plant$ models the evolution of the physical environment and has the form of $x' = \theta \,\&\phi$. That is, a CPS is modeled as unbounded repetitions of a controller actions followed by an update to the physical environment.

For example, consider a simple cooled engine system that

operates in an environment where the temperature increases at a rate of 1 degree per minute, shown in Fig. 2. Let $temp_p$ be the actual *physical temperature* of the engine, expressed in degrees. The *sensed temperature* is modeled by introducing a different variable, $temp_s$, and the sensor reading is expressed via the assignment $temp_s := temp_p$ (here, for simplicity, we assume error-free sensor reading). Let t_g be the *global clock* that denotes time passing and t_l the *local clock* of the controller that is initialized at the beginning of each scan cycle.

The plant, of the form $x' = \theta \,\&\phi$, describes how the differential equation system $x' = \theta$ evolve for an arbitrary real duration within the region described by formula ϕ. In particular, temperature changes according to $delta$ (i.e., ${temp_p}' = delta$) and (global and local) time passes constantly (i.e., ${t_g}' = 1$ and ${t_l}' = 1$). The evolution domain constraint ϕ establishes the differential equations evolve within the time interval $t_l \leq 1$ (at most 1 minute) and if $temp_p$ is nonnegative (i.e. $temp_p \geq 0$). In practice, the requirement $t_l \leq 1$ fixes an upper bound to the *closed-loop latency*, i.e., the execution time for $ctrl; plant$.

The hybrid program $ctrl$ models the system controller. If the sensed temperature is above 100 degrees, the system activates cooling and the temperature drops at a rate of 0.5 degrees per minute (i.e., $delta := -0.5$). The controller does not activate cooling under other temperatures. Then the temperature would grow at the rate of 1 degree per minute (i.e., $delta := 1$).

Thus, according to Def. 1, when assuming the precondition of a temperature of 100 degrees, i.e. ϕ_{pre}, we want to ensure the temperature stays at no greater than 105 degrees, i.e., ϕ_{post}.

B. Distance Metrics

To conduct quantitative analysis, we define a notion of *distance between states*, focusing on a set $\mathcal{H} \subseteq \mathbb{V}$ of variables. Intuitively, variables in $\mathcal{H}$ are the ones that are relevant to the safety of the systems. And thus computing distance over these variables gives us the quantitative distance of interest. We use the *Euclidean metric* defined by $\rho_{\mathcal{H}}(\omega, \nu) = \sqrt{\sum_{x \in \mathcal{H}} (\omega(x) - \nu(x))^2}$.

We will write $\rho(\omega, \nu)$ for $\rho_{\mathcal{H}}(\omega, \nu)$ when $\mathcal{H} = \mathbb{V}$.

For a state ω and a real $\epsilon > 0$, the ball of ray ϵ centered in ω is the set of states $\mathbf{B}_{\mathcal{H}}(\omega, \epsilon) = \{\nu \mid \rho_{\mathcal{H}}(\omega, \nu) \leq \epsilon\}$.

We adopt existing notions [17], [18] to specify the distance between a state and a set of states:

- The distance between a state ω and a set of states $\mathcal{S} \subseteq \text{STA}$ is the *shortest* distance between ω and states in $\mathcal{S}$, that is,
$$\mathbf{dist}_{\mathcal{H}}(\omega, \mathcal{S}) = \mathbf{inf}\{\rho_{\mathcal{H}}(\omega, \nu) \mid \nu \in \mathcal{S}\}$$
- The *depth* of ω in $\mathcal{S} \subseteq \text{STA}$ is the *shortest* distance between ω and the *boundary* of $\mathcal{S}$, that is,
$$\mathbf{depth}_{\mathcal{H}}(\omega, \mathcal{S}) = \mathbf{inf}\{\rho_{\mathcal{H}}(\omega, \nu) \mid \nu \in (\text{STA} \setminus \mathcal{S})\}$$
- The *signed distance* between ω and a set of states $\mathcal{S} \subseteq \text{STA}$ is defined as follows:
$$\mathbf{Dist}_{\mathcal{H}}(\omega, \mathcal{S}) = \begin{cases} \mathbf{depth}_{\mathcal{H}}(\omega, \mathcal{S}), & \text{if } \omega \in \mathcal{S} \\ -\mathbf{dist}_{\mathcal{H}}(\omega, \mathcal{S}), & \text{if } \omega \notin \mathcal{S} \end{cases}$$
Note that in the first case the signed distance is a positive real number, while in the second case the signed distance is negative. Thus, $\mathbf{Dist}_{\mathcal{H}}(\omega, \mathcal{S}) > 0$ implies that $\mathbf{B}_{\mathcal{H}}(\omega, \epsilon) \subseteq \mathcal{S}$ for all $\epsilon < \mathbf{Dist}(\omega, \mathcal{S})$, whereas $\mathbf{Dist}_{\mathcal{H}}(\omega, \mathcal{S}) < 0$ implies that $\mathbf{B}_{\mathcal{H}}(\omega, \epsilon) \subseteq (\text{STA} \setminus \mathcal{S})$ for all $\epsilon < -\mathbf{Dist}_{\mathcal{H}}(\omega, \mathcal{S})$. $\mathbf{Dist}_{\mathcal{H}}(\omega, \mathcal{S}) = 0$ is not very informative.

Here, we assume $\mathbf{inf}\ \emptyset = \infty$ and $\mathbf{inf}\ \mathbb{R} = -\infty$, and we let operator $\mathbf{inf}$ in $\mathbb{R} \cup \{\infty, -\infty\}$, thus every set has an infimum.

$$\begin{aligned} \phi_{pre} &\equiv temp_p = 100 \\ \phi_{post} &\equiv temp_p \leq 105 \\ ctrl &\equiv t_l := 0 \ ; temp_s := temp_p; \\ &\quad (?temp_s > 100; delta := -0.5\,) \\ &\quad \cup (?temp_s \leq 100; delta := 1) \\ plant &\equiv {temp_p}' = delta, {t_l}' = 1, {t_g}' = 1 \,\&(temp_p \geq 0 \wedge t_l \leq 1) \end{aligned}$$

Fig. 2. $\mathsf{d}\mathcal{L}$ model of a simple cooled engine system

III. A Threat Model for Timed attacks

In this section, we focus on *timed physics-based attacks* [1], [2] capable of: (i) modifying/dropping sensor readings coming from the plant; (ii) forging/dropping actuator commands addressed to the plant. These attacks are implemented through malware injected in the controllers of CPSs, with the following limitations: Any alterations of sensor signals and/or actuator commands at network level, or within the physical devices, are not represented. Such attacks can be easily simulated by writing malware that can modify sensor measurements and/or actuator commands within the controller.

Thus, the *attacker objectives* can be resumed in affecting the runtime evolution of the controlled physical process.

The focus of the paper is on timed attacks because *timing* is a critical issue when attacking systems with a physical state that changes continuously over time. In fact, some states may be more vulnerable to attacks than others [19], [20]. Furthermore, not only the timing of the attack, but also the *duration of the attack* is an important parameter to be taken into account in order to achieve a successful attack [20], [2]. For example, the duration of an attack should always be determined to ensure *stealthy attacks*, i.e., attacks that cannot be detected by *intrusion detection systems* (IDS).

In general, a clean way to model a physics-based attack on a CPS is as a function that, given a genuine system returns a compromised one, where the corruption affects the logical component, and in particular the controller, rather than the plant, which is usually harder to be reached by the attacker.

Definition 2 (Modeling attacks)**.** An attack $\texttt{ATT}$ is a function that transforms a program $P \equiv ctrl; plant$ into a new compromised program $\texttt{ATT}(P) \equiv (\texttt{ATT}(ctrl); plant)$. Moreover, for $P^* \equiv (ctrl; plant)^*$ we define $\texttt{ATT}(P^*) \equiv (\texttt{ATT}(ctrl); plant)^*$.

In the following, we use the example CPS described in Fig. 2 to show a way to model timed attacks in $\mathsf{d}\mathcal{L}$. The so-called *scheduled attacks* are attacks scheduled at a precise point in time and for a precise duration. Scheduled attacks may have a recurrent malicious pattern. This is the case of *periodic attacks* which are slightly more subtle as

$$
\begin{aligned}
ctrl_{comp} &\equiv t_l := 0\,;\,\text{MALWARE}\,;\quad (?temp_s \le 100; delta := 1)\\
&\qquad\qquad \cup (?temp_s > 100; delta := -0.5)\\
\text{MALWARE} &\equiv ?\text{TRIGGER}\,;\,\text{PLD}_\text{X} \cup ?\neg\text{TRIGGER}\,;\,temp_s := temp_p\\
\text{TRIGGER} &\equiv \text{START} \le t_g \le \text{START} + \text{DURATION}\\
\text{PLD}_\text{DoS} &\equiv ?\top\\
\text{PLD}_\text{IC} &\equiv temp_s := *\,;\\
&\qquad ?(temp_s \ge temp_p - \text{OFFSET} \wedge temp_s \le temp_p)\\
\text{PLD}_\text{IV} &\equiv o := o + inc\,;\\
&\qquad (?o > \text{OFFSET}\,;\,\text{PLD}_\text{IC})\\
&\quad \cup (?o \le \text{OFFSET}\,;\,temp_s := *\,;\\
&\qquad ?(temp_s \ge temp_p - o \wedge temp_s \le temp_p)\ )
\end{aligned}
$$

Fig. 3. Scheduled sensor attacks

they alternate periods of malicious activities (during the so-called *attack window*) with periods in which the attack is harmless (during the *sleep window*). This switching between malicious and benign activities, if well-calibrated, may make the intrusion detection activity quite difficult, inducing system engineers in suspecting their IDSs are suffering false positives, possibly due to too tight thresholds.

Here, by no means, we pretend to be exhaustive with our templates: we may design dozens of different variations to derive smarter attacks. However, we argue that our attacks are general enough for testing and verifying the timed notions of safety and robustness which we will define in the next section.

A. Scheduled Attacks

1) Sensor attacks: In this section, we model both DoS and (bounded) integrity attacks on sensor measurements in dℒ. In order to help the reader's intuition, we implement these attacks in the simple cooled engine system modeled in Fig. 2, where the only measurement is the temperature of the system.

We present the template of scheduled sensor attacks in Fig. 3. Here, $ctrl_{comp}$ denotes the controller $ctrl$ (shown in Fig. 2), compromised by a malware that is triggered at a specific time START and remains active for a duration DURATION. In our model, the malware payload can be instantiated with three different payloads: PLD_DoS, PLD_IC and PLD_IV. The first one implements *denial of service* of the measurement readings. Thus, during the attack, the controller will not get updated measurements. The second payload inserts a bounded *subtractive offset* in the measurements (additive offsets can be modeled in a similar manner); more precisely, when the malware is triggered, the controller will get measurements reduced by an arbitrary offset that is bounded by a *constant* OFFSET, determining the impact of the attack. Notice that when the attack starts, the measurement will exhibit a discontinuity that might be detected by an anomaly-based intrusion detection system. The third payload PLD_IV implements a more clever, and possibly stealthier, integrity attack. Here, the bound of the subtractive offset is not constant, but it is gradually increased until the desired maximum offset OFFSET is reached.

2) Actuator attacks: In Fig. 4, we adapt two forms of attacks to the simple cooled engine system modeled in Fig. 2.

$$
\begin{aligned}
ctrl_{comp} &\equiv t_l := 0\ ;temp_s := temp_p\,;\\
&\qquad (?temp_s > 100; \text{MALWARE}_\text{ON}\ )\\
&\qquad \cup\ (?temp_s \le 100; \text{MALWARE}_\text{OFF})\\
\text{MALWARE}_\text{ON} &\equiv ?\text{TRIGGER}\,;\,\text{PLD}_\text{X} \cup ?\neg\text{TRIGGER}\,;\,delta := -0.5\\
\text{MALWARE}_\text{OFF} &\equiv ?\text{TRIGGER}\,;\,\text{PLD}_\text{DoS} \cup ?\neg\text{TRIGGER}\,;\,delta := 1\\
\text{TRIGGER} &\equiv \text{START} \le t_g \le \text{START} + \text{DURATION}\\
\text{PLD}_\text{DoS} &\equiv ?\top\\
\text{PLD}_\text{IA} &\equiv delta := *\,;?(delta \ge -0.5 + \text{OFFSET} \wedge delta \le 1)
\end{aligned}
$$

Fig. 4. Scheduled actuator attacks

Here, there is only one actuator used to activate/deactivate the cooling system. Thus, the malware may be interested in interfering with the activation of the cooling system (via MALWARE_ON) or its deactivation (via $\text{MALWARE}_\text{OFF}$). In the latter case, our malware can only do a *denial of service* on the deactivation, dropping the command sent by the controller to deactivate the cooling system. As a consequence, the cooling system remains active, keeping the temperature in a drop. In the former case, the malware payload can be instantiated with two different codes: PLD_DoS and PLD_IA. The first implements *denial of service* on the command to activate the cooling system. As a consequence, the cooling will not be activated and the temperature will continue to grow. The second payload inserts a bounded *additive offset* in the actuator command that specifies the intensity of cooling during the attack; more precisely, when malware is triggered, cooling is less effective than usual in lowering the temperature by an arbitrary offset that is bounded by a *constant* OFFSET, determining the impact of the attack (the code to add a subtractive offset is similar).

B. Periodic Attacks

1) Sensor attacks: The template of periodic sensor attacks in Fig. 5 is a refinement of that one seen for scheduled sensor attacks in Fig. 3. Here, again, the malware is triggered at a specific time START and remains active for a duration DURATION. In this time window, the malware will alternate active windows/phases (lasting AW time units) and sleep windows/phases (lasting SW time units). As as example, we may consider both DoS and bounded integrity attacks given in Fig. 3 (PLD_DoS and PLD_IC). In both kinds of attack, during the attack window of the attack, the controller will not get correct measurements, while the system works correctly during the sleeping phase.

2) Actuator attacks: Similarly to periodic sensor attacks, periodic actuator attacks can be expressed as a refinement of the scheduled actuator attacks given in Fig. 4. This can be easily done by rewriting the malicious payloads in such a way to be periodic, as we did for periodic sensor attacks (Fig. 5.).

IV. Safety, Robustness, Attack Impact and Tolerance

A. Extend dℒ with Timed Structure

Timed semantics To help define a timed notion of safety, we extend dℒ with another syntactic structure of formulas,

$$
\begin{aligned}
\text{MALWARE} \equiv\ & ?\text{TRIGGER}\,;\,\text{PLD} \\
& \cup ?\neg\text{TRIGGER}\,;\, temp_s := temp_p \\
\text{TRIGGER} \equiv\ & \text{START} \leq t_g \leq \text{START} + \text{DURATION} \\
\text{PLD} \equiv\ & ?\text{ACTIVE}\,;\,\text{PLD}_\text{X}\,;\,\text{ACTIVE\_PHASE} \\
& \cup ?\neg\text{ACTIVE}\,;\, temp_s := temp_p\,;\,\text{SLEEP\_PHASE} \\
\text{ACTIVE\_PHASE} \equiv\ & ?(t_g - t_\text{ON} < \text{AW}) \\
& \cup ?(t_g - t_\text{ON} \geq \text{AW})\,;\,\text{ACTIVE} := \text{false}\,;\, t_\text{OFF} := t_g \\
\text{SLEEP\_PHASE} \equiv\ & ?(t_g - t_\text{OFF} < \text{SW}) \\
& \cup ?(t_g - t_\text{OFF} \geq \text{SW})\,;\,\text{ACTIVE} := \text{true}\,;\, t_\text{ON} := t_g
\end{aligned}
$$

Fig. 5. Periodic sensor attacks

denoted $\phi\langle P\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}$, which intuitively represents the *timed strongest postcondition* after the execution of P, in the time interval $[\mathtt{T}_l,\mathtt{T}_u]$, in a state satisfying the precondition ϕ. Intuitively, we assume that our hybrid programs P have a *global clock* t_g representing physical time. Thus, a common form of P is $(ctrl;plant)^*$ where $plant \equiv (x' = \theta, {t_g}' = 1)\,\&\psi$. Here, x can be a vector of variables, and then θ is a vector of terms of the same dimension. An obvious consequence of our representation of time in hybrid programs is that the safety property $\phi_{pre} \rightarrow [P]\phi_{post}$ requires that $\phi_{pre} \rightarrow t_g = 0$, where $t_g = 0$ sets the initial time of the system to 0.

Well-formedness: The formula $\phi\langle P\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}$ is well-formed if (1) $t_g \in \text{VAR}(P)$, (2) ${t_g}' = 1$ is the only program that modifies t_g, and (3) $\mathtt{T}_l \leq \mathtt{T}_u$.

For the rest of this paper, we focus on well-formed timed strongest postconditions unless otherwise specified. Its formal semantics is the following:

$$[\![\phi\langle P\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!] = \{\nu \mid \exists\,\omega \in [\![\phi]\!],\, (\omega,\nu) \in [\![P]\!] \wedge \mathtt{T}_l \leq \nu(t_g) \leq \mathtt{T}_u\}\,.$$

We will write $\phi\langle P\rangle$ as an abbreviation for $\phi\langle P\rangle_{[0,\infty]}$.

B. Timed Quantitative Safety

In the following, we provide a timed quantitative generalization of the safety property (Definition 1) of the system under investigation in a specific time interval $[\mathtt{T}_l,\mathtt{T}_u]$. In particular, for $\mathtt{T}_l = 0$, the following definition provides a sort of "look-ahead" on the safety of the system in the next $\mathtt{T}_u$ time instants.

Definition 3 (Timed quantitative safety)**.** Given two time points $\mathtt{T}_l, \mathtt{T}_u \in \mathbb{R}_{>0}$, a real $u \in \mathbb{R}$, and formulas ϕ_{pre} and ϕ_{post}, with $\mathcal{H} \equiv \text{VAR}(\phi_{post})$, we will say that a hybrid program P is u-safe for ϕ_{pre} and ϕ_{post} in the time interval $[\mathtt{T}_l,\mathtt{T}_u]$, denoted $\text{T-SAFE}_u^{[\mathtt{T}_l,\mathtt{T}_u]}(P,\phi_{pre},\phi_{post})$, if

$$u = \mathbf{inf}\{\,\mathbf{Dist}_{\mathcal{H}}(\nu,[\![\phi_{post}]\!]) \mid \nu \in [\![\phi_{pre}\langle P\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!]\,\}\,.$$

Given a hybrid program P, a precondition ϕ_{pre} and a time interval $[\mathtt{T}_l,\mathtt{T}_u]$, the real number u measures the *shortest* distance between the reachable states by the program within this duration, i.e., $[\![\phi_{pre}\langle P^{[\mathtt{T}_l,\mathtt{T}_u]}\rangle]\!]$, and the set of unsafe states. If u is positive, then all reachable states in the interval considered by the system P from initial states satisfying the precondition ϕ_{pre} stay safe. The bigger u, the safer the system. In contrast, if u is negative, then some reachable states violate the safety condition ϕ_{post}. If u is 0, then the system cannot be considered safe as its safety may depend on very small perturbations of the system variables [17].

C. Timed Quantitative Robustness and Attack Impact

Now, suppose that system engineers checked the safety of their system and got $\text{T-SAFE}_u^{[\mathtt{T}_l,\mathtt{T}_u]}(P,\phi_{pre},\phi_{post})$, with $u > 0$. Furthermore, suppose that at time $\mathtt{T}_l < \mathtt{T} < \mathtt{T}_u$, the IDS detects an attack $\mathtt{ATT}(P)$. The question is: Can the system deal with this kind of attack for the next $\mathtt{T}_u - \mathtt{T}$ time instants? In other words, is the system robust enough when exposed to such attacks? The following definitions provide some answers.

Definition 4 (Timed quantitative robustness)**.** Given a hybrid program P, time points $\mathtt{T}_l$ and $\mathtt{T}_u$, $u, u_1, \delta \in \mathbb{R}$, and formulas ϕ_{pre} and ϕ_{post}, we say that P is δ-robust in the time interval $[\mathtt{T}_l,\mathtt{T}_u]$ when exposed to attack $\mathtt{ATT}$, written $\text{T-ROBUST}_\delta^{[\mathtt{T}_l,\mathtt{T}_u]}(\mathtt{ATT}(P),\phi_{pre},\phi_{post})$, if

- $\text{T-SAFE}_u^{[\mathtt{T}_l,\mathtt{T}_u]}(P,\phi_{pre},\phi_{post})$, with $u > 0$
- $\text{T-SAFE}_{u_1}^{[\mathtt{T}_l,\mathtt{T}_u]}(\mathtt{ATT}(P),\phi_{pre},\phi_{post})$
- $\delta = u_1/u$.

For notation, we write $\text{T-ROBUST}_\delta^{\mathtt{T}}(\mathtt{ATT}(P),\phi_{pre},\phi_{post})$ to mean $\text{T-ROBUST}_\delta^{[0,\mathtt{T}]}(\mathtt{ATT}(P),\phi_{pre},\phi_{post})$.

Now, if δ is a positive number, then we can say that the attack does not compromise the safety of the system under attack because our system is robust enough. However, if $\delta < 0$ then the system under attack may reach unsafe states, and we may be interested in getting an estimate of the impact of the attack on the safety of the system.

Definition 5 (Timed quantitative impact)**.** Given a hybrid program P, time points $\mathtt{T}_l$ and $\mathtt{T}_u$, real numbers $u, u_1, \delta \in \mathbb{R}$, and formulas ϕ_{pre} and ϕ_{post}, we say that an attack $\mathtt{ATT}$ has an impact δ on program P in the time interval $[\mathtt{T}_l,\mathtt{T}_u]$, written $\text{T-IMPACT}_\delta^{[\mathtt{T}_l,\mathtt{T}_u]}(\mathtt{ATT}(P),\phi_{pre},\phi_{post})$, if

- $\text{T-SAFE}_u^{[\mathtt{T}_l,\mathtt{T}_u]}(P,\phi_{pre},\phi_{post})$, with $u > 0$
- $\text{T-SAFE}_{u_1}^{[\mathtt{T}_l,\mathtt{T}_u]}(\mathtt{ATT}(P),\phi_{pre},\phi_{post})$
- $\delta = 0$, if $u_1 > 0$; $\delta = |u_1/u|$, if $u_1 < 0$.

Again, we write $\text{T-IMPACT}_\delta^{\mathtt{T}}(\mathtt{ATT}(P),\phi_{pre},\phi_{post})$ to mean $\text{T-IMPACT}_\delta^{[0,\mathtt{T}]}(\mathtt{ATT}(P),\phi_{pre},\phi_{post})$.

D. Timed Quantitative Attack Tolerance

Here, it should be said that, for several reasons, certain attacks might be not too nasty and they might have a limited impact on the system only for a relatively small time interval in which the system reaches unsafe states. In this case, system engineers might decide not to take action to avoid stopping production, accepting a bounded degree of unsafety for a limited period of time [2].

Definition 6 (Timed quantitative tolerance)**.** Given a hybrid program P, time points $\mathtt{T}$ and $\mathtt{T}_3$, with $\mathtt{T} < \mathtt{T}_3$, a real number $u \in \mathbb{R}$, and properties ϕ_{pre} and ϕ_{post}, we say that in the time interval $[\mathtt{T},\mathtt{T}_3]$ the program P exposed to an attack $\mathtt{ATT}$ *tolerates* a potential unsafety u lasting for a time interval at most $\mathtt{TH}$ long, written as $\text{T-TOLERANT}_u^{[\mathtt{T},\mathtt{T}_3]}(\mathtt{ATT}(P),\phi_{pre},\phi_{post},\mathtt{TH})$, if there are time instants $\mathtt{T}_1$, $\mathtt{T}_2$, with $\mathtt{T} < \mathtt{T}_1 < \mathtt{T}_2 < \mathtt{T}_3$, such that:

- T-SAFE$^{[\mathrm{T},\mathrm{T}_1]}_{u_1}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post})$, for some $u_1 > 0$
- T-SAFE$^{[\mathrm{T}_1,\mathrm{T}_2]}_{u_2}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post})$, for some $u_2 \geq u$
- T-SAFE$^{[\mathrm{T}_2,\mathrm{T}_3]}_{u_3}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post})$, for some $u_3 > 0$
- $\mathrm{T}_2 - \mathrm{T}_1 \leq \mathrm{TH}$.

Here, within the time interval $[\mathrm{T}, \mathrm{T}_3]$ we have a potentially unsafe sub-interval $[\mathrm{T}_1, \mathrm{T}_2]$, whose length is bounded by a threshold TH, where the (degree of) unsafety is bounded by a constant u. We sometimes write T-TOLERANT$^{[\mathrm{T},\mathrm{T}_3]}_{u}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post}, \mathrm{TH}, \mathrm{T}_1, \mathrm{T}_2)$ to explicitly point out the timing instants T_1 and T_2.

In the following example, we show how the above definitions apply to the scheduled attacks proposed in Fig. 3. We recall that P^* denotes the genuine cooled engine system, where $P \equiv ctrl; plant$ is defined in Fig. 2. Furthermore, according to the dynamic of our engine system, the time units mentioned in the examples are expressed in minutes.

Example 1 (Scheduled integrity attack with constant bound)**.** In this example, we will focus on the attack in Fig. 3, in which the malware is instantiated with the second payload, $\mathrm{PLD}_{\mathrm{IC}}$, to implement an integrity attack on the measurements with a constant offset bound. We reason in the worst-case scenario, i.e., we assume that the attack starts when the temperature of the system reaches 101 degrees.

- Let us assume OFFSET $= 6$ and DURATION $= 2$. We recall that the malware is triggered until $t_g =$ START $+$ DURATION (this is the last moment when the attack may have a go). In this case, in the worst case scenario the plant takes 1 minute, and at time START $+$ DURATION $+ 1$ the temperature may reach 104 degrees because during the attack the controller will keep sensing a temperature less than or equal to 100 degrees and it will not activate the cooling system. When the attack is over, three minutes after the START, the controller will sense the real (high) temperature and it will activate the cooling system. The robustness of the system under attack is 1, at any time T, with T $<$ START, while it drops to $\frac{1}{4}$ at any time T, with T $\geq$ START $+$ DURATION $+ 1$; in this case the impact of the attack on the system is 0 at any time T. More formally,
  - T-ROBUST$^{\mathrm{T}}_{1}(\mathtt{ATT}(P^*), \phi_{pre}, \phi_{post})$ for $0 <$ T $<$ START
  - T-ROBUST$^{\mathrm{T}}_{\frac{1}{4}}(\mathtt{ATT}(P^*), \phi_{pre}, \phi_{post})$ for any point in time T $\geq$ START $+$ DURATION $+ 1$
  - T-IMPACT$^{\mathrm{T}}_{0}(\mathtt{ATT}(P^*), \phi_{pre}, \phi_{post})$ for any T > 0.
- Let us assume OFFSET $= 6$ and DURATION $= 5$. In this case, the temperature may reach 105 degrees after 4 minutes, and 107 degrees after 6 minutes. Then, the cooling system will be activated. Here, the robustness of the system under attack is 1, at any time T, with T $<$ START, but it drops to $-\frac{2}{4}$, at any time T, with T $\geq$ START $+$ DURATION $+ 1$. In this case the impact of the attack on the system is $\frac{2}{4}$, as the system under attack may reach at most the temperature 107. Notice that once the attack terminates, it will require 4 time instants of cooling to drop the temperature to 105 degrees. Formally,
  - T-ROBUST$^{\mathrm{T}}_{1}(\mathtt{ATT}(P^*), \phi_{pre}, \phi_{post})$ for $0 <$ T $<$ START,
  - T-ROBUST$^{\mathrm{T}}_{-\frac{2}{4}}(\mathtt{ATT}(P^*), \phi_{pre}, \phi_{post})$ for T$\geq$START $+ 6$.
  - T-IMPACT$^{\mathrm{T}}_{\frac{2}{4}}(\mathtt{ATT}(P^*), \phi_{pre}, \phi_{post})$, for T $\geq$ START $+ 6$.
  - T-TOLERANT$^{[0,\mathrm{T}]}_{-2}(\mathtt{ATT}(P^*), \phi_{pre}, \phi_{post}, \mathrm{TH})$, for TH $=$ START $+$ DURATION $+ 1 + 4 -$ (START $+ 4$) $= 6$ and T $>$ START $+$ DURATION $+ 5$.

  The last statement says that the attack drags the system into unsafe states for at most 6 time instants during the time interval [START $+ 4$, START $+$ DURATION $+ 1 + 4$], with an (un)safety -2. However, after time START$+$DURATION$+5$ the system under attack recovers safety.

More examples can be found in the Appendix (Section A).

V. Establishing Timed Properties

For a system under timed attack, we can establish timed properties, i.e., Def. 4, Def. 5, and Def. 6 by directly computing the timed safety of the system under attack. However, the computation of timed safety may be difficult, as it requires computing the infimum distance from all admissible states. This is particularly difficult for a system under timed attack, due to (1) the complications caused by the attack and (2) the interplay between system dynamics and timing aspects.

Thus, instead of directly reasoning about the timed safety for a compromised system, we define a notion of distance to measure the behavioral differences between the compromised system and the genuine one, and then use the safety of the genuine system to derive the safety of the compromised one. In particular, we use *simulation distance*, in the style of Chong et al. [9] to measure an upper bound of the behavioral distance between two programs. Intuitively, programs P and Q are in simulation at distance d if given the same initial condition, Q can mimic the behaviors of P, i.e., Q is able to reach states whose distance from those reached by P is at most d. Using simulation distance, we can derive a lower bound of safety for the compromised system without direct computation of the infimum distance from all the states reached. Moreover, the relational nature of simulation distance helps system engineers pinpoint and focus on specific components or behaviors where the compromised system diverges from the genuine one, making it more practical for reasoning with complex systems.

In this section, we first introduce the notion of timed simulation distance (Section V-A), and then we show how to use this notion to establish the introduced timed properties (Section V-B). After that, we show how to reduce proving the timed distance to proving untimed distance (Section V-C).

A. Timed Simulation Distance

In order to provide proof techniques to reason about our timed properties, we propose a notion of timed simulation distance obtained by refining an untimed version, proposed in [9].

Definition 7 (Untimed vs. Timed simulation distance)**.** For programs P, Q, a formula ϕ_{pre}, and a variable set $\mathcal{H}$, we say:

- P and Q are at distance d with respect to ϕ_{pre} and $\mathcal{H}$, written $P \sqsubseteq_{\phi_{pre},\mathcal{H},d} Q$, if for each state $\nu_1 \in [\![\phi_{pre}\langle P\rangle]\!]$, there exists a state $\nu_2 \in [\![\phi_{pre}\langle Q\rangle]\!]$ such that $\rho_{\mathcal{H}}(\nu_1, \nu_2) \leq d$.

- P and Q are at distance d with respect to ϕ_{pre} and $\mathcal{H}$ for the interval $[\mathtt{T}_1, \mathtt{T}_2]$, written $P \sqsubseteq^{[\mathtt{T}_1,\mathtt{T}_2]}_{\phi_{pre},\mathcal{H},d} Q$, if for each state $\nu_1 \in [\![\phi_{pre}\langle P\rangle_{[\mathtt{T}_1,\mathtt{T}_2]}]\!]$ there exists a state $\nu_2 \in [\![\phi_{pre}\langle Q\rangle]\!]$ such that $\rho_{\mathcal{H}}(\nu_1,\nu_2) \leq d$.

For a program P, an attack $\mathtt{ATT}$, and an interval $[\mathtt{T}_l, \mathtt{T}_u]$, the timed simulation $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_l,\mathtt{T}_u]}_{\phi_{pre},\mathcal{H},d} P$ expresses that for each state ν_1 reachable by the system under attack $\mathtt{ATT}(P)$ within the interval $[\mathtt{T}_l, \mathtt{T}_u]$, from some initial states in $[\![\phi_{pre}]\!]$, there is a state ν_2 reachable by P from some initial state in $[\![\phi_{pre}]\!]$, such that ν_1 and ν_2 are at distance at most d, for a fixed variable set $\mathcal{H}$. The distance d gives an upper bound on the perturbation introduced by the attack on the safety of the behaviors originating from $[\![\phi_{pre}]\!]$. The set $\mathcal{H}$ here often refers to variables that are relevant to the system's postcondition, i.e., $\textsc{Var}(\phi_{post})$.

Remark 1. Note that in Definition 7 we choose $[\![\phi_{pre}\langle Q\rangle]\!]$ for the program Q (which often represents the genuine system), rather than other options, such as $\phi_{pre}\langle Q\rangle_{[\mathtt{T}_1,\mathtt{T}_2]}$ or $\phi_{pre}\langle Q\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}$ for some $\mathtt{T}_l \leq \mathtt{T}_u$. This design choice is reasonable to us since the genuine system may require different timing to simulate the behavior of the compromised system.

Our timed simulation distance can be used to establish *upper bounds* on the loss of timed safety caused by an attack:

Theorem 1. Given a hybrid program P, a time interval $[\mathtt{T}_l, \mathtt{T}_u]$, reals $d, u > 0$, and formulas ϕ_{pre} and ϕ_{post}. If
- $\text{T-SAFE}^{[0,\infty]}_{u}(P, \phi_{pre}, \phi_{post})$, and
- $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_l,\mathtt{T}_u]}_{\phi_{pre},\mathcal{H},d} P$, for $\mathcal{H} = \textsc{Var}(\phi_{post})$

then $\text{T-SAFE}^{[\mathtt{T}_l,\mathtt{T}_u]}_{u_1}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post})$, for some $u_1 \geq u - d$.

The first item $\text{T-SAFE}^{[0,\infty]}_{u}(P, \phi_{pre}, \phi_{post})$ basically deals with all reachable states of P. The distance d proven in the second item leads to a *lower bound* of the safety for the attacked program $\mathtt{ATT}(P)$, as $u_1 \geq u - d$.

B. Reasoning about Timed Properties

Assuming we know the quantitative safety of the genuine system, i.e. $\text{T-SAFE}^{[0,\infty]}_{u}(P, \phi_{pre}, \phi_{post})$ (whose u is no greater than u_1 of $\text{T-SAFE}^{[\mathtt{T}_l,\mathtt{T}_u]}_{u_1}(P, \phi_{pre}, \phi_{post})$ for any valid interval $[\mathtt{T}_l, \mathtt{T}_u]$). Then for an attack $\mathtt{ATT}$, we can establish the properties of timed robustness (Def. 4), impact (Def. 5), and tolerance (Def. 6), with the help of Thm. 1, as follows:

Timed robustness Now, if we prove $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_l,\mathtt{T}_u]}_{\phi_{pre},\mathcal{H},d} P$ for $\mathcal{H} = \textsc{Var}(\phi_{post})$ and some d, then using Thm. 1 we can derive a *lower bound* on timed robustness.

Corollary 1. Given a hybrid program P, a time interval $[\mathtt{T}_l, \mathtt{T}_u]$, reals $d, u > 0$, and formulas ϕ_{pre} and ϕ_{post}. If
- $\text{T-SAFE}^{[0,\infty]}_{u}(P, \phi_{pre}, \phi_{post})$, and
- $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_l,\mathtt{T}_u]}_{\phi_{pre},\mathcal{H},d} P$, for $\mathcal{H} = \textsc{Var}(\phi_{post})$

then $\text{T-ROBUST}^{[\mathtt{T}_l,\mathtt{T}_u]}_{\delta}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post})$ for $\delta \geq (u-d)/u$.

Timed impact Similarly, relying on Thm. 1, we can derive an *upper bound* on the timed impact of a program:

Corollary 2. Given a hybrid program P, a time interval $[\mathtt{T}_l, \mathtt{T}_u]$, reals $d, u > 0$, and formulas ϕ_{pre} and ϕ_{post}. If
- $\text{T-SAFE}^{[0,\infty]}_{u}(P, \phi_{pre}, \phi_{post})$, and
- $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_l,\mathtt{T}_u]}_{\phi_{pre},\mathcal{H},d} P$, for $\mathcal{H} = \textsc{Var}(\phi_{post})$

then $\text{T-IMPACT}^{[\mathtt{T}_l,\mathtt{T}_u]}_{\delta}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post})$, with $\delta = 0$, if $d - u < 0$, and $\delta \leq \frac{(d-u)}{u}$, if $d - u > 0$.

Timed tolerance In order to derive a *lower bound* on timed tolerance (Def. 6), we use the following result based on Thm. 1.

Corollary 3. Given a hybrid program P, an attack $\mathtt{ATT}$, time points $\mathtt{T}$ and $\mathtt{T}_3$, a time duration $\mathtt{TH}$, real number $u > 0$, properties ϕ_{pre} and ϕ_{post}, and a variable set $\mathcal{H} = \textsc{Var}(\phi_{post})$. If there exist time points $\mathtt{T}_1$ and $\mathtt{T}_2$ such that
- $\text{T-SAFE}^{[0,\infty]}_{u}(P, \phi_{pre}, \phi_{post})$
- $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T},\mathtt{T}_1]}_{\phi_{pre},\mathcal{H},d_1} P$ for some d_1 such that $u - d_1 > 0$
- $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_1,\mathtt{T}_2]}_{\phi_{pre},\mathcal{H},d_2} P$
- $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_2,\mathtt{T}_3]}_{\phi_{pre},\mathcal{H},d_3} P$ for some d_3 that $u - d_3 > 0$
- $\mathtt{T}_2 - \mathtt{T}_1 \leq \mathtt{TH}$

then $\text{T-TOLERANT}^{[\mathtt{T},\mathtt{T}_3]}_{u-d_2}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post}, \mathtt{TH})$.

Note that in Corollary 3, the second, third, and fourth items correspond to the safety requirements in Def. 6.

We are often interested in proving that the system is tolerant from the very beginning to a time point $\mathtt{T}$, that is, $\text{T-TOLERANT}^{[0,\mathtt{T}]}_{u_2}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post}, \mathtt{TH})$.

C. Proving Timed Simulation Distance via Untimed Distance

Now, timed properties can be established by proving timed distances, i.e., $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_l,\mathtt{T}_u]}_{\phi_{pre},\mathcal{H},d} P$, which concerns the reachable states of $\mathtt{ATT}(P)$ within a time interval. To reason about timing, we encode it as part of the dL programs, and use the following theorems to reduce proving timed distances to proving untimed distances. The latter is easier to reason with, and benefits from techniques in existing work, e.g., [21].

The following theorem relates timed and untimed distances:

Theorem 2 (Timed vs. untimed simulation distance)**.** For a hybrid program P with a global clock t_g, a program Q, a formula ϕ_{pre}, and a time point $\mathtt{T} \geq 0$, it holds that:

$$(?0 \leq t_g \leq \mathtt{T}; P)^* \sqsubseteq_{\phi_{pre},\mathcal{H},d} Q \;\rightarrow\; P^* \sqsubseteq^{[0,\mathtt{T}]}_{\phi_{pre},\mathcal{H},d} Q\,.$$

The following two propositions help reason with programs including conditionals on the global clock. They are often useful in splitting the (compromised) system into segments based on whether an attack is active.

Proposition 1. For a hybrid program P with a global clock t_g, a formula ϕ, and timing points $0 \leq \mathtt{T}_1 \leq \mathtt{T}_2$, it holds that: $[(?0 \leq t_g < \mathtt{T}_1; P)^*; (?\mathtt{T}_1 \leq t_g \leq \mathtt{T}_2; P)^*]\phi \leftrightarrow [(?0 \leq t_g \leq \mathtt{T}_2; P)^*]\phi$

Proposition 2. For a program P with a global clock t_g, a formula ϕ, and timing points $0 \leq \mathtt{T}_1 \leq \mathtt{T}_2$, it holds that: $[(?t_g < \mathtt{T}_1; P)^*; (?\mathtt{T}_1 \leq t_g \leq \mathtt{T}_2; P)^*; (?t_g > \mathtt{T}_2; P)^*]\phi \leftrightarrow [P^*]\phi$

With the help of Theorem 2, we can focus on proving the non-timed version of simulation distance, i.e., $\mathtt{ATT}(P) \sqsubseteq_{\phi_{pre},\mathcal{H},d} P$, which is essentially a *forall-exists* relational property on two programs. Next, we introduce a system of techniques that can be used to reason with such a property, with a focus on timing aspects.

VI. Proving Untimed Distance

In this section, we present a system of mechanisms dedicated for reasoning with forall-exists properties in the setting

$\forall\exists$-DEF
$[\![(P \parallel Q)_\xi]\!\rangle\phi \leftrightarrow [P]\langle\xi(Q)\rangle\phi$

$\forall\exists$-?-L
$(\psi \to \langle\xi(Q)\rangle\phi) \leftrightarrow [\![(?\psi \parallel Q)_\xi]\!\rangle\phi$

$\forall\exists$-?-R
$([P](\xi(\psi) \wedge \phi)) \leftrightarrow [\![(P \parallel ?\psi)_\xi]\!\rangle\phi$

$\forall\exists$-;
$$\frac{\Gamma \vdash [\![(P_1 \parallel Q_1)_\xi]\!\rangle[\![(P_2 \parallel Q_2)_\xi]\!\rangle\phi}{\Gamma \vdash [\![(P_1; P_2 \parallel Q_1; Q_2)_\xi]\!\rangle\phi}$$

$\forall\exists$-MR
$$\frac{\Gamma \vdash [\![(P \parallel Q)_\xi]\!\rangle\phi \qquad \phi \vdash \psi}{\Gamma \vdash [\![(P \parallel Q)_\xi]\!\rangle\psi}$$

$\forall\exists$-;-L
$$\frac{\Gamma \vdash [P][\![(P_2 \parallel Q)_\xi]\!\rangle\phi}{\Gamma \vdash [\![(P_1; P_2 \parallel Q)_\xi]\!\rangle\phi}$$

$\forall\exists$-;-R
$$\frac{\Gamma \vdash [\![(P \parallel Q_1)_\xi]\!\rangle\langle\xi(Q_2)\rangle\phi}{\Gamma \vdash [\![(P \parallel Q_1; Q_2)_\xi]\!\rangle\phi}$$

$\forall\exists$-$\cup$-L
$$\frac{\Gamma \vdash [\![(P_1 \parallel Q)_\xi]\!\rangle\phi \wedge [\![(P_2 \parallel Q)_\xi]\!\rangle\phi}{\Gamma \vdash [\![(P_1 \cup P_2 \parallel Q)_\xi]\!\rangle\phi}$$

$\forall\exists$-$\cup$-R
$$\frac{\Gamma \vdash [\![(P \parallel Q_1)_\xi]\!\rangle\phi \vee [\![(P \parallel Q_2)_\xi]\!\rangle\phi}{\Gamma \vdash [\![(P \parallel Q_1 \cup Q_2)_\xi]\!\rangle\phi}$$

$\forall\exists$-$\wedge$
$$\frac{\Gamma \vdash [\![(P \parallel Q)_\xi]\!\rangle(\phi \wedge \psi)}{\Gamma \vdash [\![(P \parallel Q)_\xi]\!\rangle\phi \wedge [\![(P \parallel Q)_\xi]\!\rangle\psi}$$

$\forall\exists$-$\vee$
$$\frac{\Gamma \vdash [\![(P \parallel Q)_\xi]\!\rangle\phi \vee [\![(P \parallel Q)_\xi]\!\rangle\psi}{\Gamma \vdash [\![(P \parallel Q)_\xi]\!\rangle(\phi \vee \psi)}$$

$\forall\exists$-V
$$\frac{\phi \vdash [P]\phi_1 \qquad \psi \vdash \langle\xi(Q)\rangle\psi_1}{(\phi \wedge \psi) \vdash [\![(P \parallel Q)_\xi]\!\rangle(\phi_1 \wedge \psi_1)}$$ WHERE $(\text{VAR}(\phi) \cup \text{VAR}(\phi_1)) \cap \text{VAR}(\xi(Q)) = \emptyset$ AND $(\text{VAR}(\psi) \cup \text{VAR}(\psi_1)) \cap \text{VAR}(P) = \emptyset$

Fig. 6. General proof rules for $[\![(P \parallel Q)_\xi]\!\rangle\phi$

of dL. In particular, we develop a proof system (Section VI-A) for general forall-exists properties, and a collection of techniques to reason with the timing of dynamics (Section VI-B).

To reason with a forall-exists property, we develop a proof system for these properties, inspired by the technique of self-composition [14], [15]. We first rename variables used by the genuine program into a fresh set of variables, and then capture the relationship between the two programs using both the original and renamed variables. Using such an approach, $\text{ATT}(P) \sqsubseteq_{\phi_{pre},\mathcal{H},d} P$ can be encoded as a dL formula with mixed modalities, i.e., $[\_]\langle\_\rangle\phi$, which can be proven using proof rules and tools developed for dL.

First, we define renaming functions, as follows:

Definition 8 (Renaming)**.** For a program P and a set of variables V such that $V \cap \text{VAR}(P) = \emptyset$, a function $\xi : \text{VAR}(P) \to V$ is a *renaming function* for P if it is a bijection.

We write $\xi(P)$ for the program equivalent to P but whose variables have been renamed according to ξ. Renaming functions similarly apply to dL formulas and states.

Then, the following theorem captures untimed distances as dL formulas, by extending existing work on untimed distance on sensor attacks [21] to a general forall-exists setting:

Theorem 3 (dL encoding of simulation distance)**.** For a program P, an attack function ATT, formulas ψ, ϕ and ϕ_{pre}, a renaming function ξ, a variable set $\mathcal{H}$ and $d \in \mathbb{R}$, if we have

- $\psi \to [\text{ATT}(P)]\langle\xi(P)\rangle\phi$
- $\phi_{pre} \to \exists x_1, x_2 \cdots x_j$ such that $(\psi \wedge \xi(\phi_{pre}))$, where $\text{VAR}(\xi(\phi_{pre})) = \{x_1, x_2, \cdots x_j\}$
- $\phi \to \rho^\xi_{\mathcal{H}} \leq d$, for $\rho^\xi_{\mathcal{H}} = \sqrt{\sum_{x\in\mathcal{H}} (x - \xi(x))^2}$

then $\text{ATT}(P) \sqsubseteq_{\phi_{pre},\mathcal{H},d} P$.

The proof obligation is mainly on proving formulas of the form $\psi \to [\text{ATT}(P)]\langle\xi(P)\rangle\phi$. To help with that, we introduce a special notation, denoted $\forall\exists$-modality, to readily and concisely encode general forall-exists formulas. The $\forall\exists$-modality, written $[\![(P \parallel Q)_\xi]\!\rangle\phi$, is defined as

$$[\![(P \parallel Q)_\xi]\!\rangle\phi \equiv [P]\langle\xi(Q)\rangle\phi$$

where $\text{VAR}(P) \cap \text{VAR}(\xi(Q)) = \emptyset$. We will call the programs P and Q, respectively, the *left* and *right* programs. Formulas like $[\text{ATT}(P)]\langle\xi(P)\rangle\phi$ are now expressed as $[\![(\text{ATT}(P) \parallel P)_\xi]\!\rangle\phi$.

A. A Proof System for $\forall\exists$-modality

We have developed a set of proof rules, to reason with the $\forall\exists$-modality. Our proof system allows us to derive *sequents* [12] of the form $\Gamma \vdash \Delta$, where *antecedent* Γ and *succedent* Δ are finite sets of dL formulas. The semantics of $\Gamma \vdash \Delta$ is that of the dL formula $\bigwedge_{\phi\in\Gamma} \phi \to \bigvee_{\psi\in\Delta} \psi$.

General proof rules We first present a set of general proof rules, shown in Fig. 6, adapted from existing work on simulation distances [21] and relational program logics for *forall-forall* relational properties [22], [23], [24]. Most rules follow the semantics of the dL modalities, and their meanings should be straightforward to interpret.

Proof rules for loops For relational formulas that have loops such as $[\![(P^* \parallel Q^*)_\xi]\!\rangle$, the reasoning often proceeds in two possible ways: (1) lock-step that the two loops run in a synchronized manner, and (2) multi-step that the right loop runs multiple iterations for every iteration of the left loop. The following proof rules show the two cases:

$\forall\exists$-LOCK
$$\frac{\Gamma \vdash \phi_{inv} \qquad \phi_{inv} \vdash [\![(P \parallel Q)_\xi]\!\rangle\phi_{inv} \qquad \phi_{inv} \vdash \psi}{\Gamma \vdash [\![(P^* \parallel Q^*)_\xi]\!\rangle\psi}$$

$\forall\exists$-MULTI
$$\frac{\Gamma \vdash \phi_{inv} \qquad \phi_{inv} \vdash [\![(P \parallel Q^*)_\xi]\!\rangle\phi_{inv} \qquad \phi_{inv} \vdash \psi}{\Gamma \vdash [\![(P^* \parallel Q^*)_\xi]\!\rangle\psi}$$

Rule $\forall\exists$-MULTI is often quite useful in reasoning about the system's behavior after the attack starts. For example, for every control cycle of the compromised system P, the genuine system Q often needs multiple cycles to match the impact of the attack on P. Rule $\forall\exists$-MULTI is needed in such a scenario.

In addition, we have the following auxiliary rules to reason with the right program with loops. They can be derived from the definition of the $\forall\exists$-modality and the semantics of $\langle Q^*\rangle\phi$.

$\forall\exists$-LOOPN
$$\frac{\Gamma \vdash \exists n \in \mathbb{N}\ [\![(P \parallel Q^n)_\xi]\!\rangle\phi}{\Gamma \vdash [\![(P \parallel Q^*)_\xi]\!\rangle\phi}$$

$\forall\exists$-LOOP*
$$\frac{\Gamma \vdash [\![(P \parallel Q^*; Q^*)_\xi]\!\rangle\phi}{\Gamma \vdash [\![(P \parallel Q^*)_\xi]\!\rangle\phi}$$

Proof rules for dynamics The proof rules in Fig. 7 allows one to reason with two dynamics by composing them into a single program. Rule $\forall\exists$-ODE-$\forall$ reduces the reasoning with a $\forall\exists$-modality to reasoning with two modalities of necessity. Intuitively, the formula ϕ holds for all executions of $x' = \theta$

$$\forall\exists\text{-ODE-}\forall\quad \frac{\Gamma \vdash [x' = \theta][\xi(y' = \delta)](\psi(t_l, \xi(t_l)) \to \phi)}{\Gamma \vdash [\![(x' = \theta, t_l{}' = 1 \parallel y' = \delta, t_l{}' = 1)_\xi]\!\rangle\phi}$$

$$\forall\exists\text{-ODE-G}\quad \frac{\Gamma \vdash [x' = \theta,\ \xi(y' = \delta)\,\&(\phi_x \wedge \xi(\phi_y))]\phi \qquad \Gamma \vdash [x' = \theta,\ \xi(y' = \delta)](\phi_x \to \xi(\phi_y))}{\Gamma \vdash [\![(x' = \theta\,\&\phi_x \parallel y' = \delta\,\&\phi_y)_\xi]\!\rangle\phi}$$

$$\forall\exists\text{-ODE-I}\quad \frac{\Gamma \vdash [x' = \theta,\ \xi(y' = \delta)]\phi}{\Gamma \vdash [\![(x' = \theta, t_l{}' = 1 \parallel y' = \delta, t_l{}' = 1)_\xi]\!\rangle\phi}$$

$$\forall\exists\text{-ODE-C}\quad \frac{\epsilon_x \le \epsilon_y \qquad \Gamma \vdash [x' = \theta,\ \xi(y' = \delta)\,\&(\phi_x \wedge \xi(\phi_y))]\phi \qquad \Gamma \vdash [x' = \theta,\ \xi(y' = \delta)](\phi_x \to \xi(\phi_y))}{\Gamma, t_l = 0, \xi(t_l = 0) \vdash [\![(x' = \theta, t_l{}' = 1\,\&(\phi_x \wedge t_l \le \epsilon_x) \parallel y' = \delta, t_l{}' = 1\,\&(\phi_y \wedge t_l \le \epsilon_y))_\xi]\!\rangle\phi}$$

Fig. 7. Proof rules for the modality $[\![(P \parallel Q)_\xi]\!\rangle\phi$ on ODEs

and $\xi(y' = \delta)$ if a timing condition ψ holds. Note that formula ψ only refers to variables t_l and $\xi(t_l)$: it captures a relationship between the two local clocks. Here t_l is neither in x nor in y. Therefore, there always exists a value of $\xi(t_l)$ that can make $\psi(t_l, \xi(t_l))$ true. Building from this rule, Rule $\forall\exists$-ODE-I obtains a composition of dynamics by merging the two modalities of necessity in $\forall\exists$-ODE-$\forall$ into a single modality of necessity on the dynamics $x' = \theta$ and $\xi(y' = \delta)$. With this merge, all solutions of $x' = \theta$ and $\xi(y' = \delta)$ evolve for the same amount of time. Formulas like $[x' = \theta,\ \xi(y' = \delta)]\phi$, is often easier to verify.

Rule $\forall\exists$-ODE-C is designed to reason with two dynamics with evolution constraints. Note that the constraints ϕ_x and ϕ_y are time-invariant (i.e., they don't concern timing related variables like t_g and t_l). The third premise is the key that, for any execution of the composition, the evolution constraint for the right dynamics ϕ_y holds, so long as the left evolution constraint ϕ_x holds. That means, intuitively, the evolution of the right dynamics would not restrict the trajectory of the left dynamics, which captures the "forall executions of the left" requirement of the $\forall\exists$-modality. The condition on clocks is straightforward: the maximum interval for the right dynamics shall be no less than the maximum interval of the left one, so the right dynamics always have enough time for evolution. Rule $\forall\exists$-ODE-G is a generalization of Rule $\forall\exists$-ODE-C, where the constraint ϕ_x and ϕ_y may involve timing.

The proof system introduced in this section is a sound proof technique to derive generic forall-exist properties.

Theorem 4 (*Soundness of the proof system*)**.** If a sequent $\Gamma \vdash \Delta$ can be derived through the proof system then the formula $\bigwedge_{\phi\in\Gamma}\phi \to \bigvee_{\psi\in\Delta}\psi$ holds.

Detailed proof can be found in the Appendix (Section C).

B. Reasoning with Timing of Dynamics

A major difficulty in reasoning with $\forall\exists$-modality in the setting of timed attacks, is how to deal with the timing of dynamics. It comes with at least two main challenges: (1) how to express the accumulated impact of the attack on a system's dynamics, as the attack often lasts for a duration in timed attacks, and (2) how to properly align the right dynamics, e.g., its execution time, with the left so we can prove properties on a composition of the two dynamics. We elaborate on our solutions to both challenges.

Reasoning with the accumulated impact of dynamics The first main challenge is how to reason about the impact of the dynamics *accumulated* in multiple cycles of evolution. For example, under a scheduled DoS attack, the left dynamics would run for at least the duration of the attack under an incorrect control action, while the right dynamics may switch control actions back and forth and stay stable around one equilibrium. Reasoning about the relationship between the two dynamics requires first expressing the continuous impact of the attack on the left dynamics during multiple cycles.

To help express the accumulated impact of dynamics of multiple cycles, we introduce a *timed structure for dynamics* for a program modeling the dynamics: $plant \equiv (x' = \theta, t_g{}' = 1)\,\&\phi$. The $\mathtt{T}$ duration program construct of $plant$, denoted $plant^{\langle\mathtt{T}\rangle}$, concerns, intuitively, for a starting state ω, all reachable states of the program in the duration $[\mathtt{T}_1, \mathtt{T}_1 + \mathtt{T}]$, where $\mathtt{T}_1$ is the current time of ω. Formally, its semantics is the following:

$$[\![plant^{\langle\mathtt{T}\rangle}]\!] = \{(\omega, \nu) \mid (\omega, \nu) \in [\![plant]\!] \text{ and } \nu(t_g) - \omega(t_g) \le \mathtt{T}\}$$

The following proposition holds naturally:

Proposition 3. For a formula ψ and a timed program $plant \equiv (x' = \theta, t_g{}' = 1)\,\&\phi$, it holds that

$$[t_o := t_g\,; (x' = \theta, t_g{}' = 1)\,\&(\phi \wedge t_g \le t_o + \mathtt{T})]\psi \to [plant^{\langle\mathtt{T}\rangle}]\psi$$

where t_o is a fresh variable and ϕ doesn't involve t_o.

To express and reason about accumulated impact, we are often more interested in a continuous sequence of evolution with the same control action. For example, the DoS attack forces the cooling system to deploy the same control $delta := 1$ for multiple cycles. These actions are *permanent* control actions [25]. Intuitively, an action is permanent if executing it more than once in a row has no consequence for the system dynamics. This is true in the common case of actions that only assign constant values to control variables that are read but not modified by the plant, such as $delta := 1$ and $delta := -0.5$ in the examples.

The following proposition connects the timed program to $\mathsf{d}\mathcal{L}$ programs with timing encoded (and thus Theorem 2). With this proposition, we reduce the reasoning of the accumulated impact of an attack in multiple cycles, i.e., $(?\mathtt{T}_1 \le t_g \le \mathtt{T}_2\,; t_l := 0; action; plant)^*$ to the reasoning of a timed program $action; plant_s{}^{\langle\mathtt{T}_2 - \mathtt{T}_1 + \epsilon\rangle}$:

Proposition 4. For a control action $action$, a program $plant \equiv (x' = \theta, t_g{}' = 1, t_l{}' = 1)\,\&(\phi \wedge t_l \le \epsilon)$, a formula ψ, and two time points $\mathtt{T}_1$, $\mathtt{T}_2$, if (i) $action$ is permanent and doesn't modify ϕ, and (ii) ϕ and ψ are time-invariant, then:

$$[action; plant_s{}^{\langle\mathtt{T}_2 - \mathtt{T}_1 + \epsilon\rangle}]\psi \to [(?\mathtt{T}_1 \le t_g \le \mathtt{T}_2\,; t_l := 0; action; plant)^*]\psi$$

where $plant_s \equiv (x' = \theta, t_g{}' = 1)\,\&\phi$.

$$(\textit{System Constants}: \epsilon = 1 \land x_L = 100 \land r = 3)$$
$$\phi_{pre} \equiv x_p = 36$$
$$\phi_{post} \equiv 29.5 \leq x_p \leq 42.1$$
$$inc \equiv ?x_s < 35; sc := 1$$
$$dec \equiv ?x_s \geq 35; sc := 1/4$$
$$ctrl \equiv t_l := 0\,; x_s := x_p\,; (inc \cup dec\,)$$
$$plant \equiv (x_p{}' = r(sc - \frac{x_p}{x_L}), t_g{}' = 1, t_l{}' = 1)\,\&(x_p \geq 0 \land t_l \leq \epsilon)$$

Fig. 8. dℒ model of a water tank with sensing

Intuitively, the proposition holds due to the flow property of autonomous differential equations [26], the evolution constraint being time-invariant, and the action is permanent. Detailed proof can be found in the Appendix (Section C).

Aligning timing of dynamics Another main challenge is how to reason with two dynamics running for different durations. For example, consider formulas of the form $[\![(x' = \theta, t_l{}' = 1\,\&t_l \leq \epsilon_x \quad \| \quad y' = \delta, t_l{}' = 1\,\&t_l \leq \epsilon_y)_\xi]\!]\phi$, where the left and right dynamics may have different control intervals, e.g., $\epsilon_x \geq \epsilon_y$. It is non-trivial to prove a property on these two dynamics.

We tackle this problem by properly *aligning* the execution time of right dynamics with the left so we can prove properties on a composition of the two dynamics, e.g., using rules introduced in Fig. 7. We achieve this alignment by *scaling* the time of the right dynamics, so that its state space stays the same but its duration after scaling aligns with the left's duration. After that, we can apply rules introduced in Fig. 7. We can achieve such a scaling using the technique of time re-parametrization [26] (or time stretching [27]) for autonomous differential equations (the way how dynamics are modeled in dℒ). Time re-parametrization changes the way time is measured or expressed along the trajectory of the system without altering the actual trajectory in the state space. In particular, the following proposition holds:

Proposition 5. For a program with dynamics $plant \equiv (x' = \theta, t_g{}' = 1, t_l{}' = 1)\,\&(\phi \land t_l \leq \epsilon)$, a constant $k > 0$, and a formula ψ, if the constraint ϕ and formula ψ are time-invariant, then the following property holds:

$$\langle (x' = k * \theta, t_l{}' = k)\,\&(\phi \land t_l \leq \epsilon)\rangle\psi \to \langle plant\rangle\psi\,.$$

Up to now, we have introduced a system of techniques to reason with untimed distance and timing aspects. Next, we showcase these techniques with a case study.

VII. Case Study: A Water Tank System

In this case study, we demonstrate how to use the introduced machinery to establish timed properties. We focus on the proof derivations for a scheduled DoS attack on sensor readings and discuss relevant timed properties. We then briefly describe how the example derivations can be adjusted for other attacks.

Consider an example of a water tank shown in Fig. 8, which is inspired by literature [28]. It mixes the salt and water inside the tank. Initially, it contains 36 lb of salt ($x_p = 36$) dissolved in 100 gal of water ($x_L = 100$). An inflow of water with a salt concentration rate sc (lb of salt/gal) is entering the tank at a rate of $r = 3$ gal/min. The well-stirred mixture is draining from the tank at the same rate r. The tank has two modes of control: a salt decreasing mode with sc set to $1/4$ if the measured salt level is high ($?x_s \geq 35$) and a salt increasing mode with sc set to 1 if the measured salt level is low ($?x_s < 35$). The rate of change of salt in the tank $x_p{}'$ is equal to the rate at which salt is flowing in minus the rate at which is flowing out: $x_p{}' = r(sc - x_p/x_L)$, where x_p/x_L computes the concentration of the salt. The postcondition we are interested is that the salt level stays within a certain level ($29.5 \leq x_p \leq 42.1$). The reachable salt level of the water tank is (approximately) $34.5 \leq x_p \leq 37.1$ and its quantitative safety is (at least) 5.

Scheduled Sensor DoS Attack Consider a scheduled DoS attack that disables the sensor for a duration of 2 minutes ($\text{DURATION} = 2$) starting at time 5 ($\text{START} = 5$). The attack can be modeled as:

$$ctrl_{comp} \equiv t_l := 0\,; \text{MALWARE}\,; (inc \cup dec\,)$$
$$\text{MALWARE} \equiv (?\text{TRIGGER}\,; ?\top) \cup (?\neg\text{TRIGGER}\,; x_s := x_p)$$
$$\text{TRIGGER} \equiv \text{START} \leq t_g \leq \text{START} + \text{DURATION}$$

Note that it's non-trivial to calculate the quantitative safety of the compromised system, especially, the duration when the attack is active. In addition, the solution to the differential equation is an exponential function that cannot be expressed in tools like KeYmaera X [29].

$$\phi^\xi_{pre} \equiv x_p = 36 \land x_{p_1} = 36 \land t_g = 0 \land t_{g_1} = 0$$
$$\phi^\xi_{post} \equiv \sqrt{(x_p - x_{p_1})^2} \leq 6$$
$$\xi \equiv \{x_p \mapsto x_{p_1}, x_s \mapsto x_{s_1}, sc \mapsto sc_1, t_l \mapsto t_{l_1}, t_g \mapsto t_{g_1}\}$$
$$P \equiv \text{ATT}_{dos}(ctrl; plant)$$
$$Q \equiv (ctrl; plant)$$
$$\phi_{t_1} \equiv 0 \leq t_g < \text{START}$$
$$\phi_{t_2} \equiv \text{START} \leq t_g \leq \text{START} + \text{DURATION}$$
$$\phi_{t_3} \equiv t_g > \text{START} + \text{DURATION}$$
$$\phi_1 \equiv x_p = x_{p_1} \land 34.5 < x_p < 37.1$$
$$\phi_2 \equiv (-1.1 \leq x_p - x_{p_1} \leq 6 \land 34.5 < x_{p_1} < 37.1)$$
$$\phi_3 \equiv (0 \leq x_p{}' - x_{p_1}{}' \leq 1.3) \land (t_g - t_o \leq \text{DURATION} + \epsilon) \land x_p - x_{p_1} \leq (2.1 + (t_g - t_o) * 1.3))$$
$$\phi_m \equiv \phi_{m_1} \lor \phi_{m_2} \lor \phi_{m_3} \lor \phi_{m_4}$$
$$\phi_{m_1} \equiv x_s < 35 \land x_p \geq 35 \land x_{p_1} < 35 \land 0 \leq x_p - x_{p_1} \leq 2.1$$
$$\phi_{m_2} \equiv x_s < 35 \land x_p < 35 \land x_{p_1} < 35 \land -0.5 \leq x_p - x_{p_1} \leq 0$$
$$\phi_{m_3} \equiv x_s \geq 35 \land x_p < 35 \land x_{p_1} = 35 \land -0.5 \leq x_p - x_{p_1} \leq 0$$
$$\phi_{m_3} \equiv x_s \geq 35 \land x_p \geq 35 \land x_{p_1} = 35 \land 0 \leq x_p - x_{p_1} \leq 2.1$$
$$plant_s \equiv (x_p{}' = r(sc - x_p/x_L), t_g{}' = 1)\,\&(x_p \geq 0)$$
$$\phi \equiv x_p \geq 0 \land t_l \leq \epsilon$$
$$plant_i \equiv (x_p{}' = r(1 - x_p/x_L), t_g{}' = 1, t_l{}' = 1)\,\&\phi$$
$$plant_k \equiv (x_p{}' = 1/3 * r(1 - x_p/x_L), t_l{}' = 1/3)\,\&\phi$$
$$\phi_{e_o} \equiv x_p \geq 0 \land t_g \leq t_o + \text{DURATION} + \epsilon$$
$$plant_o \equiv (x_p{}' = r(1 - x_p/x_L), t_g{}' = 1)\,\&\phi_{e_o}$$
$$plant^\infty_m \equiv x_p{}' = r(1 - x_p/x_L), t_g{}' = 1, x_{p_1}{}' = 1/3 * r(1 - x_{p_1}/x_L), t_{l_1}{}' = 1/3$$
$$plant_{merge} \equiv plant^\infty_m\,\&\,(\phi \land \xi(\phi_{e_o}))$$

Fig. 10. Definitions of formulas and programs used in the example proof

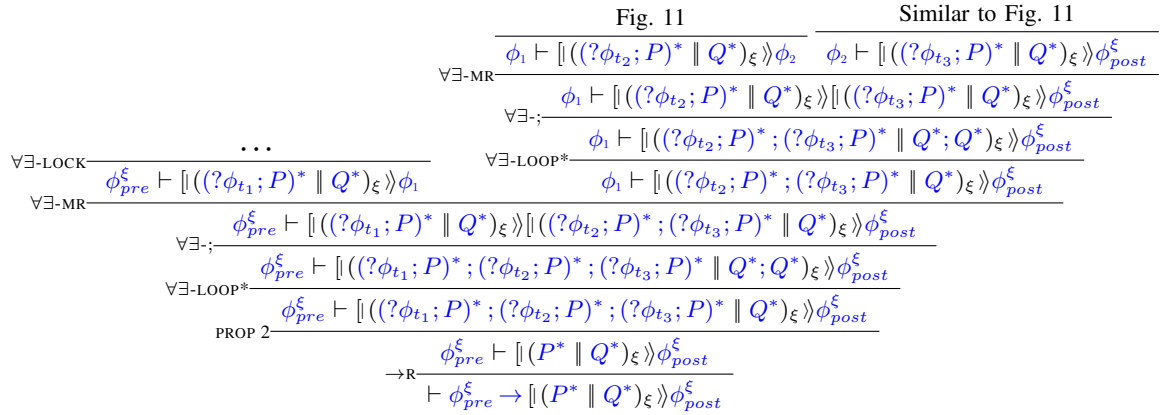

Fig. 9. Proof of a simulation distance for the water tank

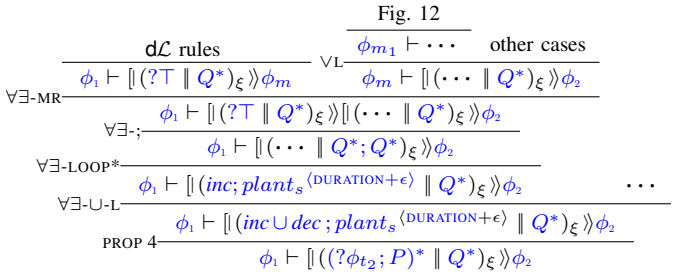

Fig. 11. Proof of the distance when the attack is active

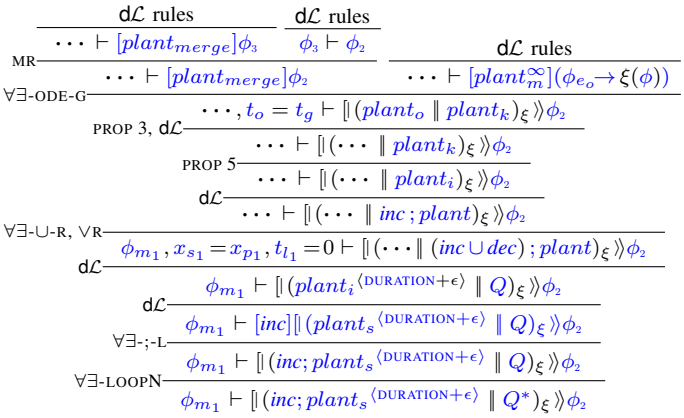

Fig. 12. Proof for the ϕ_{t_2} case (increasing mode)

We can use the introduced techniques and the original dℒ axioms and rules to prove that the compromised system and the genuine one are at simulation distance 6 with respect to ϕ_{pre} and $\mathcal{H} = \{x_p\}$. We show the proof derivation in Fig. 9, which refers to the definitions of formulas and programs shown in Fig. 10. The proof rules used are shown at the left of each derivation step (the notation dℒ in the derivation means applying the normal dℒ rules and axioms). In the proof derivation, we ignore certain parts using the notation $\cdots$, if the contents can be inferred from the context, e.g., succedent, or the derivation can be constructed similarly to the other ones.

The derivation first splits the top-level obligation using Proposition 2 into sub-goals corresponding to different time durations based on whether the attack is active. Then, the derivation focuses on the duration in which the attack is active (Fig. 11). Fig. 11 and its sub-derivation in Fig. 12 present the worst-case impact of the attack. Intuitively, they show the scenario that the DoS attack starts when the salt level is reaching the maximum under normal operation. After the attack starts, the salt level will keep increasing and deviate from the normal range the most. In particular, the derivation in Fig. 11 focuses on the *inc* mode and leads to four cases, one of which corresponds to the worst-case scenario, i.e., ϕ_{m_1} (Fig. 12). A key step in Fig. 11 is to collapse the impact of the attack from multiple control cycles $(?\phi_{t_2}; P)^*$ into one timed dynamic, i.e., $plant_s^{\langle \text{DURATION}+\epsilon \rangle}$, using Proposition 4. The derivation in Fig. 12 focuses on the impact on dynamics in the worst case scenario. A key step here is Proposition 5, where we scale the dynamics of the genuine system so one *inc* control cycle of the genuine system would align with multiple cycles of the program $plant_s$ with respect to time. After the scaling, we can reason with the property of two dynamics, using the proof rule ∀∃-ODE-G, by reasoning with their combination $plant_{merge}$. The two promises of the rule can be proven using dℒ proof rules.

Now, using the corollaries from Section V-B, we can establish the properties of timed robustness, impact, and tolerance from derivations like the one shown in Fig. 9, For example, from the derivation of $\phi^{\xi}_{pre} \vdash [\![((?\phi_{t_1}; P)^* \mid\mid Q^*)_\xi]\!]\phi_1$, we get untimed distance $(?\phi_{t_1}; P)^* \sqsubseteq_{\phi_{pre},\mathcal{H},0} Q^*$ as $\phi_1 \to \rho^{\xi}_{\mathcal{H}} = 0$. Then by Theorem 2, we get timed distance $(?0 \le t \le \text{T}; P)^* \sqsubseteq^{[0,\text{T}]}_{\phi_{pre},\mathcal{H},d} Q$ for $\text{T} = \text{START}$, and thus by Corollary 1, we get a lower bound of time robustness for the interval $[0, \text{START}]$: $\text{T-ROBUST}_1^{[0,\text{START}]}(\text{ATT}(P^*), \phi_{pre}, \phi_{post})$. Similarly, from the derivation of $\phi^{\xi}_{pre} \vdash [\![((?\phi_{t_1}; P)^*; (?\phi_{t_1}; P)^* \parallel Q^*)_\xi]\!]\phi_2$ and Corollary 2, we get an upper bound of time impact for the time interval $[0, \text{START} + \text{DURATION}]$: $\text{T-IMPACT}_\delta^{[0,\text{START}+\text{DURATION}]}(\text{ATT}(P^*), \phi_{pre}, \phi_{post})$ for $\delta \le (6-5)/5 = 0.2$. Furthermore, with the derivation for $\phi_2 \vdash [\![((?\phi_{t_3}; P)^* \parallel Q^*)_\xi]\!]\phi^{\xi}_{post}$ (omitted in Fig. 9), we can prove that the simulation distance drops from 6 to 5

under 6.7 minutes. That means, the system already returns to safety for the time interval $[0, \text{START} + \text{DURATION} + 6.7 = 13.7]$. Therefore, by Corollary 3, we get timed tolerance $\text{T-TOLERANT}_{u_2}^{[0,13.7]}(P^*, \phi_{pre}, \phi_{post}, 8.7)$ for some $u_2 \geq -1$.

Scheduled Bounded Sensor Attack We briefly discuss how to adjust the derivation to prove other timed attacks. Consider a bounded sensor attack that deviates the sensor reading x_s for up to 0.5 lb, and the attack lasts for a duration of 2 minutes ($\text{DURATION} = 2$). Similar to examples shown in Section III, we can replace the PLD with $x_s := *; ?x_p - 0.5 \leq x_s \leq x_p + 0.5$. We can prove a simulation distance of 0.5 for the compromised system and the genuine system, by constructing a derivation analogous to the example one shown in Fig. 9. The key steps are (1) choosing the appropriate programs and formulas in Fig. 10, and (2) constructing new derivations for the three cases, especially the ϕ_{t_2} case. In particular, the formula ϕ_2 should be $34.5 \leq x_{p_1} \leq 37.1$ and the derivation for $\phi_1 \vdash [\![((?\phi_{t_2}; P)^* \parallel Q^*)_\xi]\!]\phi_2$ is constructed using the multi-step invariant rule $\forall\exists$-MULTI with $\phi_{inv} = 0 \leq x_{p_1} - x_p \leq 0.5 \wedge 34.5 < x_{p_1} < 37.1$ into the obligation $\phi_{inv} \vdash [\![((?\phi_{t_2}; P) \parallel Q^*)_\xi]\!]\phi_{inv}$. Then the derivation proceeds by analysis of different cases of x_p. More details about this derivation can be found in the Appendix.

Scheduled Actuator Integrity Attack Consider such an attack that can double the sc output by the controller, e.g., insert a program $sc := sc * 2$ after the $ctrl$ program. Such an attack would change the behavior of the water tank drastically. In particular, under such an attack, both *inc* and *dec* modes *increase* the salt level in the water tank. The derivation shown in Fig. 11 can be similarly constructed, by considering the worst-case scenario consists of two phases: (1) the compromised system starts evolving around $x_p = 35$ within the *inc* mode with the maximum interval ϵ, and then (2) it continues increasing the salt level within the *dec* mode until the time $\text{START} + \text{DURATION} + \epsilon$.

VIII. Related Work

Robustness of safety of CPSs Robustness in CPSs can be intended in several ways, which are classified by Fränzle et al. [4] as follows: (i) input/output robustness; (ii) robustness with respect to system parameters; (iii) robustness in real-time system implementation; (iv) robustness due to unpredictable environment; (v) robustness to faults. The notion of robustness considered in this paper falls in category (iv), where the attacks are the source of environment's unpredictability. Other works study robustness properties for CPSs [5], [30], [8], [6]. Some of them focus on robustness against attacks [5], [30], even adopting quantitative reasonings [8], [6].

Our *language-based* notion of robustness shares similarities with some existing notions of robustness, such as invariance [31] and input-to-state stability [32]. These notions concern if a system stays in a safe region when small changes happen to initial conditions, while our robustness concerns if a system stays in a safe region when under attack.

Other language-based approaches to robustness can be found in [7], [9], [21]. Xiang et al. [7], propose the notion of *robustness of safety* for CPSs represented in dL, which consists of the ability of the system to stay safe in the presence of (unbound) sensor attacks. Chong et al. [9], [21], study a quantified version of the robustness of safety proposed in [7], to measure the *impact* of *bounded sensor attacks* targeting CPSs. Our paper extends and generalizes the results in [9], [21] to a timed setting, in the presence of timed attacks that can tamper with both *sensor measurements* and/or *actuator commands*. In particular, our proof system is more powerful than the one proposed in [21] as it allows one to reason on generic forall-exist relational properties in dL, and not only simulation distances. Moreover, with the techniques we develop in Section VI-B, our proof system provides novel ways to relationally reason with the timing of dynamics, which is often lacking in prior work.

Relational reasoning for dL One key contribution of the work is the framework for relational reasoning in the setting of dL. Various approaches have been proposed to analyze specific relational properties in dL. A primitive is introduced to express a refinement relation between two hybrid programs [33]. An expressive modal logic based on dL has been introduced to reason with nondeducibility [34]. Neither work provides tool support. Recent work by Xiang et al. develops a general extension to dynamic logics and the extension supports automated and semi-automated verification of certain relational properties [35]. Kolčák et al. introduce a relational extension of dL that uses the technique of time re-parametrization to reason about *forall-forall* on two dynamics without evolution constraint [27]. Our Proposition 5 extends their work to dynamics with evolution constraints. The proof system, especially the general proof rules, in our work is inspired by relational program logic [22], [23], [24], most of which focus on forall-forall relational properties and do not work on dL dynamics. Some recent work investigate the alignment problem within relational reasoning [36], [24], the results of which may be encoded as more proof rules, which we leave to future work.

IX. Conclusion

We have introduced a formal framework for quantitative analysis of the robustness of safety for CPSs under timed attacks. We have defined a few time related properties and an ad hoc notion of timed simulation distance to reason with these properties. The centerpiece of the reasoning is a sound proof system and techniques to reason with the timing of dynamics.

Our case study has shown promising results in using our relational reasoning approach for the quantitative analysis of non-trivial dynamics, while directly establishing quantitative safety for either the genuine or the compromised system is challenging. The full potential of the relational formulation remains to be further explored.

Limitations The proof system and relevant results are designed to promote verifying the forall-exists modality. One key result is Proposition 5 that scales the time of one system to match the time of the other. In certain systems, such a matching may not always exist and the result won't apply. Finding a forall-forall alignment with a stretching function may be impossible.

One potential solution and relevant future work is to integrate numerical solvers, e.g., dReal, to help identify that a matching execution exists.

Future work An immediate future work is to automate and/or optimize the forall-exists reasoning, by leveraging automated tools like SMT solvers. Another interesting future work is to enhance our quantitative analysis with stronger safety metrics, and explore alternative distance measures that may better support relational reasoning. We also plan to explore more sophisticated timed attacks, and in particular, periodic attacks, that can achieve their malicious goals ensuring better stealthiness.

ACKNOWLEDGMENT

Ruggero Lanotte and Simone Tini received funding from the European Union - Next-GenerationEU - National Recovery and Resilience Plan (NRRP) – MISSION 4 COMPONENT 2, INVESTMENT N. 1.1, CALL PRIN 2022 D.D. 104 02-02-2022 – MEDICA Project, CUP N. J53D23007180006. Massimo Merro has been partially supported by the SERICS project (PE00000014) under the *MUR National Recovery and Resilience Plan*, funded by the EU - NextGenerationEU.

REFERENCES

[1] J. Giraldo, D. I. Urbina, A. Cardenas, J. Valente, M. Faisal, J. Ruths, N. O. Tippenhauer, H. Sandberg, R. Candell, A Survey of Physics-Based Attack Detection in Cyber-Physical Systems, ACM Comput. Surv. 51 (4) (2018) 76:1–76:36.

[2] R. Lanotte, M. Merro, A. Munteanu, L. Viganò, A Formal Approach to Physics-based Attacks in Cyber-physical Systems, ACM Transactions on Privacy and Security 23 (1) (2020) 3:1–3:41.

[3] F. Arnold, H. Hermanns, R. Pulungan, M. Stoelinga, Time-Dependent Analysis of Attacks, in: Principles of Security and Trust POST, Vol. 8414 of LNCS, Springer, 2014, pp. 285–305.

[4] M. Fränzle, J. Kapinski, P. Prabhakar, Robustness in cyber-physical systems, Dagstuhl Reports 6 (9) (2016) 29–45.

[5] F. Hu, Y. Lu, A. V. Vasilakos, Q. Hao, R. Ma, Y. Patil, T. Zhang, J. Lu, X. Li, N. N. Xiong, Robust cyber-physical systems: concept, models, and implementation, Future Gener. Comput. Syst. 56 (2016) 449–475.

[6] M. Rungger, P. Tabuada, A notion of robustness for cyber-physical systems, IEEE Trans. Autom. Control. 61 (8) (2016) 2108–2123.

[7] J. Xiang, N. Fulton, S. Chong, Relational analysis of sensor attacks on Cyber-Physical Systems, in: CSF, IEEE, 2021, pp. 1–16.

[8] P. Tabuada, S. Y. Caliskan, M. Rungger, R. Majumdar, Towards robustness for cyber-physical systems, IEEE Trans. Autom. Control. 59 (12) (2014) 3151–3163.

[9] S. Chong, R. Lanotte, M. Merro, S. Tini, J. Xiang, Quantitative robustness analysis of sensor attacks on cyber-physical systems, in: HSCC, ACM, 2023, pp. 20:1–20:12.

[10] A. Giacalone, C. Jou, S. A. Smolka, Algebraic reasoning for probabilistic concurrent systems, in: M. Broy, C. B. Jones (Eds.), Programming concepts and methods: Proceedings of the IFIP Working Group 2.2, 2.3 Working Conference on Programming Concepts and Methods, Sea of Galilee, Israel, North-Holland, 1990, pp. 443–458.

[11] A. Platzer, Differential dynamic logic for hybrid systems, Journal of Automated Reasoning 41 (2) (2008) 143–189.

[12] A. Platzer, Logical foundations of cyber-physical systems, Vol. 662, Springer, 2018.

[13] A. Platzer, A complete uniform substitution calculus for differential dynamic logic, Journal of Automated Reasoning 59 (2) (2017) 219–265.

[14] G. Barthe, P. R. D'Argenio, T. Rezk, Secure information flow by self-composition, in: CSF, 2004, pp. 100–114.

[15] T. Terauchi, A. Aiken, Secure information flow as a safety problem, in: SAS, 2005, pp. 352–367.

[16] D. Kozen, Kleene algebra with tests, TOPLAS 19 (3) (1997) 427–443.

[17] G. E. Fainekos, G. J. Pappas, Robustness of temporal logic specifications for continuous-time signals, Theoretical Computer Science 410 (42) (2009) 4262–4291.

[18] S. Boyd, S. P. Boyd, L. Vandenberghe, Convex optimization, Cambridge university press, 2004.

[19] M. Krotofil, A. A. Cárdenas, Resilience of Process Control Systems to Cyber-Physical Attacks, in: NordSec, Vol. 8208 of LNCS, Springer, 2013, pp. 166–182.

[20] M. Krotofil, A. A. Cárdenas, J. Larsen, D. Gollmann, Vulnerabilities of cyber-physical systems to stale data – Determining the optimal time to launch attacks, International Journal of Critical Infrastructure Protection 7 (4) (2014) 213–232.

[21] J. Xiang, R. Lanotte, S. Tini, S. Chong, M. Merro, Measuring robustness in cyber-physical systems under sensor attacks, Nonlinear Analysis: Hybrid Systems 56 (2025) 101559.

[22] N. Benton, Simple relational correctness proofs for static analyses and program transformations, in: POPL, 2004, pp. 14–25.

[23] H. Yang, Relational separation logic, Theoretical Computer Science 375 (1-3) (2007) 308–334.

[24] R. Nagasamudram, D. A. Naumann, Alignment completeness for relational hoare logics, in: LICS, 2021, pp. 1–13.

[25] A. Kabra, J. Laurent, S. Mitsch, A. Platzer, Cesar: Control envelope synthesis via angelic refinements, in: TACAS, 2024, pp. 144–164.

[26] C. Chicone, Ordinary differential equations with applications, Vol. 34, Springer Science & Business Media, 2006.

[27] J. Kolčák, J. Dubut, I. Hasuo, S.-y. Katsumata, D. Sprunger, A. Yamada, Relational differential dynamic logic, in: TACAS, 2020, pp. 191–208.

[28] W. E. Boyce, R. C. DiPrima, Elementary differential equations and boundary value problems, Wiley, 2012.

[29] N. Fulton, S. Mitsch, J.-D. Quesel, M. Völp, A. Platzer, KeYmaera X: An axiomatic tactical theorem prover for hybrid systems, in: CADE, Vol. 9195 of LNCS, Springer, 2015, pp. 527–538.

[30] K. Huang, C. Zhou, Y. Tian, S. Yang, Y. Qin, Assessing the Physical Impact of Cyberattacks on Industrial Cyber-Physical Systems, IEEE Trans. Industrial Electronics 65 (10) (2018) 8153–8162.

[31] A. D. Ames, X. Xu, J. W. Grizzle, P. Tabuada, Control barrier function based quadratic programs for safety critical systems, IEEE Transactions on Automatic Control 62 (8) (2016) 3861–3876.

[32] A. A. Agrachev, A. S. Morse, E. D. Sontag, H. J. Sussmann, V. I. Utkin, E. D. Sontag, Input to state stability: Basic concepts and results, Nonlinear and optimal control theory: lectures given at the CIME summer school held in Cetraro, Italy June 19–29, 2004 (2008) 163–220.

[33] S. M. Loos, A. Platzer, Differential refinement logic, in: LICS, 2016, pp. 505–514.

[34] B. Bohrer, A. Platzer, A hybrid, dynamic logic for hybrid-dynamic information flow, in: LICS, 2018, pp. 115–124.

[35] J. Xiang, S. Chong, Extending dynamic logics with first-class relational reasoning, in: NASA formal method symposium, 2025.

[36] R. Nagasamudram, A. Banerjee, D. A. Naumann, Alignment complete relational hoare logics for some and all, arXiv preprint arXiv:2307.10045 (2023).

APPENDIX

A. Technical results and further examples from Section IV

The following two propositions hold:

Proposition 6. $[\![\phi\langle P\rangle_{[\mathtt{T}_1,\mathtt{T}_2]}]\!]\cup[\![\phi\langle P\rangle_{[\mathtt{T}_2,\mathtt{T}_3]}]\!] = [\![\phi\langle P\rangle_{[\mathtt{T}_1,\mathtt{T}_3]}]\!]$ for $\mathtt{T}_1 \leq \mathtt{T}_2 \leq \mathtt{T}_3$.

Proposition 7. $[\![\phi\langle P\rangle_{[\mathtt{T}_1,\mathtt{T}_2]}]\!] \subseteq [\![\phi\langle P\rangle_{[\mathtt{T}_3,\mathtt{T}_4]}]\!]$ if $\mathtt{T}_3 \leq \mathtt{T}_1 \leq \mathtt{T}_2 \leq \mathtt{T}_4$.

Our notion of safety obviously depends on the size on the safety interval.

Proposition 8. If $\text{T-SAFE}_{u_1}^{[\mathtt{T}_1,\mathtt{T}_2]}(P,\phi_{pre},\phi_{post})$ and also $\text{T-SAFE}_{u_2}^{[\mathtt{T}_3,\mathtt{T}_4]}(P,\phi_{pre},\phi_{post})$, for $[\mathtt{T}_1,\mathtt{T}_2]\subseteq[\mathtt{T}_3,\mathtt{T}_4]$, then $u_2\leq u_1$.

Furthermore, as expected, Definition 3 allows one to join safety intervals taking the minimum value of safety.

Proposition 9. If $\text{T-SAFE}^{[\mathtt{T}_1,\mathtt{T}_2]}_{u_1}(P,\phi_{pre},\phi_{post})$ and also $\text{T-SAFE}^{[\mathtt{T}_2,\mathtt{T}_3]}_{u_2}(P,\phi_{pre},\phi_{post})$ then, for $u = min(u_1,u_2)$, we have $\text{T-SAFE}^{[\mathtt{T}_1,\mathtt{T}_3]}_{u}(P,\phi_{pre},\phi_{post})$.

Here, we provide further examples showing how we can used our timed notions of safety, robustness and tolerance.

Example 2 (Scheduled DoS attack on sensor reading)**.** Let us focus on the attack described in Fig. 3 when the malware is instantiated with the first payload, PLD_{DoS}, to implement a DoS attack on the reading of the measurements. We recall that in the genuine system P^* the temperature oscillates in the interval $(99.5, 101]$. In this example, we will reason in the worst-case scenario, i.e., we assume that the attack starts when the last temperature sensed by the controller was 100 degrees, while the real temperature has increased to 101 degrees. This scenario provides the maximum impact on the safety and the robustness of the system under attack.

- Let us assume $\text{DURATION} = 2$. This means that the malware is triggered until $t_g = \text{START} + 2$ (this is the last moment when the attack may have a go). In this case, in the worst case scenario the plant takes 1 minute, and at time $\text{START} + 3$ the temperature may reach 104 degrees because during the attack the controller will keep sensing 100 degrees and it will not activate the cooling system. When the attack is over, three minutes after the START, the controller will sense the real (high) temperature and it will activate the cooling system. The robustness of the system under attack will be 1, at any time $\mathtt{T}$ with $\mathtt{T} < \text{START}$, and it will drop to $\frac{1}{4}$ at any time $\mathtt{T} \geq \text{START} + \text{DURATION} + 1$. The impact of the attack on the system is 0 at any time $\mathtt{T}$. More formally,
  - $\text{T-ROBUST}^{\mathtt{T}}_{1}(\text{ATT}(P^*),\phi_{pre},\phi_{post})$, for any point in time $\mathtt{T}$, with $0 < \mathtt{T} < \text{START}$.
  - $\text{T-ROBUST}^{\mathtt{T}}_{\frac{1}{4}}(\text{ATT}(P^*),\phi_{pre},\phi_{post})$, for any point in time $\mathtt{T}$, with $\mathtt{T} \geq \text{START} + \text{DURATION} + 1$.
  - For any $\mathtt{T} > 0$, $\text{T-IMPACT}^{\mathtt{T}}_{0}(\text{ATT}(P^*),\phi_{pre},\phi_{post})$.

  Since the impact is zero and the robustness remains positive, this means that the attack is tolerated by the system without causing violations of the safety.
- Let us assume $\text{DURATION} = 6$. In this case, the temperature of the engine under attack may reach 105 degrees after 4 minutes, and 108 degrees, after 7 minutes. The robustness of the system under attack will be 1 at any time $\mathtt{T}$, with $\mathtt{T} < \text{START}$, but it will drop to $-\frac{3}{4}$ at any time $\mathtt{T} \geq \text{START} + \text{DURATION} + 1$. In this case the impact of the attack on the system will be $\frac{3}{4}$. Notice that, since the cooling system is able to cool down the system by 0.5 degrees per minute, when the attack will terminate, the system will need 6 minutes to cool down at 105 degrees towards safety. More formally,
  - $\text{T-ROBUST}^{\mathtt{T}}_{1}(\text{ATT}(P^*),\phi_{pre},\phi_{post})$, for any point in time $\mathtt{T}$, with $0 < \mathtt{T} < \text{START}$.
  - $\text{T-ROBUST}^{\mathtt{T}}_{-\frac{3}{4}}(\text{ATT}(P^*),\phi_{pre},\phi_{post})$, for any point in time $\mathtt{T}$, with $\mathtt{T} \geq \text{START} + \text{DURATION} + 1$.
  - $\text{T-IMPACT}^{\mathtt{T}}_{\frac{3}{4}}(\text{ATT}(P^*),\phi_{pre},\phi_{post})$, for any point in time $\mathtt{T}$, with $\mathtt{T} \geq \text{START} + \text{DURATION} + 1$.
  - $\text{T-TOLERANT}^{[0,\mathtt{T}]}_{-3}(\text{ATT}(P),\phi_{pre},\phi_{post},\text{TH})$, where $\text{TH} = \text{START}+\text{DURATION}+1+6-(\text{START}+4) = 9$, for any point in time $\mathtt{T}$, with $\mathtt{T} > \text{START} + \text{DURATION} + 7$.

  The last statement says that the attack drags the system into unsafe states for at most 9 time instants during the time interval $[\text{START}+4, \text{START}+\text{DURATION}+1+6]$, with an (un)safety -3. However, after time $\text{START}+\text{DURATION}+7$ the system under attack recovers safety.

B. Proofs of results in Section V

We start with the proof of Theorem 1.

Proof of Theorem 1. We need to prove

$$\text{T-SAFE}^{[\mathtt{T}_l,\mathtt{T}_u]}_{u_1}(\text{ATT}(P),\phi_{pre},\phi_{post}) \text{ for } u_1 \geq u - d$$

where, by definition of timed safety, we know that $\text{T-SAFE}^{[\mathtt{T}_l,\mathtt{T}_u]}_{u_1}(\text{ATT}(P),\phi_{pre},\phi_{post})$ is equivalent to

$$u_1 = \mathbf{inf}\{\mathbf{Dist}_{\mathcal{H}}(\nu,[\![\phi_{post}]\!]) \mid \nu \in [\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!]\}$$

for $\mathcal{H} \equiv \text{VAR}(\phi_{post})$. Therefore, the proof obligation is

$$\mathbf{inf}\{\mathbf{Dist}_{\mathcal{H}}(\nu,[\![\phi_{post}]\!]) | \nu \in [\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!]\} \geq u - d \tag{1}$$

Consider any state $\nu \in [\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!]$. The hypothesis $\text{ATT}(P) \sqsubseteq^{[\mathtt{T}_l,\mathtt{T}_u]}_{\phi_{pre},\mathcal{H},d} P$, with $\mathcal{H} = \text{VAR}(\phi_{post})$, ensures that there is some state $\nu' \in [\![\phi_{pre}\langle P\rangle]\!]$ with $\rho_{\mathcal{H}}(\nu,\nu') \leq d$. We can show that $\mathbf{Dist}_{\mathcal{H}}(\nu',[\![\phi_{post}]\!]) \geq u$. Indeed, the hypothesis $\text{T-SAFE}^{[0,\infty]}_{u}(P,\phi_{pre},\phi_{post})$ coincides, by definition, with property

$$u = \mathbf{inf}\{\mathbf{Dist}_{\mathcal{H}}(\nu,[\![\phi_{post}]\!]) \mid \nu \in [\![\phi_{pre}\langle P\rangle]\!]\}$$

from which we infer $\mathbf{Dist}_{\mathcal{H}}(\nu',[\![\phi_{post}]\!]) \geq u$ since $\nu' \in [\![\phi_{pre}\langle P\rangle]\!]$.

We distinguish the cases $[\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!] \subseteq [\![\phi_{post}]\!]$ and $[\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!] \not\subseteq [\![\phi_{post}]\!]$.

We start with case $[\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!] \subseteq [\![\phi_{post}]\!]$.

Since $\rho_{\mathcal{H}}(\_,\_)$ is a metric, it is symmetric, thus implying $\rho_{\mathcal{H}}(\nu,\nu') = \rho_{\mathcal{H}}(\nu',\nu)$, and satisfies the triangular property. By the triangular property we infer that for any state $\nu'' \notin [\![\phi_{post}]\!]$, it holds

$$\rho_{\mathcal{H}}(\nu',\nu'') \leq \rho_{\mathcal{H}}(\nu',\nu) + \rho_{\mathcal{H}}(\nu,\nu'') \tag{2}$$

By definition of $\mathbf{Dist}_{\mathcal{H}}(\nu',[\![\phi_{post}]\!])$ and the properties $\mathbf{Dist}_{\mathcal{H}}(\nu',[\![\phi_{post}]\!]) \geq u$ and $\nu'' \notin [\![\phi_{post}]\!]$ we infer $\rho_{\mathcal{H}}(\nu',\nu'') \geq u$. From this inequality, property $\rho_{\mathcal{H}}(\nu,\nu') \leq d$ and Equation 2, we infer $\rho_{\mathcal{H}}(\nu,\nu'') \geq u - d$. By definition of $\mathbf{Dist}_{\mathcal{H}}(\nu,[\![\phi_{post}]\!])$ and the arbitrariness of $\nu'' \in \text{STA} \setminus [\![\phi_{post}]\!]$ we derive $\mathbf{Dist}_{\mathcal{H}}(\nu,[\![\phi_{post}]\!]) \geq u-d$. Then, by the arbitrariness of ν in $[\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!]$, we infer

$$\mathbf{inf}\{\mathbf{Dist}_{\mathcal{H}}(\nu,[\![\phi_{post}]\!]) \mid \nu \in [\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!]\} \geq u - d$$

which coincides with the proof obligation Equation 1. This completes the proof for case $[\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!] \subseteq [\![\phi_{post}]\!]$.

Consider now the case $[\![\phi_{pre}\langle \text{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!] \not\subseteq [\![\phi_{post}]\!]$.

Since $\mathbf{Dist}_{\mathcal{H}}(\nu',[\![\phi_{post}]\!]) \geq u$, we infer $\mathbf{B}_{\mathcal{H}}(\nu',u) \subseteq [\![\phi_{post}]\!]$. Consider $\mathbf{B}_{\mathcal{H}}(\nu,d-u+\epsilon)$, for an arbitray $\epsilon > 0$.

From $\rho_{\mathcal{H}}(\nu,\nu') \leq d$ we derive $\mathbf{B}_{\mathcal{H}}(\nu',u) \cap \mathbf{B}_{\mathcal{H}}(\nu, d-u+\epsilon) \neq \emptyset$. From this property and $\mathbf{B}_{\mathcal{H}}(\nu',u) \subseteq [\![\phi_{post}]\!]$ we infer $\mathbf{B}_{\mathcal{H}}(\nu, d-u+\epsilon) \cap [\![\phi_{post}]\!] \neq \emptyset$. By the arbitrarity of ϵ, we get $\mathbf{dist}_{\mathcal{H}}(\nu, [\![\phi_{post}]\!]) = \mathbf{inf}\{\rho_{\mathcal{H}}(\nu,\nu'') \mid \nu'' \in [\![\phi_{post}]\!]\} \leq d-u$. Namely, $\mathbf{Dist}_{\mathcal{H}}(\nu, [\![\phi_{post}]\!]) \geq u-d$. By the arbitrariness of ν in $[\![\phi_{pre}\langle \mathtt{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!]$, we infer

$$\mathbf{inf}\{\mathbf{Dist}_{\mathcal{H}}(\nu, [\![\phi_{post}]\!]) \mid \nu \in [\![\phi_{pre}\langle \mathtt{ATT}(P)\rangle_{[\mathtt{T}_l,\mathtt{T}_u]}]\!]\} \geq u-d$$

which coincides with the proof obligation Equation 1. This completes the proof. □

Proof of Corollary 3.. By definition of timed tolerance, the thesis $\text{T-TOLERANT}_{u_2}^{(\mathtt{T},\mathtt{T}_3)}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post}, \mathtt{TH})$ follows by

- $\text{T-SAFE}_{u_1}^{[\mathtt{T},\mathtt{T}_1]}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post})$, with $u_1 > 0$
- $\text{T-SAFE}_{u_2}^{[\mathtt{T}_1,\mathtt{T}_2]}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post})$, with $u_2 \geq u-d$
- $\text{T-SAFE}_{u_3}^{[\mathtt{T}_2,\mathtt{T}_3]}(\mathtt{ATT}(P), \phi_{pre}, \phi_{post})$, with $u_3 > 0$
- $\mathtt{T}_2 - \mathtt{T}_1 \leq \mathtt{TH}$.

The first of these statements follows by the hypothesis $\text{T-SAFE}_u^{[0,\infty]}(P, \phi_{pre}, \phi_{post})$, $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T},\mathtt{T}_1]}_{\phi_{pre},\mathcal{H},d_1} P$ and $u-d_1 > 0$, by applying Theorem 1 and choosing some $u_1 \geq u-d_1$. Then, the second stetement follows by the hypothesis $\text{T-SAFE}_u^{[0,\infty]}(P, \phi_{pre}, \phi_{post})$, and $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_1,\mathtt{T}_2]}_{\phi_{pre},\mathcal{H},d_2} P$, by applying Theorem 1 and choosing $u_2 \geq u-d_2$.
Finally, the third statement follows by the hypothesis $\text{T-SAFE}_u^{[0,\infty]}(P, \phi_{pre}, \phi_{post})$, $\mathtt{ATT}(P) \sqsubseteq^{[\mathtt{T}_2,\mathtt{T}_3]}_{\phi_{pre},\mathcal{H},d_3} P$ and $u-d_3 > 0$ by applying Theorem 1 and choosing $u_3 \geq u-d_3$.
Finally, the last item coincides with an hypothesis. □

Proof of Theorem 2. According to the definition of timed semantics, for any $\nu \in [\![\phi_{pre}\langle P^*\rangle_{[0,\mathtt{T}]}]\!]$, there is a state $\omega \in [\![\phi_{pre}]\!]$ such that $(\omega,\nu) \in [\![P^*]\!]$ and $0 \leq \nu(t_g) \leq \mathtt{T}$. By the nature of the global clock t_g, we know $0 \leq \omega(t_g) \leq \mathtt{T}$. That means $(\omega,\nu) \in [\![(?0 \leq t_g \leq \mathtt{T}; P)^*]\!]$, which concludes the proof. □

C. Proofs of results in Section VI

Proof of Theorem 4. We present the proofs as follows:

- The soundness proofs for rules in Fig. 6 can be given by adapting the arguments in the proof of Proposition 5 in [21].
- For the rules for loops, we can prove the soundness of both rules by induction on the iteration number of program P^*.
- The rule ∀∃-loopN and rule ∀∃-loop* are sound as both can be derived by unfolding the definition of ∀∃-modality, and existing $\mathsf{d}\mathcal{L}$ axioms and rules for Q^* and $\langle\_\rangle\_$.
- The rule ∀∃-ODE-∀ is sound because, given $[x'=\theta](\xi(x'=\theta))(\psi(t_l, \xi(t_l)) \to \phi)$, we know that for all solutions of $x'=\theta$ and $\xi(x'=\theta)$, their reachable states satisfy ϕ when the condition ψ holds. And since ψ only concerns time, and we have no constraints on the time in the model of dynamics, there must exists states where ψ holds.
- The rule ∀∃-ODE-I is sound because it is a specialized variant of the rule ∀∃-ODE-∀. A box modality $[x'=\theta,\ \xi(x'=\theta)]\phi$ forces the two dynamics evolve for the same duration. Thus, if $[x'=\theta,\ \xi(x'=\theta)]\phi$ holds, then for any reachable state of $x'=\theta$, a solution of $\xi(x'=\theta)$ can reach a state so ϕ holds, by evolving for the same duration as $x'=\theta$.
- For the rule ∀∃-ODE-G, if both promises hold, then that means it holds that $\Gamma \vdash [x'=\theta,\ \xi(y'=\delta)\,\&\phi_x]\phi$, since $\xi(y'=\delta)$ doesn't affect $x'=\theta$, then (follow the same reasoning for the rule ∀∃-ODE-I) for any reachable state of $x'=\theta\,\&\phi_x$, a solution of $\xi(y'=\delta)$ can reach a state that ϕ holds, by evolving for the same duration as $x'=\theta\,\&\phi_x$.
- The rule ∀∃-ODE-C is sound because it is a specialized variant of the rule ∀∃-ODE-G.

That concludes the proof. □

Proof of Proposition 4 *Well-formedness:* A timed dynamics $plant^{\langle\mathtt{T}\rangle}$ is well-formed if (1) $t_g \in \mathrm{VAR}(plant)$, (2) ${t_g}' = 1$ is the only program modifying t_g, and (3) $0 \leq \mathtt{T}$. From now on, we only consider well-formed timed programs.

We first introduce useful tools. The following proposition merges two sequentially evolving dynamics into a single one:

Proposition 10. For a program $plant \equiv (x'=\theta, {t_g}'=1)\,\&\phi$, and timed programs $plant^{\langle\mathtt{T}_1\rangle}$ and $plant^{\langle\mathtt{T}_2\rangle}$ with time duration $\mathtt{T}_1$ and $\mathtt{T}_2$, if the formula ϕ is time-invariant, then it holds that $[\![plant^{\langle\mathtt{T}_1\rangle}; plant^{\langle\mathtt{T}_2\rangle}]\!] = [\![plant^{\langle\mathtt{T}_1+\mathtt{T}_2\rangle}]\!]$.

The proposition holds due to the flow property of autonomous differential equations [26] and the evolution constraint being time-invariant.

The following proposition focuses on programs with permanent actions:

Proposition 11. For a program $plant \equiv (x'=\theta, {t_g}'=1)\,\&\phi$, a permanent control action $action$, durations $\mathtt{T}_1$, $\mathtt{T}_2$, if the constraint ϕ is time-invariant, then the following holds:

$$[\![action; plant^{\langle\mathtt{T}_1\rangle}; action; plant^{\langle\mathtt{T}_2\rangle}]\!] = [\![action; plant^{\langle\mathtt{T}_1+\mathtt{T}_2\rangle}]\!]$$

Now we can prove Proposition 4:

Proof. We consider two main cases of $(?\mathtt{T}_1 \leq t \leq \mathtt{T}_2; t_l := 0; action; plant)^*$: the first of which is $(?\mathtt{T}_1 \leq t \leq \mathtt{T}_2; t_l := 0; action; plant)^0$, and the second case is $(?\mathtt{T}_1 \leq t \leq \mathtt{T}_2; t_l := 0; action; plant)^k$ for $k \geq 1$. The first case can be stated as: for any state pair $(\omega,\omega) \in [\![(?\mathtt{T}_1 \leq t \leq \mathtt{T}_2; t_l := 0; action; plant)^0]\!]$ (which is $[\![?\top]\!]$) and $\omega \in [\![\phi]\!]$, we have that $(\omega,\omega) \in [\![action; {plant_s}^{\langle\mathtt{T}_2-\mathtt{T}_1+\epsilon\rangle}]\!]$. The first case holds since (1) $(\omega,\omega) \in {plant_s}^{\langle 0\rangle}$ for any state $\omega \in [\![\phi]\!]$, (2) $[\![{plant_s}^{\langle 0\rangle}]\!] \subseteq [\![{plant_s}^{\langle\mathtt{T}_2-\mathtt{T}_1+\epsilon\rangle}]\!]$, and (3) $(\omega,\omega) \in [\![action; {plant_s}^{\langle\mathtt{T}_2-\mathtt{T}_1+\epsilon\rangle}]\!]$ as $action$ is permanent.

For the second case, we proceed by proving for every state pair $(\omega,\nu) \in [\![(?\mathtt{T}_1 \leq t \leq \mathtt{T}_2; t_l := 0; action; plant)^k]\!]$ that $k \geq 1$, it holds that $(\omega,\nu) \in [\![action; {plant_s}^{\langle\mathtt{T}_2-\mathtt{T}_1+\epsilon\rangle}]\!]$. We prove this case by induction on k. The base case is that for every $(\omega,\nu) \in [\![(?\mathtt{T}_1 \leq t \leq \mathtt{T}_2; t_l := 0; action; plant)]\!]$, it holds that $(\omega,\nu) \in [\![action; {plant_s}^{\langle\mathtt{T}_2-\mathtt{T}_1+\epsilon\rangle}]\!]$. From the promise, by the semantics of timed programs, we get $(\omega,\nu) \in [\![action; {plant_s}^{\langle\epsilon\rangle}]\!]$ (analogous to Proposition 3). For the induction case, assume that the proposi-

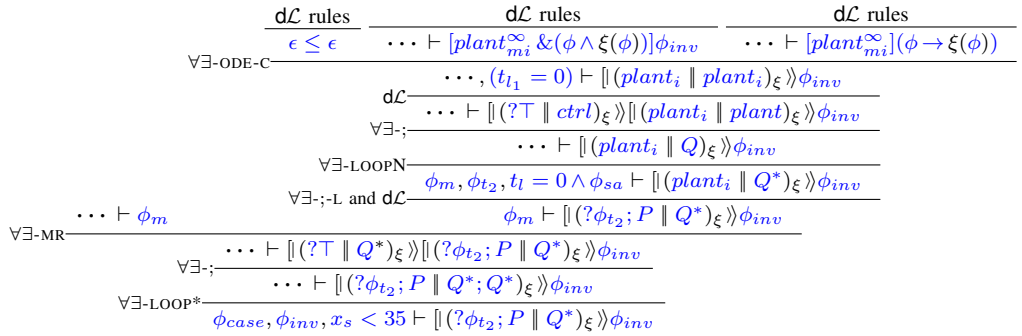

Fig. 13. Proof for the case $\phi_{case} \equiv 34.5 \le x_p < 35 \wedge x_s \ge 35$ (here, $\phi_m \equiv \phi_{case} \wedge \phi_{inv} \wedge x_s < 35 \wedge x_{p_1} < 35 \wedge 0 \le x_{p_1} - x_p \le 0.5$, $\phi_{sa} \equiv x_p - 0.5 \le x_s \le x_p + 0.5$, $plant_{mi}^{\infty} \equiv x_p' = r(1 - x_p/x_L), t_g' = 1, t_l' = 1, \xi(x_p' = r(1 - x_p/x_L)), \xi(t_g' = 1), \xi(t_l' = 1)$, and $\phi_{inv} = 0 \le x_{p_1} - x_p \le 0.5 \wedge \wedge 34.5 < x_{p_1} < 37$)

tion holds for the count of iteration k. Consider a state pair $(\omega, \nu) \in [\![(?\mathrm{T}_1 \le t \le \mathrm{T}_2; t_l := 0; action; plant)^{(k+1)}]\!]$, we know exists ν_1 that $(\omega, \nu_1) \in [\![action; plant_s^{\langle \mathrm{T}_2 - \mathrm{T}_1 + \epsilon \rangle}]\!]$ and $(\nu_1, \nu) \in [\![(?\mathrm{T}_1 \le t \le \mathrm{T}_2; t_l := 0; action; plant)]\!]$. Due to the test $?\mathrm{T}_1 \le t \le \mathrm{T}_2$, we know $\mathrm{T}_1 \le \omega(t) \le \mathrm{T}_2$ and $\mathrm{T}_1 \le \nu_1(t) \le \mathrm{T}_2$. Thus, we get $(\omega, \nu_1) \in [\![action; plant_s^{\langle \mathrm{T}_2 - \mathrm{T}_1 \rangle}]\!]$ as $\nu_1 - \omega \le \mathrm{T}_2 - \mathrm{T}_1$. In addition, we know that $(\nu_1, \nu) \in [\![(action; plant_s^{\langle \epsilon \rangle}]\!]$ from the semantics of $plant$. Then, by Proposition 11, we get $(\omega, \nu) \in [\![(action; plant_s^{\langle \mathrm{T}_2 - \mathrm{T}_1 + \epsilon \rangle}]\!]$. That concludes the proof. □

D. Additional proof derivations for the case study

The bounded sensor attack The main proof structure for the simulation distance under this attack is the same as shown in Fig. 9. The key difference is (1) choosing the appropriate programs and formulas in Fig. 10, and (2) constructing new derivations for the three cases, especially the ϕ_{t_2} case. In particular, in this proof, the formula ϕ_2 should be $34.5 \le x_{p_1} \le 37.1$ and the derivation for $\phi_1 \vdash [\![((?\phi_{t_2}; P)^* \parallel Q^*)_\xi]\!]\phi_2$ is constructed using the multi-step invariant rule ∀∃-MULTI with $\phi_{inv} = 0 \le x_{p_1} - x_p \le 0.5 \wedge \wedge 34.5 < x_{p_1} < 37.1$ into the obligation $\phi_{inv} \vdash [\![((?\phi_{t_2}; P) \parallel Q^*)_\xi]\!]\phi_{inv}$. Then the derivation proceeds by analysis of the following cases:

- $x_p < 34$: this cannot happen due to conflicts with ϕ_{inv}.
- $34 \le x_p < 34.5$: we know that $x_{p_1} < 35$ from ϕ_{inv}, so both P and Q would set to increasing mode. Then $x_{p_1} - x_p$ would monotonically decrease, but still positive. This is because $(x_{p_1} - x_p)' = -(x_{p_1} - x_p) * r/x_L$ is a exponential decay function.
- $34.5 \le x_p < 35.5$: Fig. 13 showcases this case. The invariant holds because the Q^* can first reach a point close to $x_{p_1} = 35$ where $0 < x_{p_1} - x_p \le 0.5$ holds, and after that both P and Q would set to the same mode. In particular, for the case of $x_s < 35$, the Q^* can first reach to a point that $x_{p_1} < 35$ and $x_{p_1} - x_p \le 0.5$, and then both P and Q would set to increasing mode. For the case of $x_s \ge 35$, the Q^* reaches to $x_{p_1} = 35$ first, which means $0 < x_{p_1} - x_p \le 0.5$ and both P and Q would then set to decreasing mode. In both cases, $x_{p_1} - x_p$ would monotonically decrease but stay positive.
- $35.5 \le x_p$: both P and Q would then set to decreasing mode. And again $x_{p_1} - x_p$ would monotonically decrease but still positive.

With the simulation distance being 0.5, the system is timed robust for a $\delta \ge 5 - 0.5/5 = 0.9$. The tolerance notion doesn't apply here as the system has never been in unsafe region.