Verifying Safety for Resilient Cyber-Physical Systems via Reactive Software Restart

Luyao Niu
lniu@wpi.edu
Worcester Polytechnic Institute
Worcester, MA, USA

Dinuka Sahabandu
sdinuka@uw.edu
University of Washington
Seattle, WA, USA

Andrew Clark
aclark@wpi.edu
Worcester Polytechnic Institute
Worcester, MA, USA

Radha Poovendran
rp3@wpi.edu
University of Washington
Seattle, WA, USA

2022 ACM/IEEE 13th International Conference on Cyber-Physical Systems (ICCPS) | 978-1-6654-0967-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCPS54341.2022.00016

ABSTRACT

Resilient cyber-physical systems (CPS) must ensure safety and perform required tasks in the presence of malicious cyber attacks. Recently, restart-based defenses have been proposed in which a CPS mitigates attacks by reverting to an initial safe state. In this paper, we consider a class of reactive restart approaches for CPS under malicious attacks with verifiable safety guarantees. We consider a setting where the controllers are engineered to crash and reboot following faults or attacks. We present a hybrid system model that captures the trade-off between security, availability, and safety of the CPS due to the reactive restart. We develop sufficient conditions under which an affine controller provides verifiable safety guarantees for the physical plant using a barrier certificate approach. We synthesize safety-critical controllers using control barrier functions to guarantee system safety under given timing parameters. We present two case studies on the proposed approach using a warehouse temperature control system and a two-dimensional nonlinear system. Our proposed approach guarantees the safety for both cases.

KEYWORDS

Cyber-physical system, cyber attack, safety verification, restoration, safety-critical synthesis

1 INTRODUCTION

The tight coupling between cyber and physical components of CPS exposes them to new threats. Malicious cyber attacks have been reported in multiple CPS domains, including power systems [39], automobiles [18, 24], and surgical robots [3]. Cyber attacks may lead to safety violations that damage physical infrastructures or harm human operators [22]. To this end, the concept of resilient CPS has attracted increasing research attention. A resilient CPS should be able to withstand known attacks and effectively recover from failures and unknown attacks while performing desired tasks and maintaining safety [9, 15].

Defenses against cyber attacks on CPS have been extensively studied using control- and game-theoretic approaches [17, 20, 27, 30, 37, 44]. These approaches focus on preventing, detecting, and mitigating attacks by constraining the lower level controller behavior so that the system continues to perform its task in spite of the attack. As CPS become increasingly complex, adversaries can disrupt the system by exploiting software vulnerabilities [2]. Such exploits enable an adversary to compromise the low-level control inputs and all sensor data of the CPS, and thus overwhelm and neutralize game- and control-theoretic mitigation mechanisms.

To this end, restart- [1, 2, 7, 8] and software rejuvenation-based mechanisms [6, 35, 36] have been proposed to recover the cyber component of the CPS to a 'clean' state [1] where the impacts from the adversary are limited, at the expense of temporarily losing control over the CPS. These methods leverage the fact that the physical component of CPS can tolerate loss of controller availability for a small number of functioning cycles due to inertia [29]. Once the cyber component is recovered in time, the safety of physical component can still be guaranteed. Nevertheless, the controller being offline during system restart reduces the availability.

Restart-based mechanisms can be broadly classified into two categories. *Proactive* restart approaches periodically restart the system in order to prevent adversaries from gaining a foothold [1, 6–8, 35, 36]. Since the proactive restarts occur at the time of the operator's choosing, online reachability methodologies have been proposed to ensure that the system only restarts if safety can be guaranteed during the time when the controller is inactive [1].

Reactive restart methodologies reboot when certain conditions are met, e.g., when the system crashes due to an erroneous input [2]. Reactive restart approaches are compatible with software diversity and randomization techniques, which cause the system to crash and restart following an adversarial input instead of allowing the adversary to remain undetected [29]. Since the system only restarts following intrusions and software faults, reactive restarts will occur less frequently compared to proactive approaches, but are unpredictable. This unpredictability requires a fundamentally different approach to ensuring and verifying safety that is currently not available in the existing literature.

In this paper, we present a reactive restart-based approach for CPS under cyber attacks in which we can guarantee system safety via barrier certificates. We consider two problems, namely, verifying system safety under a given affine controller, and synthesizing safety-critical controllers under given timing constraints. We make the following specific contributions:

- We formulate a hybrid system model of CPS under cyber attack and reactive restart. Our hybrid model divides the

This work was supported by the National Science Foundation via grant CNS-1941670 and the Office of Naval Research via grant N00014-20-1-2636.

978-1-6654-0967-4/22/$31.00 ©2022 IEEE
DOI 10.1109/ICCPS54341.2022.00016

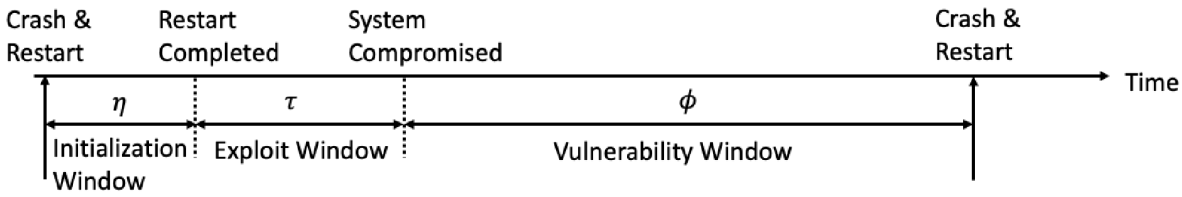

Figure 1: The system timing parameters under malicious attack. The horizon between two consecutive restarts consists of three phases: the initialization window, the exploit window, and the vulnerability window.

system operation into *exploit*, *initialization*, and *vulnerability* phases which are expressed in terms of time windows between restarts (Fig. 1).

- We propose (control) barrier certificate approach to safety verification safety-critical control synthesis of CPS, respectively. The barrier certificate can be computed efficiently using sum-of-squares optimization, avoids time-consuming online reachable set calculation, and is applicable to systems with nonlinearities and uncertainties. We show that the proposed approach can be extended to systems with state estimation by developing a data reload strategy.
- We validate the proposed approach using two case studies, with one on warehouse temperature control system and one on a two-dimensional non-linear system. We demonstrate that the proposed approach guarantees the system to stay within the safety set.

The remainder of this paper is organized as follows. Section 2 presents the related work. Section 3 presents the problem formulation. Our proposed solution approaches for safety verification and safety-critical control synthesis are presented in Section 4 and 5, respectively. Section 6 studies a data reload strategy for CPS whose state is not directly observed. Section 7 presents our case studies. We conclude the paper in Section 8. Preliminary background and the technical proofs are presented in the Appendix.

2 RELATED WORK

Safety verification [31, 33] and control synthesis under safety constraints [5, 12, 14, 34] for CPS in the absence of adversaries or faults have been studied. When faults occur in CPS, fault tolerant control has been extensively studied. See [19] for a detailed survey for fault detection, isolation, and reconfiguration. The faults considered are caused by random failures, which are different with those caused by malicious adversaries. This is because the adversaries are intelligent and can strategically adjust their actions to behave differently compared with random failures.

Cyber attacks can cause safety violations of the CPS, which can damage the physical plant and threaten human operators [3, 18, 25]. There are two main approaches to address malicious attacks in CPS. The first body of literature studies defending and protecting CPS against different types of malicious attacks such as false data injection [17, 32] and denial-of-service [10]. These works aim at preventing the attacks by minimizing the detrimental impact from the adversary. Another category of research focuses on designing fault and intrusion tolerant systems [11, 29, 41] instead of attack prevention. Some components are allowed to fail in intrusion tolerant systems, and will be recovered later. The present paper belongs to the latter category. We propose an attack resilient approach for safety-critical CPS in the presence of a malicious adversary that can exploit the software vulnerabilities.

Our proposed approach mitigates the malicious attack using a reactive restart approach to revert the system to its initial safe state. Restart-based approaches have been proposed in existing literature [1, 2, 6–8, 35, 36]. *Proactive* restart-based approaches are proposed in [1, 6–8, 35, 36], whereas we consider a reactive restart-based approach in this paper where restarts are triggered by controller crashes. In practice, reactive restart occurs less frequently compared with proactive restart, and thus provides higher system availability. Moreover, we provide a verifiable safety guarantee for CPS under malicious attacks using a barrier certificate approach. Compared with the formal guarantees obtained using reachable set computation in [1, 6, 35, 36], our approach provides guarantees through offline construction of barrier certificates and is applicable to nonlinear systems.

In [2], a fault-tolerant reactive restart-based approach is proposed, assuming that the set of verified software components never malfunction. Our paper considers the presence of a malicious adversary who can intelligently corrupt the system including the trustworthy and verified components to alter their outputs.

In Section 6, we will use a buffer to store system state estimate to fasten initialization after each reboot, which can be viewed as a variant of the checkpointing methods introduced in [23, 43].

3 PROBLEM FORMULATION

3.1 System and Adversary Models

We consider a CPS whose physical plant follows dynamics

$$\dot{x} = f(x) + g(x)u, \tag{1}$$

where $\dot{x}$ is the derivative of state variable $x \in \mathbb{R}^n$ with respect to time t, $x \in \mathcal{X} \subseteq \mathbb{R}^n$ is the system state, and $u \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control input. We assume that the vector fields f and g are Lipschitz continuous, and $\mathcal{U}$ is a bounded admissible input set. We denote the solution to system (1) as $x : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$. Let $C = \{x : h(x) \geq 0\} \subseteq \mathcal{X}$, where $h : \mathcal{X} \to \mathbb{R}$ is a continuously differentiable function. We say set C is forward invariant if $x(t) \in C$ for all $t \geq 0$. Given a state x, a feedback controller is defined as $\mu : \mathcal{X} \to \mathcal{U}$, which maps from the state to the set of admissible inputs. Using the system state x, the controller calculates the input u at each time to actuate the physical plant. We denote the solution to Eqn. (1) under controller

μ as $x^{\mu}(t)$. We say controller μ satisfies the safety constraint if

$$x^{\mu}(t) \in C, \ \forall t \geq 0. \tag{2}$$

For example, the safety set C for an autonomous vehicle is the set of vehicle locations where the distance $h(x)$ with the obstacle exceeds a certain threshold. In this paper, we assume that $h(x)$ is a control barrier function (CBF) for the safety set $C = \{x : h(x) \geq 0\}$. We provide the definition of CBF in Appendix A.1. For scenarios where $h(x)$ is not a CBF, our proposed algorithms in Section 4 will report a failure, indicating no feasible solution has been found. When a failure is reported, we can first synthesize a CBF $\tilde{h}(x)$ for a subset $\tilde{C} \subseteq C$ of the safety set, and then apply our proposed approach. The synthesis of CBF, which is not the focus of this work, has been investigated in [13, 42].

We consider the presence of a malicious adversary in the CPS. The objective of the adversary is to destabilize the physical plant so that the system violates the safety constraint. The malicious adversary exploits the vulnerabilities of the cyber component in the CPS to intrude into the system. Once the adversary completes the intrusion, it gains root access and can manipulate the code and/or data in the CPS, e.g., via memory corruption attacks. In addition, we assume that the adversary does not have physical access to the plant and it takes $\tau > 0$ time for the adversary to exploit the vulnerability and gain root access in the system. In practice, parameter τ varies depending on multiple factors including the vulnerabilities of the system and the capability of the adversary.

Once the adversary gains root access in the CPS, it can manipulate the control input u to be any $\tilde{u} \in \mathcal{U}$. As a consequence, the adversary manipulates the behavior of the system and cause the physical plant to deviate from the safety set C, leading to safety violation.

3.2 System Timing Parameters

In this subsection, we identify the relevant system timing parameters that impact the system behavior and introduce how the CPS evolves based on these timing parameters, as shown in Fig. 1. The system observes the state $x(t)$ at each time t and computes the feedback control input u based on $x(t)$. Using the controller crash as an indicator signal, a complete restoration of the CPS is triggered. The restoration is achieved by restarting the whole system and reloading a trusted and uncompromised image of the controller to the system. After the restart, the CPS takes η amount of time to complete the *initialization*, which consists of loading the operating system and controller, initializing the data structures, and re-learning the system state. Parameter η varies for different systems, depending on multiple factors such as the operating system and controller/processor frequency. In practice, the scale of η varies from microseconds to seconds. For instance, restarting the rusEFI engine control unit takes about $20ms$ [8]. Note that re-learning the system state occurs after the operating system is reloaded and the data structure initialization is completed. Thus there is no control input applied to the physical plant during the initialization window.

After the initialization window, the adversary can *exploit* the vulnerabilities in the system and attempt to gain root access in the system during the exploit window. We denote the time required for the adversary to complete exploitation and corresponding attacks as τ. The existence of exploit window τ has been demonstrated in real-world attacks that follow cyber kill chain [28]. The exploit window can also be created by disabling the vulnerable external interface utilized by the adversary [1, 35, 36]. In practice, the value of τ varies depending on multiple factors such as the diversification [16, 29] adopted by the system and the computation power of the adversary. The software running by the CPS can be regarded as safe and uncorrupted during the exploit window, even if the system vulnerabilities are still present.

We note that the controllers are engineered to crash when they receive erroneous inputs from the adversary [29]. If the adversary gains root access to the system at time t, then the controller is corrupted for a limited time. We let t' denote the time at which the controller crashes and define the time interval $[t, t']$, during which the adversary can introduce arbitrary control inputs into the system, as the vulnerability window (Fig. 1). The length ϕ of the vulnerability window is determined by the sensitivity of the controller to faults introduced by the adversary, and can be reduced by software diversification, memory randomization, and other methods that cause the system to crash following erroneous inputs. If the controller compromises immediately after the adversary compromises the system, then ϕ becomes zero.

3.3 Problem Statement

This subsection presents the problem statements. We investigate the following two problems in this paper.

Problem 3.1 (Safety Verification). *Given a feedback controller in the form of $u = Kx + b$, compute $(\overline{\phi}, \overline{\eta}, \underline{\tau}) \in \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}$ so that system (1) is safe with respect to C.*

Problem 3.1 is a verification problem which studies the worst-case behavior of the system so that we can verify the safety of an affine controller [38]. Here we use $\overline{\phi}$ and $\overline{\eta}$ to represent the upper bounds of ϕ and η, and use $\underline{\tau}$ to denote the lower bound of τ. We assume that $u = Kx + b \in \mathcal{U}$ for all $x \in \mathcal{X}$. We aim at solving Problem 3.1 in an *offline* manner in order to minimize online computations by resource-constrained CPS. We present the solution to Problem 3.1 in Section 4.

Problem 3.2 (Safety-Critical Control Synthesis). *Given $(\overline{\phi}, \overline{\eta}, \underline{\tau})$, synthesize a feedback controller during the exploit window so that system (1) is safe with respect to C.*

Problem 3.2 investigates the safety-critical control synthesis problem when the system timing parameters are known. Problem 3.2 needs to be solved in an *online* fashion to calculate the feedback controller. We formulate a quadratic program using control barrier functions to solve Problem 3.2 in Section 5.

4 BARRIER CERTIFICATE-BASED SAFETY VERIFICATION

This section presents the solution approach to Problem 3.1. We first formulate the system operated under the timing parameters in Section 3.2 as a hybrid system (See Appendix A.1 for background on hybrid systems). We then develop sufficient conditions for system safety using a barrier certificate approach. The developed sufficient conditions are later verified using sum-of-squares (SOS) programs. Preliminary background on hybrid systems and barrier certificates can be found in Appendix A.1.

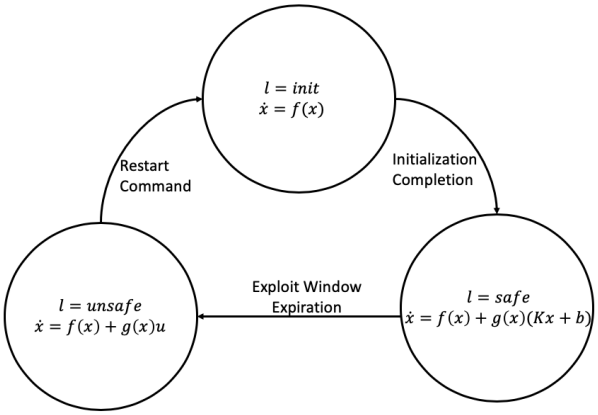

Figure 2: An illustration of the hybrid system H. The set of discrete locations are depicted using circles, and the discrete location transitions are described by arrows. The trigger for each discrete location transition is labeled with each arrow.

Based on the system timing parameters introduced in Section 3.2, system (1) can be in one of the following three modes:

- Safe mode: The system is in the safe mode during the exploit window. When the system is in the safe mode, the adversary has not completed its attack, and thus the correct control input $u = Kx + b$ is applied to the system.
- Initialization mode: The system is in the initialization mode during the initialization window. In this mode, no control input is applied to the system.
- Unsafe mode: The system is in the unsafe mode during the vulnerability window since the adversary gains the root access in the system and thus can arbitrarily manipulate the system behavior.

The transitions among these three modes introduce discrete behavior to the system model. Thus we construct a hybrid system $H = (\mathcal{X}, \mathcal{L}, \mathcal{S}, \mathcal{S}_0, Inv, \mathcal{F}, \Sigma)$ to to jointly model the continuous system dynamics, the discrete transitions among different modes, and the system timing parameters. Each element of the hybrid system is given as follows.

- $\mathcal{X}$ is the set of continuous system states.
- $\mathcal{L} = \{safe, unsafe, init\}$ is the set of discrete locations modeling the modes of the system, where $safe$, $unsafe$, and $init$ correspond to safe, unsafe, and initialization modes of the system, respectively.
- $\mathcal{S} = \mathcal{X} \times \mathcal{L}$ is the state space of the hybrid system.
- $\mathcal{S}_0 \subseteq \mathcal{S}$ is the set of initial states of the hybrid system.
- $Inv : \mathcal{L} \to 2^{\mathcal{X}}$ is the invariant.
- $\mathcal{F} = \{f_{cl}^{safe}, f_{cl}^{init}, f_{cl}^{unsafe}\}$ is the set of continuous system dynamics for each discrete location, where $f_{cl}^{safe}(x; K, b) = f(x) + g(x)(Kx + b)$, $f_{cl}^{init}(x) = f(x)$, and $f_{cl}^{unsafe}(x, u) = f(x) + g(x)u$.
- $\Sigma = \{((x, safe), (x, unsafe)), ((x, unsafe), (x, init)), ((x, init), (x, safe))\}$ is the set of discrete transitions among the locations.

We show the constructed hybrid system in Fig. 2, where the discrete locations $\mathcal{L}$ and the set of transitions in Σ are depicted using circles and arrows, respectively. Note that the discrete location transitions are defined by modes of the system, and no other discrete location transitions are allowed except those modeled by Σ. Autonomous transitions and time-dependent transition coexist in hybrid system H. The transition from $l = safe$ to $l = unsafe$ occurs when the exploit window expires. The other transitions from $l = unsafe$ to $init$ and from $l = init$ to $safe$ are triggered by the restart command and the completion of initialization, and are independent of the control input u. According to the system timing parameters, the time that the system spends in each discrete location is bounded between two restarts. When the system is in location $l = safe$, the closed-loop dynamics are parameterized by K and b. When the system is location $l = init$, no control input is applied to the system and the closed loop dynamics are given as $f_{cl}^{init} = f(x)$. When the system is in location $l = unsafe$, the control input $u \in \mathcal{U}$ is maliciously chosen by the adversary, and thus needs to be treated as a disturbance. The closed-loop dynamics in this case are given as $f_{cl}^{unsafe} = f(x) + g(x)u$. Note that the continuous system state does not incur any jump during any discrete location transition.

Given the hybrid system H, we develop sufficient conditions under which controller $u = Kx + b$ guarantees that system (1) is safe with respect to C. Our idea is that if the system starts from a 'sufficiently safe' state, then the system trajectory will not leave the safety set C if we can limit the amount of time that the system is compromised by the adversary. Since the system is correctly controlled only when in location $l = safe$, we thus need to guarantee that the system will reach the 'sufficiently safe' state before transitioning to location $l = unsafe$. Based on the closed-loop system dynamics in $\mathcal{F}$, we then need to guarantee that (i) the system trajectory remains in C when $l = init$ and $l = unsafe$ by bounding the lengths of the vulnerability window and initialization window, and (ii) the system trajectory reaches the 'sufficiently safe' state when $l = safe$. The sufficient condition to guarantee the safety of system (1) under attack is given in the following theorem, whose proof is presented in Appendix A.2.

Theorem 4.1. *Consider the system in Eqn. (1) under attack and a safety set C. Let $h_1(x) = h(x) - c_1$ and $h_2(x) = h_1(x) - c_2$. We define $C_1 = \{x : h_1(x) \geq 0\}$ and $C_2 = \{x : h_2(x) \geq 0\}$. Suppose $x(0) \in C_2$. If there exist constants $c_1, c_2 \geq 0$ and a class $\mathcal{K}$ function $\alpha(\cdot)$ such that*

$$\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x; K, b) \geq \frac{c_1 + c_2}{\tau}, \ \forall x \in C \setminus C_2 \tag{3a}$$

$$\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x; K, b) \geq -\alpha(h_2(x)), \ \forall x \in C_2 \tag{3b}$$

$$\frac{\partial h}{\partial x}(x) f_{cl}^{init}(x) \geq -\frac{c_1}{\eta}, \ \forall x \in C \tag{3c}$$

$$\frac{\partial h}{\partial x}(x) f_{cl}^{unsafe}(x, u) \geq -\frac{c_2}{\phi}, \ \forall (x, u) \in C_1 \times \mathcal{U} \tag{3d}$$

then system (1) is safe with respect to C.

Theorem 4.1 indicates that if we can find parameters c_1, c_2, ϕ, η, and τ, then the system is safe using controller $u = Kx + b$. However, verifying the conditions in Eqn. (3) is not straightforward for arbitrary systems. When the the following semi-algebraic conditions

are satisfied, we show that we can verify Eqn. (3) via a sum-of-squares (SOS) program. We make the following assumption.

Assumption 4.1. *We assume that vector field $f_{cl}(x)$ and function $h(x)$ are polynomial in x. In addition, we let $\mathcal{U} = \{u : v(u) \geq 0\}$, where $v(u)$ is polynomial in u.*

When Assumption 4.1 holds, we can verify the conditions in Eqn. (3) using the following SOS program.

Proposition 4.2. *If there exist parameters c_1, c_2, z_1, z_2, and z_3 and a class $\mathcal{K}$ function $\alpha(\cdot)$ so that the following expressions are SOS:*

$$\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x;K,b) - z_3 + l(x)h(x)h_2(x) \tag{4a}$$
$$\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x;K,b) + \alpha(h_2(x)) - r(x)h_2(x) \tag{4b}$$
$$\frac{\partial h}{\partial x}(x) f_{cl}^{init}(x) + z_1 - p(x)h(x) \tag{4c}$$
$$\frac{\partial h}{\partial x}(x) f_{cl}^{unsafe}(x,u) + z_2 - q(x,u)h_1(x)v(u) \tag{4d}$$

where $l(x), r(x), p(x), q(x,u)$ are SOS and $z_1, z_2, z_3 \geq 0$, then c_1, c_2, ϕ, η, and τ satisfying

$$\frac{c_1}{\eta} = z_1, \quad \frac{c_2}{\phi} = z_2, \quad \frac{c_1 + c_2}{\tau} = z_3 \tag{5}$$

meet the conditions in Eqn. (3).

The proof of Proposition 4.2 can be found in Appendix A.2. Proposition 4.2 implies that we can verify the safety of system (1) by the existence of SOS polynomials $l(x), p(x), r(x)$, and $q(x,u)$ along with non-negative scalars c_1, c_2, z_1, z_2 and z_3 such that Eqn. (4) and (5) are satisfied. However, directly verifying the existence of all the aforementioned variables in Eqn. (4) is challenging since it requires us to solve for C_1, C_2 and parameters ϕ, η, and τ simultaneously, leading to bilinearities in Eqn. (4). In the remainder of this section, we address this challenge by developing approximate solution algorithms to verify the existence of these variables in Eqn. (4). The idea is to constrain Eqn. (3) for all $x \in C$ to convert the bilinear terms in Eqn. (4) to linear terms. We consider a relaxation that replaces $h_1(x)$ and $h_2(x)$ in the last terms of Eqn. (4a) to Eqn. (4d) with $h(x)$. This relaxation eliminates variables c_1 and c_2 at the expense of conservatively constraining the system behavior over C.

Lemma 4.3. *If there exist parameters c_1, c_2, z_1, z_2, and z_3 so that the following expressions are SOS:*

$$\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x;K,b) - z_3 - l(x)h(x) \tag{6a}$$
$$\frac{\partial h}{\partial x}(x) f_{cl}^{init}(x) + z_1 - p(x)h(x) \tag{6b}$$
$$\frac{\partial h}{\partial x}(x) f_{cl}^{unsafe}(x,u) + z_2 - q(x,u)h(x)v(u) \tag{6c}$$

where $l(x), p(x), q(x,u)$ are SOS polynomials and $z_1, z_2, z_3 \geq 0$, then c_1, c_2, ϕ, η, and τ satisfying

$$\frac{c_1}{\eta} = z_1, \quad \frac{c_2}{\phi} = z_2, \quad \frac{c_1 + c_2}{\tau} = z_3 \tag{7}$$

Algorithm 1 Solution algorithm for computing $\overline{\phi}$, $\overline{\eta}$, and $\underline{\tau}$ for worst-case c_1 and c_2.

Input: Parameters ϵ_1 and ϵ_2. The maximum value $\overline{c}_1$ and $\overline{c}_2$ for c_1 and c_2, respectively
Output: Parameters $\overline{\phi}$, $\overline{\eta}$, $\underline{\tau}$, c_1, and c_2
Initialization: Initialize $c_{UB,1} \leftarrow \overline{c}_1$, $c_{LB,1} \leftarrow 0$, $c_{UB,2} \leftarrow \overline{c}_2$, $c_{LB,2} \leftarrow 0$
while $|c_{UB,1} - c_{LB,1}| \geq \epsilon_1$ or $|c_{UB,2} - c_{LB,2}| \geq \epsilon_2$ **do**
  $c_1 \leftarrow (c_{UB,1} + c_{LB,1})/2$, $c_2 \leftarrow (c_{UB,2} + c_{LB,2})/2$
  if Eqn. (4) is feasible **then**
    if $z_1 = 0$ **then**
      $\eta \leftarrow \infty$
    else
      $\eta \leftarrow c_1/z_1$
    end if
    if $z_2 = 0$ **then**
      $\phi \leftarrow \infty$
    else
      $\phi \leftarrow c_2/z_2$
    end if
    if $z_3 = 0$ **then**
      $\tau \leftarrow (c_1 + c_2)/(\inf_{x \in C} f_{cl}^{safe}(x;K,b))$
    else
      $\tau \leftarrow (c_1 + c_2)/z_3$
    end if
    $(\overline{\phi}, \overline{\eta}, \underline{\tau}) \leftarrow (\phi, \eta, \tau)$
    $c_{UB,1} \leftarrow c_1$, $c_{UB,2} \leftarrow c_2$
  else
    $c_{LB,1} \leftarrow c_1$, $c_{LB,2} \leftarrow c_2$
  end if
end while
return $c_{UB,1}$, $c_{UB,2}$, $\overline{\phi}$, $\overline{\eta}$, and $\underline{\tau}$

meet the following conditions

$$\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x;K,b) \geq \frac{c_1 + c_2}{\tau}, \ \forall x \in C \tag{8a}$$
$$\frac{\partial h}{\partial x}(x) f_{cl}^{init}(x) \geq -\frac{c_1}{\eta}, \ \forall x \in C \tag{8b}$$
$$\frac{\partial h}{\partial x}(x) f_{cl}^{unsafe}(x,u) \geq -\frac{c_2}{\phi}, \ \forall (x,u) \in C \times \mathcal{U} \tag{8c}$$

The proof of Lemma 4.3 can be found in Appendix A.2. Lemma 4.3 indicates that the timing parameters ϕ, η, and τ can be solved using an SOS program whose constraint set is given as Eqn. (6). Note that $h_2(x) \geq 0 \implies h_1(x) \geq 0 \implies h(x) \geq 0$. We can thus characterize the relaxation used in Lemma 4.3 as follows.

Corollary 4.4. *Any feasible solution z_1, z_2, z_3 to Eqn. (4) is also feasible to Eqn. (6).*

In some scenarios, the relaxation in Lemma 4.3 may be overly conservative, rendering Eqn. (6) to be infeasible. To this end, we present two approximate solution algorithms to compute ϕ, η, and τ, along with sets C_1 and C_2. Our idea is that the bilinear terms in Eqn. (4) become linear once one of the variables is given. Thus we can fix one parameter and search for the other that yields Eqn. (4) to be feasible.

We first discuss how to compute the timing parameters ϕ, η, and τ if we are given sets C_1 and C_2. Assume that we know the maximum and minimum values of c_1 and c_2. The maximum and minimum values of c_1 and c_2 can be chosen as $\bar{c}_1 \in (0, \sup_{x\in C} h(x))$ and $\bar{c}_2 = \sup_{x\in C} h(x) - \bar{c}_1$. We then aim at computing the timing parameters for the worst-case choices of C_1 and C_2. The worst-case choices of C_1 and C_2 are defined as $C_1 = \{x : h(x) - \underline{c}_1 \geq 0\}$ and $C_2 = \{x : h_1(x) - \underline{c}_2 \geq 0\}$, respectively, where $\underline{c}_1$ and $\underline{c}_2$ are the lower bounds of parameters c_1 and c_2 that renders the SOS program in Eqn. (4) to be feasible. We note that as c_1 and c_2 decrease, the volumes of C_1 and C_2 increase, which tolerates less delay for the controller crash when $l = unsafe$ and less time consumption for initialization in the worst-case.

We propose Algorithm 1 to compute the parameters. Algorithm 1 first initializes the upper and lower bounds for the search range of c_1 and c_2, respectively. We use subscript UW to represent upper bound and subscript LB to represent lower bound. Algorithm 1 then verifies if the expressions in Eqn. (4) are SOS for given c_1 and c_2 computed at line 5. When the values of c_1 and c_2 are given, verifying the feasibility of Eqn. (4) can be done via an SOS program. If there exist SOS polynomials $l(x)$, $p(x)$, $r(x)$, and $q(x,u)$ such that Eqn. (4) is feasible, then Algorithm 1 updates the upper bounds of search ranges to decrease the values of c_1 and c_2 as shown in line 23. Otherwise the lower bounds are updated (line 25). This process is repeated until the upper and lower bounds of both c_1 and c_2 become close.

Algorithm 2 Solution algorithm for computing c_1 and c_2 for worst-case ϕ, η, and τ.

1: **Input**: Parameters ϵ_1, ϵ_2 and ϵ_3. The maximum value ϕ_{UB}, η_{UB}, and τ_{UB} and minimum values ϕ_{LB}, η_{LB}, and τ_{LB} for ϕ, η and τ, respectively
2: **Output:** Parameters ϕ, η, τ, c_1, and c_2
3: **Initialization**: Initialize ϕ_{UB}, ϕ_{LB}, η_{UB}, η_{LB}, τ_{UB}, and τ_{LB}
4: **while** $|\phi_{UB}-\phi_{LB}| \geq \epsilon_1$ or $|\eta_{UB}-\eta_{LB}| \geq \epsilon_2$ or $|\tau_{UB}-\tau_{LB}| \geq \epsilon_3$ **do**
5: $\phi \leftarrow (\phi_{UB} + \phi_{LB})/2$, $\eta \leftarrow (\eta_{UB} + \eta_{LB})/2$, $\tau \leftarrow (\tau_{UB} + \tau_{LB})/2$
6: $c_1 \leftarrow \max\left\{-\eta \inf_{x\in C}\left\{\frac{\partial h}{\partial x}(x) f_{cl}^{init}(x)\right\}, 0\right\}$
7: $c_2 \leftarrow \max\left\{-\phi \inf_{x\in C, u\in\mathcal{U}}\left\{\frac{\partial h}{\partial x}(x) f_{cl}^{unsafe}(x,u)\right\}, 0\right\}$
8: $z_1 \leftarrow c_1/\eta$, $z_2 \leftarrow c_2/\phi$, $z_3 \leftarrow (c_1 + c_2)/\tau$
9: **if** Eqn. (4) is feasible **then**
10: $(\bar{c}_1, \bar{c}_2) \leftarrow (c_1, c_2)$
11: $\phi_{LB} \leftarrow \phi$, $\eta_{LB} \leftarrow \eta$, $\tau_{UB} \leftarrow \tau$
12: **else**
13: $\phi_{UB} \leftarrow \phi$, $\eta_{UB} \leftarrow \eta$, $\tau_{LB} \leftarrow \tau$
14: **end if**
15: **end while**
16: **return** $\bar{c}_1$, $\bar{c}_2$, ϕ_{UB}, η_{UB}, and τ_{UB}

We next investigate how to compute c_1 and c_2 given the worst-case timing parameters ϕ, η, and τ. We assume that the upper and lower bounds of the timing parameters are given. Here we say a choice of ϕ is worse than ϕ' if $\phi < \phi'$ since a narrower vulnerability window indicates that the crash signal needs to be issued in a more timely manner and thus requires a more delicate design. Similar arguments can be made for η and τ.

We use Algorithm 2 to compute c_1 and c_2, along with the worst-case timing parameters. Algorithm 2 first initializes the search range for ϕ, τ, and η. Similar to Algorithm 1, we use subscripts UB and LB to represent upper and lower bounds, respectively. Then at each iteration from line 4 to 15, Algorithm 2 computes c_1 and c_2 using the given values of ϕ, τ, and η as shown in lines 6 and 7. Note that lines 6 and 7 bound the values of c_1 and c_2 from above since we search over all $x \in C$. Particularly, if c_1 or c_2 is zero, it indicates that the closed-loop dynamics are monotonically non-decreasing, which yields largest C_1 or C_2. Given a pair of (c_1, c_2) that is non-zero, we verify if Eqn. (4) is feasible under this choice of (c_1, c_2). If Eqn. (4) is feasible, we then update the lower bounds of ϕ and η and the upper bound of τ (line 11). Otherwise we update the upper bounds of ϕ and η and the lower bound of τ (line 13).

5 CONTROL BARRIER FUNCTION-BASED SAFETY-CRITICAL CONTROL SYNTHESIS

In this section, we propose a control barrier function-based approach to synthesize a controller with safety guarantee to solve Problem 3.2. Given ϕ, η, and τ, we can approximately calculate c_1 and c_2 as

$$c_1 = \max\left\{-\eta \inf_{x\in C}\left\{\frac{\partial h}{\partial x}(x) f(x)\right\}, 0\right\}, \tag{9a}$$

$$c_2 = \max\left\{-\phi \inf_{x\in C, u\in\mathcal{U}}\left\{\frac{\partial h}{\partial x}(x) f(x) + \frac{\partial h}{\partial x}(x) g(x) u\right\}, 0\right\}. \tag{9b}$$

Note that the synthesized controller needs to guarantee that the system remains in C_2 before the discrete transition from location $l = safe$ to $unsafe$ occurs. Moreover, it needs to ensure that the system trajectory reaches C_2 during the exploit window. Using parameters ϕ, η, τ, c_1, and c_2, we can compute a feedback controller satisfying these two properties as follows.

Proposition 5.1. *During the exploit window, if the control input at each time t is computed as*

$$\min_{u\in\mathcal{U}} \quad u^\top Q u \tag{10a}$$

$$s.t. \quad \frac{\partial h}{\partial x}(x) f(x) + \frac{\partial h}{\partial x}(x) g(x) u + \alpha(h(x)) \geq 0, \tag{10b}$$

$$\frac{\partial h_2}{\partial x}(x) f(x) + \frac{\partial h_2}{\partial x}(x) g(x) u + \gamma\, sgn(h_2(x)) |h_2(x)|^\rho \geq 0 \tag{10c}$$

where Q is a positive definite matrix, and parameters $\rho \in [0, 1)$ and $\gamma > 0$ are chosen so that $\underline{\tau} \geq \frac{1}{\gamma(1-\rho)} |c_1 + c_2|^{1-\rho}$, then system (1) remains safe and reaches C_2 within finite time $\underline{\tau}$.

Proof. The proposition holds by the properties of ZCBF and FCBF presented in Lemma A.3 and Lemma A.4, respectively. □

In Proposition 5.1, we can first pick some $\rho \in [0, 1)$. Then we can choose parameter $\gamma = \frac{1}{\underline{\tau}(1-\rho)} |c_1 + c_2|^{1-\rho}$ for any given $\underline{\tau}$, c_1, and c_2. Proposition 5.1 indicates that we can synthesize a controller when the closed-loop system is at location $l = safe$ by solving a sequence of quadratic programs (QP) when $\mathcal{U}$ is a convex set.

6 SAFETY VERIFICATION FOR CPS WITH STATE ESTIMATION

In Section 4 and 5, we assume that the system state x is observable. However, this may not always hold for all CPS. In practice, we rely on sensor measurements to estimate the system state. We denote the state estimate of system state x as $\hat{x}$. In this case, the control input is calculated as $u = K\hat{x} + b$ using the estimate. Incorporating state estimation imposes the following challenges: (i) the estimate can be compromised by the adversary if the system is compromised, and (ii) the system needs additional re-learning time during the initialization window to re-estimate the system state after the reboot. This section presents a data reload-based approach to guarantee the safety of the system when the system state is not directly observed.

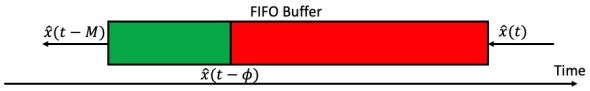

Figure 3: A FIFO buffer of size M is used to checkpoint the historical state estimate. The red region represents the estimates that may be compromised, and the green region represents the estimates that are safe.

We introduce an additional FIFO buffer of size $M \geq \phi$ to store the previous state estimate for later reload use, as shown in Fig. 3, where ϕ can be chosen as the maximum value determined by the CPS design. The storage should allow read and write instructions. Moreover, the data in storage should not be wiped after the system reboot. Finally, we assume that the adversary can manipulate the data to be pushed into the buffer $\hat{x}(t)$, while it cannot compromise the data that has already been stored in the buffer $\hat{x}(t')$ for all $t' < t$. In addition, we assume that the time consumption required for reloading data from storage is negligible compared with re-learning, and thus data reload is instantaneous.

In the following, we study if we can shorten the re-learning process by leveraging data reload from the buffer. Suppose the system reboots at time t. Then after the initialization, the system reloads $\hat{x}(t')$ as the state estimate, where $t' = t-\phi$ is the most recent time instant when the system is guaranteed not to be compromised by the adversary. By reloading $\hat{x}(t')$, the system saves the estimation time. We let the state estimate evolves as the following dynamics:

$$\dot{\hat{x}} = f(\hat{x}) + g(\hat{x})(K\hat{x} + b) + L\left(w(x, K\hat{x} + b) - w(\hat{x}, K\hat{x} + b)\right),$$

where $w(x, u)$ is the observation function. The last term $L(w(x, K\hat{x} + b) - w(\hat{x}, K\hat{x} + b))$ allows the system estimate $\hat{x}$ to converge to x when $\hat{x} \neq x$, and to correctly track x when $\hat{x} = x$.

We define $y = [x, \hat{x}]^\top$. Then following a similar analysis as given in Section 4, we can construct a hybrid system $H = (\mathcal{Y}, \mathcal{L}, \mathcal{S}, \mathcal{S}_0, Inv, \mathcal{F}, \Sigma)$, where $\mathcal{F} = \{f_{cl}^{safe}(y; K, b), f_{cl}^{init}(y), f_{cl}^{unsafe}(y, u)\}$ with

$$f_{cl}^{safe}(x; K, b) = f(x) + g(x)(K\hat{x} + b)$$

$$f_{cl}^{safe}(\hat{x}; K, b) = f(\hat{x}) + g(\hat{x})(K\hat{x} + b) + L(w(x, K\hat{x} + b) - w(\hat{x}, K\hat{x} + b))$$

$$f_{cl}^{safe}(y; K, b) = \begin{bmatrix} f_{cl}^{safe}(x; K, b) \\ f_{cl}^{safe}(\hat{x}; K, b) \end{bmatrix},$$

$$f_{cl}^{init}(y) = \begin{bmatrix} f_{cl}^{init}(x) \\ \mathbf{0} \end{bmatrix}, \; f_{cl}^{unsafe}(y, u) = \begin{bmatrix} f_{cl}^{unsafe}(x, u) \\ \mathbf{0} \end{bmatrix}.$$

We define $d(y) = c_3 - \|x - \hat{x}\|$ for some $c_3 > 0$ to model the set of states y where the distance between the state estimate $\hat{x}$ and state x upper bounded by c_3. We then develop the sufficient conditions under which the system is guaranteed to be safe.

Proposition 6.1. *Consider the hybrid system $H = (\mathcal{Y}, \mathcal{L}, \mathcal{S}, \mathcal{S}_0, Inv, \mathcal{F}, \Sigma)$ and a safety set C. Let $d(y) = c_3 - \|x - \hat{x}\|$. If there exist constants $c_1, c_2 \geq 0$, $c_3 > 0$ and class $\mathcal{K}$ functions $\alpha_1(\cdot), \alpha_2(\cdot)$ such that*

$$\frac{\partial d}{\partial y}(y) f_{cl}^{safe}(y; K, b) \geq -\alpha_1(d(y)), \; \forall x \in C, \hat{x} \in \mathcal{B}(x) \tag{11a}$$

$$\frac{\partial h_2}{\partial y}(y) f_{cl}^{safe}(y; K, b) \geq \frac{c_1 + c_2}{\tau}, \; \forall x \in C \setminus C_2, \hat{x} \in \mathcal{B}(x) \tag{11b}$$

$$\frac{\partial h_2}{\partial y}(y) f_{cl}^{safe}(y; K, b) \geq -\alpha_2(h_2(x)), \; \forall x \in C_2, \hat{x} \in \mathcal{B}(x) \tag{11c}$$

$$\frac{\partial h}{\partial y}(y) f_{cl}^{init}(y) \geq -\frac{c_1}{\eta}, \; \forall x \in C, \hat{x} \in \mathcal{B}(x) \tag{11d}$$

$$\frac{\partial h}{\partial y}(y) f_{cl}^{unsafe}(y, u) \geq -\frac{c_2}{\phi}, \; \forall (x, \hat{x}, u) \in C_1 \times \mathcal{B}(x) \times \mathcal{U} \tag{11e}$$

where $\mathcal{B}(x) = \{\hat{x} : c_3 - \|x - \hat{x}\| \geq 0\}$, then the system is safe with respect to C.

The proof of Proposition 6.1 can be found in Appendix A.2. Using the sufficient conditions in Eqn. (11), we can verify the safety of CPS whose states are not directly observed.

7 SIMULATION CASE STUDIES

In this section, we present two case studies. The first study is on a warehouse temperature control system [1, 40], and the second one is on a non-linear dynamical system introduced in [21, 33].

7.1 Warehouse Temperature Control System

In this subsection, we consider a warehouse temperature control system, consisting of a heater and cooler to the room and a conditioner in the floor [1, 40]. Let the temperature of the floor, room, and outside environment be x_1, x_2, and T, respectively. The dynamics modeling the heat transfer between the warehouse and outside environment are given as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -1.8087(x_1 - x_2) \\ 0.4628(x_2 - T) + 22.2985(x_1 - x_2) \end{bmatrix} + \begin{bmatrix} \frac{1}{6000 \times 115}, & 0 \\ 0 & \frac{1}{69.96 \times 800} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}. \tag{12}$$

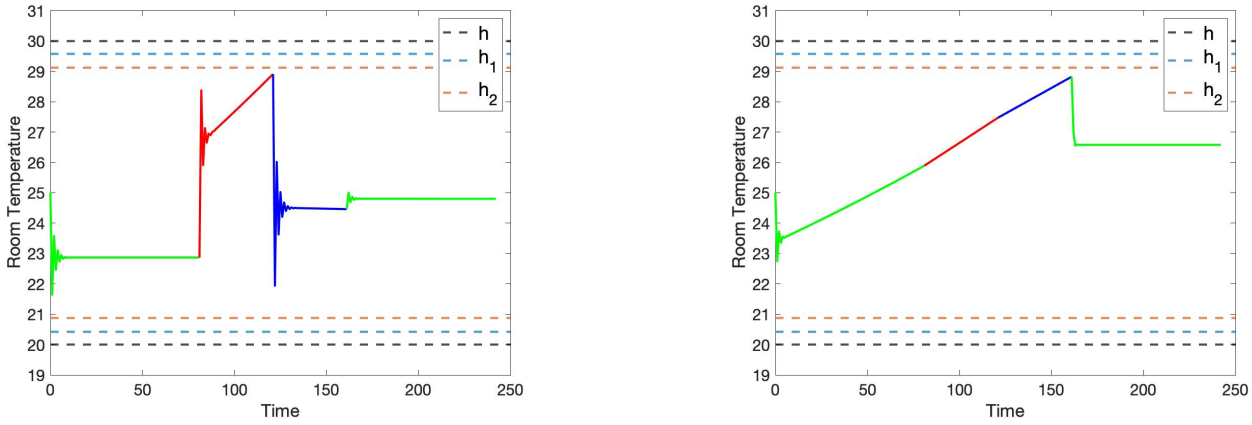

Figure 4: Fig. 4a shows the evolution of room temperature x_2 in system (12) over time when an affine controller $u = Kx + b$ is applied during the exploit window. Fig. 4b shows the room temperature evolution when the safety-critical controller synthesized using Proposition 5.1 is applied during the exploit window, with parameters $\alpha(h(x)) = h(x)$, $\rho = 0.99$ and $\gamma = 20$. In Fig. 4a and 4b, the dashed lines represent the boundaries of C, C_1, and C_2. The system trajectories are shown using solid lines. The parts of the trajectory in green, red, and blue colors represent the parts corresponding to the exploit window, vulnerability window, and initialization window, respectively.

Here the coefficients are jointly determined by the mass of the floor and the air inside the room, the heat capacities of the floor and air, and the heat transfer coefficients. Control inputs u_1 and u_2 model the heat transfer from the floor heater to the floor, and heat transfer from the room heater to the room air, respectively. More detailed explanations on the dynamics in Eqn. (12) can be found in [40].

We set the safety set for the temperature control system as $C = \{x : x_2 \in [20°C, 30°C]\}$, i.e., the room temperature should be maintained within $20°C$ to $30°C$. In this case, we define $h(x) = (30 - x_2)(x_2 - 20)$. We design an affine controller $u = Kx + b$ where

$$K = \begin{bmatrix} 7210 & 0 \\ 0 & 7210 \end{bmatrix}, b = \begin{bmatrix} 1100 & 0 \\ 0 & 1100 \end{bmatrix},$$

using which the room temperature converges to safety set C. We let the outside temperature $T = 10°C$. We set the initial floor and room temperature as $x_1(0) = 23°C$ and $x_2(0) = 25°C$, respectively. Using Algorithm 1, we obtain that $\phi = 40.073$, $\eta = 40.074$, and $\tau = 80.147$. In addition, we have that $C_1 = \{x_2 : x_2 \in [20.4174, 29.5826]\}$ and $C_2 = \{x_2 : x_2 \in [20.8769, 29.1231]\}$. We assume that the controller is updated every $1s$, and simulate the room temperature using our proposed approach. We mark the time period for the exploit window, vulnerability window, and initialization window using green, red, and blue colors, respectively, as shown in Fig. 4a. We observe that the room temperature x_2 is bounded within C, and thus our proposed approach guarantees the safety property.

We investigate the controller synthesis for warehouse temperature control system using the quadratic program presented in Proposition 5.1. We let $\alpha(h(x)) = h(x)$, $\rho = 0.99$, and $\gamma = 20$. The timing parameters are computed by Algorithm 1. The evolution of the room temperature over time is then presented in Fig. 4b. We observe that the room temperature is guaranteed to stay within $[20°C, 30°C]$. Moreover, the controller implemented during the exploit window (i.e., when the hybrid system is at location $l = safe$) tends to maintain the room temperature around $26.58°C$ (i.e., the sharp decrease after the initialization mode) so as to tolerate the potential increase or decrease introduced by the adversary during the future vulnerability window.

7.2 System with Polynomial Dynamics

In this subsection, we demonstrate that our proposed approach is applicable to non-linear systems. We consider a two-dimensional system whose dynamics are given as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -x_1 + \frac{1}{3}x_1^3 - x_2 \end{bmatrix} + \begin{bmatrix} 0.1 \\ 0.1 \end{bmatrix} u \tag{13}$$

where $x = [x_1, x_2]^\top \in \mathcal{X} \subseteq \mathbb{R}^2$ is the system state, and $u \in \mathcal{U} \subseteq \mathbb{R}$ is the control input. We let $C = \{x : h(x) \geq 0\}$ be the safety set, where $h(x)$ is given as

$$\begin{aligned} h(x) = {} & 0.1973x_1^4 + 0.42741x_1^3x_2 + 0.17451x_1^2x_2^2 + 0.1079x_1x_2^3 \\ & -8.335\times10^{-7}x_2^4 + 3.3808\times10^{-6}x_1^3 + 3.0606\times10^{-6}x_1^2x_2 + 1.0894x_1x_2^2 \\ & + 0.43842x_2^3 - 1.1838x_1^2 - 1.2822x_1x_2 - 2.1238x_2^2 - 5.7966\times10^{-7}x_1 \\ & - 6.5873\times10^{-7}x_2 + 0.014414. \end{aligned}$$

We represent the boundary of the safety set, i.e., $\{x : h(x) = 0\}$ using the dashed black line in Fig. 5a and 5b. The system in Eqn. (13) can be stabilized via an affine controller $u = Kx + b$, where $K = [13, -14.5]$ and $b = -15$ are designed such that the state x converges to safety set C.

We study the safety verification problem of system (13) when an affine controller $u = [13, -14.5]x - 15$ is applied. Note that when the system is compromised by the adversary (i.e., during

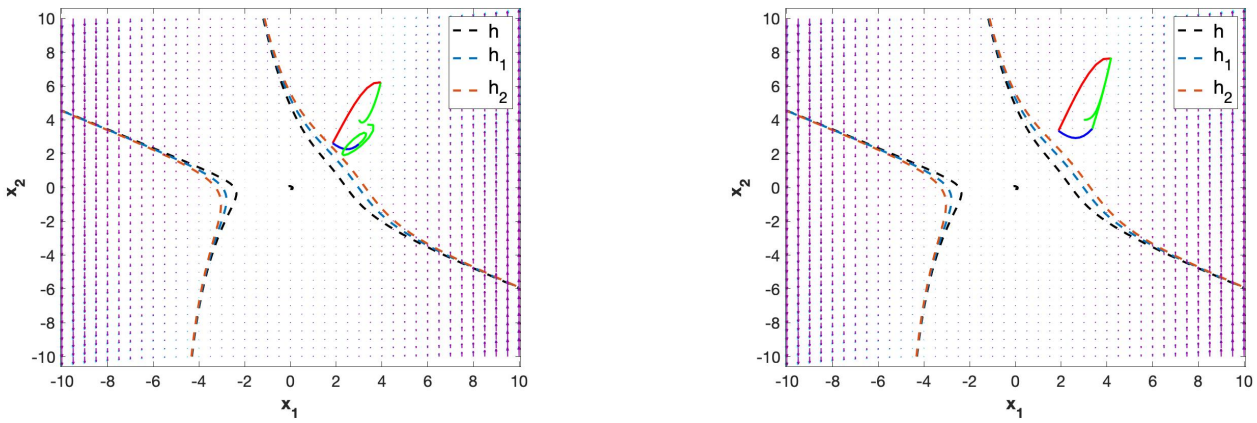

Figure 5: Fig. 5a shows the system trajectory for system (13) generated using an affine controller $u = [13, -14.5]x - 15$ during the exploit window with parameters $c_1 = c_2 = 5.0085$, $\phi = 8$, $\eta = 8.4$, and $\tau = 16.9$ calculated by Algorithm 1. Fig. 5b shows the system trajectory for system (13) generated using the the safety-critical controller during the exploit window with parameters $\alpha(h(x)) = 2h(x)$, $\rho = 0.99$ and $\gamma = 10$. In both figures, the dashed lines represent the boundaries of C, C_1, and C_2. The system trajectories are represented using solid lines. The parts of the trajectories in green, red, and blue colors represent the parts corresponding to the exploit window, vulnerability window, and initialization window, respectively.

the vulnerability window), the affine controller is manipulated to arbitrary $\tilde{u} \in \mathcal{U}$ such that $\tilde{u} \neq Kx + b$. In Fig. 5a, we present the vector fields of the closed-loop dynamics for f_{cl}^{safe} and f_{cl}^{unsafe} using cyan and magenta arrows, respectively.

We set the upper and lower bounds of c_1 and c_2 in Algorithm 1 as 300 and 0, respectively. Using Algorithm 1, we obtain that $c_1 = c_2 = 5.0085$. In Fig. 5a and 5b, the boundaries of sets C_1 and C_2 are plotted using the dashed lines in blue and orange colors, respectively. In addition, we have that $\phi = 8$, $\eta = 8.4$, and $\tau = 16.9$. We set the controller update period as $0.05s$, i.e., frequency of $20Hz$, and simulate the system trajectory with initial state $x(0) = [3, 4]^\top$ as shown in Fig. 5a. We observe that the system is always safe with respect to C despite the system moves towards the boundary of the safety set when the adversary compromises the system (the part of the trajectory in red color). Moreover, the system is steered away from the boundary of C during the exploit window (i.e., the part of the trajectory in green color).

We finally consider the safety-critical control synthesis problem stated in Problem 3.2. We synthesize the controller during the exploit window using the quadratic program formulated in Proposition 5.1. We choose $\alpha(\cdot)$ as $\alpha(h(x)) = 2h(x)$. We also let $\rho = 0.99$ and $\gamma = 10$. Let the timing parameters and c_1, c_2 be calculated by Algorithm 1. We present the system trajectory in Fig. 5b. We observe that the safety property with respect to C is guaranteed using the synthesized safety-critical controller.

8 CONCLUSION

In this paper, we studied the problem of ensuring safety of CPS under malicious cyber attacks. We proposed a reactive restart approach with verifiable safety guarantees for a class of CPS under malicious cyber attacks. The proposed approach restarts the system when the controller crashes following faults or attacks. We presented a hybrid model of the system behavior under malicious attack and reactive restart. We developed sufficient conditions for safety of the hybrid model using a barrier certificate approach. We formulated a sum-of-squares program and developed two approximate solution algorithms to verify the developed sufficient conditions and compute the timing parameters for the CPS. We proposed a data reload strategy for safety verification of CPS whose states need to be estimated using sensor measurements, which reduces the time needed for CPS to re-learn the system state after restart. We developed a quadratic program subject control barrier function constraints to compute the control input at each time during the exploit window to solve the safety-critical control synthesis problem. We proved that the synthesized controller guarantees the safety of the system. We demonstrated the proposed approach using two case studies on a warehouse temperature control system and a two-dimensional non-linear system. We showed that our proposed approach guaranteed the safety property for both case studies.

ACKNOWLEDGEMENTS

We thank Dr. J. Sukarno Mertoguno, Dr. Marco Caccamo, and the anonymous reviewers for all the helpful discussions and comments.

REFERENCES

[1] Fardin Abdi, Chien-Ying Chen, Monowar Hasan, Songran Liu, Sibin Mohan, and Marco Caccamo. 2018. Guaranteed physical security with restart-based design for cyber-physical systems. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. ACM/IEEE, 10–21.

[2] Fardin Abdi, Rohan Tabish, Matthias Rungger, Majid Zamani, and Marco Caccamo. 2017. Application and system-level software fault tolerance through full system restarts. In *ACM/IEEE 8th International Conference on Cyber-Physical Systems (ICCPS)*. ACM/IEEE, 197–206.

[3] Homa Alemzadeh, Daniel Chen, Xiao Li, Thenkurussi Kesavadas, Zbigniew T Kalbarczyk, and Ravishankar K Iyer. 2016. Targeted attacks on teleoperated surgical robots: Dynamic model-based detection and mitigation. In *46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 395–406.

[4] Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A Henzinger, P-H Ho, Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, and Sergio Yovine. 1995. The algorithmic analysis of hybrid systems. *Theoretical Computer Science* 138, 1 (1995), 3–34.

[5] Aaron D Ames, Xiangru Xu, Jessy W Grizzle, and Paulo Tabuada. 2016. Control barrier function based quadratic programs for safety critical systems. *IEEE Trans. Automat. Control* 62, 8 (2016), 3861–3876.

[6] T Arauz, JM Maestre, R Romagnoli, B Sinopoli, and EF Camacho. 2021. A linear programming approach to computing safe sets for software rejuvenation. *IEEE Control Systems Letters* 6 (2021), 1214–1219.

[7] Miguel Arroyo, Hidenori Kobayashi, Simha Sethumadhavan, and Junfeng Yang. 2017. FIRED: frequent inertial resets with diversification for emerging commodity cyber-physical systems. *arXiv preprint arXiv:1702.06595* (2017).

[8] Miguel A Arroyo, M Tarek Ibn Ziad, Hidenori Kobayashi, Junfeng Yang, and Simha Sethumadhavan. 2019. YOLO: frequently resetting cyber-physical systems for security. In *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2019*, Vol. 11009. International Society for Optics and Photonics, 110090P.

[9] Fredrik Björck, Martin Henkel, Janis Stirna, and Jelena Zdravkovic. 2015. Cyber resilience–fundamentals for a definition. In *New Contributions in Information Systems and Technologies*. Springer, 311–316.

[10] Alvaro A Cárdenas, Saurabh Amin, and Shankar Sastry. 2008. Research Challenges for the Security of Control Systems.. In *Proceedings of the 3rd Conference on Hot Topics in Security*, Vol. 5. USENIX Association, 15.

[11] Miguel Castro and Barbara Liskov. 2002. Practical Byzantine fault tolerance and proactive recovery. *ACM Transactions on Computer Systems (TOCS)* 20, 4 (2002), 398–461.

[12] Mo Chen, Qie Hu, Jaime F Fisac, Kene Akametalu, Casey Mackin, and Claire J Tomlin. 2017. Reachability-based safety and goal satisfaction of unmanned aerial platoons on air highways. *Journal of Guidance, Control, and Dynamics* 40, 6 (2017), 1360–1373.

[13] Andrew Clark. 2021. Verification and Synthesis of Control Barrier Functions. *arXiv preprint arXiv:2104.14001* (2021).

[14] Max H Cohen and Calin Belta. 2020. Approximate optimal control for safety-critical systems with control barrier functions. In *59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2062–2067.

[15] Interagency Security Committee. 2015. *Presidential policy directive 21 implementation: An Interagency security committee white paper*. White Paper. Cybersecurity & Infrastructure Security Agency. https://www.cisa.gov/sites/default/files/publications/ISC-PPD-21-Implementation-White-Paper-2015-508.pdf

[16] Lucas Davi, Christopher Liebchen, Ahmad-Reza Sadeghi, Kevin Z Snow, and Fabian Monrose. 2015. Isomeron: Code randomization resilient to (Just-In-Time) return-oriented programming. In *The Network and Distributed System Security (NDSS) Symposium*. The Internet Society, 323–338.

[17] Hamza Fawzi, Paulo Tabuada, and Suhas Diggavi. 2014. Secure estimation and control for cyber-physical systems under adversarial attacks. *IEEE Transactions on Automatic control* 59, 6 (2014), 1454–1467.

[18] Andy Greenberg. 2015. Hackers remotely kill a Jeep on the highway–with me in it. https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/

[19] Inseok Hwang, Sungwan Kim, Youdan Kim, and Chze Eng Seah. 2009. A survey of fault detection, isolation, and reconfiguration methods. *IEEE Transactions on Control Systems Technology* 18, 3 (2009), 636–653.

[20] Radoslav Ivanov, Miroslav Pajic, and Insup Lee. 2016. Attack-resilient sensor fusion for safety-critical cyber-physical systems. *ACM Transactions on Embedded Computing Systems (TECS)* 15, 1 (2016), 1–24.

[21] Hassan K Khalil. 2002. *Nonlinear Systems*. Prentice hall.

[22] John C Knight. 2002. Safety critical systems: challenges and directions. In *24th International Conference on Software Engineering*. IEEE, 547–550.

[23] Fanxin Kong, Meng Xu, James Weimer, Oleg Sokolsky, and Insup Lee. 2018. Cyber-physical system checkpointing and recovery. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. ACM/IEEE, 22–31.

[24] Karl Koscher, Stefan Savage, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Alexei Czeskis, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, and Stefan Savage. 2010. Experimental security analysis of a modern automobile. In *IEEE Symposium on Security and Privacy*. IEEE, 447–462.

[25] M. Robert Lee, J. Michael Assante, and Tim Conway. 2016. Analysis of the cyber attack on the Ukrainian power grid. https://www.eisac.com/cartella/Asset/00006542/TLP_WHITE_E-ISAC_SANS_Ukraine_DUC_6_Modular_ICS_Malware%20Final.pdf?parent=64412.

[26] Anqi Li, Li Wang, Pietro Pierpaoli, and Magnus Egerstedt. 2018. Formally correct composition of coordinated behaviors using control barrier certificates. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 3723–3729.

[27] Mohammad Hossein Manshaei, Quanyan Zhu, Tansu Alpcan, Tamer Başar, and Jean-Pierre Hubaux. 2013. Game theory meets network security and privacy. *ACM Computing Surveys (CSUR)* 45, 3 (2013), 1–39.

[28] Lockheed Martin. [n. d.]. The Cyber Kill Chain. https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html

[29] J Sukarno Mertoguno, Ryan M Craven, Matthew S Mickelson, and David P Koller. 2019. A physics-based strategy for cyber resilience of CPS. In *Autonomous Systems: Sensors, Processing, and Security for Vehicles and Infrastructure 2019*, Vol. 11009. International Society for Optics and Photonics, 110090E.

[30] Yilin Mo and Bruno Sinopoli. 2010. False data injection attacks in control systems. In *Preprints of the 1st workshop on Secure Control Systems*. 1–6.

[31] Miroslav Pajic, Zhihao Jiang, Insup Lee, Oleg Sokolsky, and Rahul Mangharam. 2014. Safety-critical medical device development using the UPP2SF model translation tool. *ACM Transactions on Embedded Computing Systems (TECS)* 13, 4s (2014), 1–26.

[32] Miroslav Pajic, James Weimer, Nicola Bezzo, Paulo Tabuada, Oleg Sokolsky, Insup Lee, and George J Pappas. 2014. Robustness of attack-resilient state estimators. In *ACM/IEEE International Conference on Cyber-Physical Systems (ICCPS)*. ACM/IEEE, 163–174.

[33] Stephen Prajna, Ali Jadbabaie, and George J Pappas. 2007. A framework for worst-case and stochastic safety verification using barrier certificates. *IEEE Trans. Automat. Control* 52, 8 (2007), 1415–1428.

[34] Zengyi Qin, Kaiqing Zhang, Yuxiao Chen, Jingkai Chen, and Chuchu Fan. 2021. Learning safe multi-agent control with decentralized neural barrier certificates. *arXiv preprint arXiv:2101.05436* (2021).

[35] Raffaele Romagnoli, Paul Griffioen, Bruce H Krogh, and Bruno Sinopoli. 2020. Software rejuvenation under persistent attacks in constrained environments. *IFAC-PapersOnLine* 53, 2 (2020), 4088–4094.

[36] Raffaele Romagnoli, Bruce H Krogh, and Bruno Sinopoli. 2019. Design of software rejuvenation for CPS security using invariant sets. In *2019 American Control Conference (ACC)*. IEEE, 3740–3745.

[37] Yasser Shoukry and Paulo Tabuada. 2015. Event-triggered state observers for sparse sensor noise/attacks. *IEEE Trans. Automat. Control* 61, 8 (2015), 2079–2091.

[38] Joelle Skaf and Stephen P Boyd. 2010. Design of affine controllers via convex optimization. *IEEE Trans. Automat. Control* 55, 11 (2010), 2476–2487.

[39] Julia E Sullivan and Dmitriy Kamensky. 2017. How cyber-attacks in Ukraine show the vulnerability of the US power grid. *The Electricity Journal* 30, 3 (2017), 30–35.

[40] Siri Hofstad Trapnes. 2013. *Optimal temperature control of rooms for minimum energy cost*. Master's thesis. Institutt for Kjemisk Prosessteknologi.

[41] Paulo Esteves Veríssimo, Nuno Ferreira Neves, and Miguel Pupo Correia. 2003. Intrusion-tolerant architectures: Concepts and design. In *Architecting Dependable Systems*. Springer, 3–36.

[42] Qiuye Wang, Mingshuai Chen, Bai Xue, Naijun Zhan, and Joost-Pieter Katoen. 2021. Synthesizing Invariant Barrier Certificates via Difference-of-Convex Programming. *arXiv preprint arXiv:2105.14311* (2021).

[43] Lin Zhang, Pengyuan Lu, Fanxin Kong, Xin Chen, Oleg Sokolsky, and Insup Lee. 2021. Real-time Attack-recovery for Cyber-physical Systems Using Linear-quadratic Regulator. *ACM Transactions on Embedded Computing Systems (TECS)* 20, 5s (2021), 1–24.

[44] Quanyan Zhu and Tamer Başar. 2015. Game-theoretic methods for robustness, security, and resilience of cyberphysical control systems: games-in-games principle for optimal cross-layer resilient control systems. *IEEE Control Systems Magazine* 35, 1 (2015), 46–65.

A APPENDIX

In this appendix, we first introduce some preliminary background. We then present the technical proofs that are omitted in the paper.

A.1 Preliminaries

This subsection presents preliminary background. A continuous function $\alpha : [-b, a) \to \mathbb{R}$ is an extended class $\mathcal{K}$ function if $\alpha(\cdot)$ is strictly increasing and $\alpha(0) = 0$ for some $a, b > 0$. We also denote the set of real numbers and the set of non-negative real numbers as $\mathbb{R}$ and $\mathbb{R}_{\geq 0}$, respectively. A multivariate polynomial $p(x)$ is a sum-of-squares (SOS) polynomial if there exists a set of polynomials $k_1(x), \ldots, K_N(x)$ such that $p(x) = \sum_{i=1}^{N} k_i(x)^2$. If $p(x)$ is an SOS polynomial, we have that $p(x) \geq 0$.

Control barrier functions (CBFs) have been used to guarantee forward invariance of system (1). We consider two types of CBF in

this work, named zeroing CBF (ZCBF) and finite time convergence CBF (FCBF).

Definition A.1 (ZCBF [5]). Consider a dynamical system (1) and a continuously differentiable function $h : \mathcal{X} \to \mathbb{R}$. If there exists an extended class $\mathcal{K}$ function $\alpha(\cdot)$ such that for all $x \in \mathcal{X}$ the following inequality holds:

$$\sup_{u \in \mathcal{U}} \left\{ \frac{\partial h}{\partial x}(x) f(x) + \frac{\partial h}{\partial x}(x) g(x) u + \alpha(h(x)) \right\} \geq 0, \tag{14}$$

then function h is a ZCBF.

Definition A.2 (FCBF [26]). Consider a dynamical system (1) and a continuously differentiable function $h : \mathcal{X} \to \mathbb{R}$. If there exist parameter $\gamma > 0$ and $\rho \in [0, 1)$ such that for all $x \in \mathcal{X}$ the following inequality holds:

$$\sup_{u \in \mathcal{U}} \left\{ \frac{\partial h}{\partial x}(x) f(x) + \frac{\partial h}{\partial x}(x) g(x) u + \gamma \cdot \operatorname{sgn}(h(x)) |h(x)|^{\rho} \right\} \geq 0, \tag{15}$$

then function h is an FCBF.

The sets of control inputs satisfying Eqn. (14) and (15) provide the following guarantees, respectively.

Lemma A.3 ([5]). *Given a dynamical system (1) and a set $C = \{x : h(x) \geq 0\}$, if h is a ZCBF defined on $\mathcal{X}$, then the control signals satisfying Eqn. (14) guarantee that C is forward invariant.*

Lemma A.4 ([26]). *Consider a dynamical system (1) and a set $C = \{x : h(x) \geq 0\}$. If h is an FCBF defined on $\mathcal{X}$, then the control signals satisfying Eqn. (15) guarantees that there exists some finite $T \in \left[0, \frac{|h(x(0))|^{1-\rho}}{\gamma(1-\rho)}\right]$ such that $x(T) \in C$ for any initial state $x(0) \in \mathcal{X}$. Moreover, the system trajectory $x(t') \in C$ for all $t' \geq T$.*

We next introduce background on hybrid system. A hybrid system is defined as follows [4].

Definition A.5. A hybrid system is a tuple $H = (\mathcal{X}, \mathcal{L}, \mathcal{S}, \mathcal{S}_0, Inv, \mathcal{F}, \Sigma)$ with each element being defined as

- $\mathcal{X} \subseteq \mathbb{R}^n$ is the continuous system state space.
- $\mathcal{L}$ is a finite set of discrete locations.
- $\mathcal{S} = \mathcal{X} \times \mathcal{L}$ is the state space of hybrid system H, and $\mathcal{S}_0 \subseteq \mathcal{S}$ is the set of initial states.
- $Inv : \mathcal{L} \to 2^{\mathcal{X}}$ is the invariant that maps from the set of locations to the power set of $\mathcal{X}$. That is, $Inv(l) \subseteq \mathcal{X}$ specifies the set of possible continuous states when the system is at location l.
- $\mathcal{F}$ is the set of vector fields. For each $f \in \mathcal{F}$, the continuous system state evolves as $\dot{x} = f(x, l)$, where $\dot{x}$ is the time derivative of continuous state x.
- $\Sigma \subseteq \mathcal{S} \times \mathcal{S}$ is the set of transitions between the states of the hybrid system. A transition $\sigma = ((x, l), (x', l'))$ models that the hybrid system state transitions from (x, l) to (x', l').

Consider a hybrid system H as defined in Definition A.5. Let $l \neq l'$ be two discrete locations. Then a guard set $\mathcal{G}(l, l')$ is defined as $\mathcal{G}(l, l') = \{x \in \mathcal{X} : ((x, l), (x', l')) \in \Sigma\}$, which models the set of continuous states starting from which the system can take transition from location l to l'. We define a set valued function $\mathcal{R}(l, l') : x \to \{x' \in \mathcal{X} : ((x, l), (x', l')) \in \Sigma\}$, which captures the set of continuous states that can be reached from $\mathcal{G}(l, l')$ via discrete transition l to l'. We also let $Init(l) = \{x \in \mathcal{X} : (x, l) \in \mathcal{S}_0\}$ and $Unsafe(l) = \{x \in \mathcal{X} : (x, l) \in \mathcal{S}_u\}$.

The safety of hybrid system H is given as follows.

Definition A.6 (Safety of Hybrid System [33]). Consider a hybrid system H and an unsafe set $\mathcal{S}_u \subseteq \mathcal{S}$. The safety property of H holds if there exist no time $T \geq 0$ and a finite sequence of times $0 \leq t_1 \leq \ldots \leq t_N \leq T$ such that the trajectory $(x, l) : [0, T] \to \mathcal{S}$ satisfying $(x(0), l(0)) \in \mathcal{S}_0$, $x(t) \in Inv(l(t))$ for all $t \in [0, T]$, and $(x(t), l(t)) \in \mathcal{S}_u$.

The safety given in Definition A.6 for hybrid system H is certified by a collection of barrier certificates $\{B_l(x)\}$ as follows.

Lemma A.7 ([33]). *Consider a hybrid system H as defined in Definition A.5 and an unsafe set $\mathcal{S}_u \subseteq \mathcal{S}$. Suppose there exists a collection of continuously differentiable functions, denoted as $\{B_l(x) : l \in \mathcal{L}\}$, such that for all $l \neq l'$ the following relations hold:*

$$B_l(x) \leq 0, \ \forall x \in Init(l), \tag{16a}$$

$$B_l(x) > 0, \ \forall x \in Unsafe(l), \tag{16b}$$

$$\frac{\partial B_l}{\partial x}(x) f_l(x) < 0, \ \forall x \in Inv(l) \text{ s.t. } B_l(x) = 0 \tag{16c}$$

$$B_{l'}(x') \leq 0, \ \forall x' \in \mathcal{R}(l, l')(x), \ \forall x \in \mathcal{G}(l, l') \text{ s.t. } B_l(x) \leq 0 \tag{16d}$$

then the safety of hybrid system H is satisfied.

A.2 Technical Proofs

In this subsection, we provide the proofs of Theorem 4.1, Proposition 4.2, Lemma 4.3, and Proposition 6.1.

Proof of Theorem 4.1. We prove the theorem by first characterizing the hybrid system H we constructed in Section 4. We will show that $Inv(unsafe) = C_1$ and $Inv(init) = Inv(safe) = C$ when $\mathcal{G}(safe, unsafe) \subseteq C_2$ and Eqn. (3) hold. We then prove the safety property by showing that $-h(x)$ is a barrier certificate satisfying Lemma A.7 for hybrid system H, and hence safety is satisfied.

Suppose $\mathcal{G}(safe, unsafe) \subseteq C_2$. We let $x(t) \in C_2$ and the system be in location $l = unsafe$. Thus $h(x(t)) \geq c_1 + c_2$. Suppose the next discrete transition $((x, unsafe), (x, init))$ happens at time $t' \geq t$. By Eqn. (3d) and integrating $\dot{h}(x)$ over time, we have that $h(x(t')) \geq c_1 + c_2 - \frac{c_2}{\phi}(t' - t)$. When $t' \in [t, t + \phi]$, we have that $h(x(t')) \geq c_1 \geq 0$ and thus $x(t') \in C_1$ if Eqn. (3d) holds. This also implies that $\mathcal{G}(unsafe, init) \subseteq C_1$ and $Inv(unsafe) \subseteq C_1$.

Consider that location transition $((x(t), unsafe), (x(t), init))$ happens at time t. Since $\mathcal{G}(unsafe, init) \subseteq C_1$, we have that $x(t) \in C_1$ and $h(x(t)) \geq c_1$. By Eqn. (3c) and integrating $\dot{h}(x)$ over $[t, t + \eta]$, we have that $h(x(t')) \geq c_1 - \frac{c_1}{\eta}(t' - t) \geq 0$ for all $t' \in [t, t + \eta]$. Therefore, $h(x(t')) \in C$. This indicates that $\mathcal{G}((x, init), (x, safe)) \subseteq C$ and $Inv(init) \subseteq C$.

Consider that transition $((x(t), init), (x(t), safe))$ happens at time t. Since $\mathcal{G}((x, init), (x, safe)) \subseteq C$, we have that $h(x(t)) \geq 0$. We then divide our discussion into two cases. We first consider $0 \leq h(x(t)) \leq c_1 + c_2$. If Eqn. (3a) holds, then integrating the left-hand side of Eqn. (3a) over $t' \in [t, t + \tau']$ with $\tau' \geq \tau$ yields that $h(x(t')) \geq 0 + \frac{c_1 + c_2}{\tau}(t' - t) \geq c_1 + c_2$ for all $t' \in [t, t + \tau']$. Using the definition that $h_2(x) = \{x : h(x) \geq c_1 + c_2\}$, we have that $x(t + \tau') \in C_2 \subset C$. Moreover, we have that if the length of

the exploit window is at least τ, the system trajectory will reach C_2 and remains in it before transition $((x, safe), (x, unsafe))$ occurs using Lemma A.3. We next consider the second case where $h(x(t)) \geq c_1 + c_2$. If Eqn. (3b) holds, we have that $h(x(t')) \geq c_1 + c_2$ by Lemma A.3 for all $t' \in [t, t + \tau]$. Summarizing the above two cases, we have that $\mathcal{G}(safe, unsafe) \subseteq C_2$ holds. Moreover, we have that $Inv(safe) \subseteq C$.

We finally show that $-h(x)$ is a barrier certificate satisfying Lemma A.7 for hybrid system H. Since $x(0) \in C_2 \subset C$, we have that $-h(x(0)) \leq 0$, and thus condition (16a) is satisfied. When $x \notin C$, we have that $-h(x) > 0$, implying that Eqn. (16b) is met. By our previous analysis, we have that $Inv(unsafe) \subseteq C_1 \subset C$ and $Inv(init) \subseteq C$. Thus $h(x) = 0$ can only hold after transition $((x, init), (x, safe))$ takes place where $Inv(safe) \subseteq C$. Using Eqn. (3a) and the relation $h_2(x) = h(x) - c_1 - c_2$, we have that

$$\frac{\partial(-h)}{\partial x}(x) f_{cl}^{safe}(x; K, b) = \frac{\partial(-h_2)}{\partial x}(x) f_{cl}^{safe}(x; K, b) \leq -\frac{c_1 + c_2}{\tau}$$

holds for all $x \in Inv(safe)$ such that $h(x) = 0$. Therefore, condition (16c) holds. Using the definition of Σ, we have that the continuous state x does not have any jump when location transition occurs, implying that Eqn. (16d) holds. Therefore, $-h$ is a barrier certificate satisfying Lemma A.7, and thus hybrid system H is safe with respect C. Hence, we have that system (1) is safe with respect to C. □

Proof of Proposition 4.2. When $x \in C \setminus C_2$, we have that $h(x) \geq 0$ and $h_2(x) < 0$. Since $l(x)$ is an SOS polynomial, we have that $-l(x)h(x)h_2(x) \geq 0$ for all $x \in C \setminus C_2$. If the expression in Eqn. (4a) is an SOS, we have that

$$\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x; K, b) - z_3 \geq -l(x)h(x)h_2(x) \geq 0, \ \forall x \in C \setminus C_2.$$

We thus have that if the expression in Eqn. (4a) is an SOS and $z_3 = \frac{c_1 + c_2}{\tau}$, then Eqn. (3a) holds.

When $x \in C_2$, we have that $h_2(x) \geq 0$. Since $r(x)$ is an SOS polynomial, $r(x)h_2(x) \geq 0$ holds for all $x \in C_2$. If Eqn. (4b) is an SOS, we then have that $\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x; K, b) + \alpha(h_2(x)) - r(x)h_2(x) \geq 0$, which implies that $\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x; K, b) \geq -\alpha(h_2(x))$ for all $x \in C_2$. Therefore, Eqn. (3b) holds when Eqn. (4b) is an SOS.

When $x \in C$, $h(x) \geq 0$ holds by the definition of C. Since $p(x)$ is an SOS polynomial, we have that $p(x)h(x) \geq 0$ for all $x \in C$. If Eqn. (4c) is an SOS, then $\frac{\partial h}{\partial x} f_{cl}^{init}(x) \geq -z_1 = \frac{c_1}{\eta}$, indicating that Eqn. (3c) holds.

When $x \in C_1$ and $u \in \mathcal{U}$, $h_1(x)v(x) \geq 0$. Since $q(x, u)$ is an SOS polynomial, we have that $q(x, u)h_1(x)v(u) \geq 0$ for all $(x, u) \in C_1 \times \mathcal{U}$. If Eqn. (4d) is an SOS, then $\frac{\partial h}{\partial x} f_{cl}^{unsafe}(x, u) \geq -z_2 = \frac{c_2}{\phi}$, indicating that Eqn. (3d) holds.

Combining the arguments above completes the proof. □

Proof of Lemma 4.3. If the expressions in Eqn. (6) are SOS, we then have that

$$\frac{\partial h_2}{\partial x}(x) f_{cl}^{safe}(x; K, b) - z_3 \geq l(x)h(x), \tag{17a}$$

$$\frac{\partial h}{\partial x}(x) f_{cl}^{init}(x) + z_1 \geq p(x)h(x), \tag{17b}$$

$$\frac{\partial h}{\partial x}(x) f_{cl}^{unsafe}(x, u) + z_2 \geq q(x, u)h(x)v(u). \tag{17c}$$

When $x \in C$ and $u \in \mathcal{U}$, we have that $h(x) \geq 0$ and $v(u) \geq 0$. Also note that $l(x)$, $p(x)$, and $q(x, u)$ are SOS polynomials. We thus have that if the expressions in Eqn. (6) are SOS, then Eqn. (8) holds. □

Proof of Proposition 6.1. We first construct a hybrid system $H = (\mathcal{Y}, \mathcal{L}, \mathcal{S}, \mathcal{S}_0, Inv, \mathcal{F}, \Sigma)$, where $\mathcal{Y}$ is the set of continuous states y, $\mathcal{L} = \{safe, unsafe, init\}$, and $\mathcal{F} = \{f_{cl}^{safe}(y; K, b), f_{cl}^{init}(y), f_{cl}^{unsafe}(y, u)\}$. When Eqn. (11a) holds, we have that $y \in \{y : d(y) \geq 0\}$ if the system is in discrete location $l = safe$. When Eqn. (11c) to (11e) hold, we have that $Inv(unsafe) = (C_1, \mathcal{B}(C_1))$, $Inv(init) = (C, \mathcal{B}(C))$, and $Inv(safe) = (C, \mathcal{B}(C))$, where $\mathcal{B}(A)$ is defined as $\mathcal{B}(A) = \{\hat{x} : c_3 - \|x - \hat{x}\| \geq 0, \ \forall x \in A\}$ for some $A \subseteq \mathcal{X}$. Finally, we can verify that $-h(x)$ is a barrier certificate satisfying Lemma A.7 using similar approach in Theorem 4.1, which implies that the safety property with respect to C holds. □