Skip to Secure: Securing Cyber-Physical Control Loops with Intentionally Skipped Executions

Sunandan Adhikary
Indian Institute of Technology
Kharagpur, India
sunandana@iitkgp.ac.in

Ipsita Koley
Indian Institute of Technology
Kharagpur, India
ipsitakoley@iitkgp.ac.in

Saurav Kumar Ghosh*
Indian Institute of Technology
Kharagpur, India
saurav.kumar.ghosh@cse.iitkgp.ernet.in

Sumana Ghosh
Technical University of Munich
Germany
sumana.ghosh@tum.de

Soumyajit Dey
Indian Institute of Technology
Kharagpur, India
soumya@cse.iitkgp.ac.in

Debdeep Mukhopadhyay
Indian Institute of Technology
Kharagpur, India
debdeep@cse.iitkgp.ac.in

ABSTRACT

We consider the problem of provably securing a given control loop implementation in the presence of adversarial interventions on data exchange between plant and controller. Such interventions can be thwarted using continuously operating monitoring systems and also cryptographic techniques, both of which consume network and computational resources. We provide a principled approach for intentional skipping of control loop executions which may qualify as a useful control-theoretic countermeasure against stealthy attacks which violate message integrity and authenticity. As can be seen, such an approach helps in lowering the resource consumption caused by monitoring/cryptographic security measures.

CCS CONCEPTS

• **Security and privacy** → **Intrusion detection systems**; • **Computer systems organization** → *Embedded and cyber-physical systems.*

KEYWORDS

CPS, Intrusion Detection System, False Data Injection, Control Performance, Formal Verification, CAN, Automotive Security

ACM Reference Format:
Sunandan Adhikary, Ipsita Koley, Saurav Kumar Ghosh, Sumana Ghosh, Soumyajit Dey, and Debdeep Mukhopadhyay. 2020. Skip to Secure: Securing Cyber-Physical Control Loops with Intentionally Skipped Executions. In *2020 Joint Workshop on CPS&IoT Security and Privacy (CPSIOTSEC'20), November 9, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3411498.3419966

*Also with Security Verification and Validation (ESY), Robert Bosch Engineering & Business Solutions PVT LTD.

CPSIOTSEC'20, November 9, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8087-4/20/11…$15.00
https://doi.org/10.1145/3411498.3419966

1 INTRODUCTION

The proliferation of network connectivity has increased the application domain for Cyber-Physical Systems (CPS) in today's connected world. However, increased connectivity manifests security vulnerability in terms of an increased number of possible attack surfaces for such systems. Recent results have established that network-based *Man-in-the-Middle* type attacks, like False Data Injection (FDI), on CPS are quite capable of disturbing closed-loop stability as well as degrading the control performance of such systems [24]. In such an attack, an adversary injects false data in the communication medium between the plant and the controller intending to drive the system to an unsafe state by changing the set point of the system.

To detect such attacks, the most commonly used control-theoretic countermeasures are threshold-based anomaly detectors that generate an alarm if the state estimation error crosses the threshold over a single or multiple control loop iterations. Though such control-theoretic primitives can limit the amount of false data injection, stealthy attacks can drive the system to an unsafe state [23] as long as the attack length is not bounded. Alternatively, for safety-critical systems, the messages exchanged between the plant and the controller can be authenticated with cryptographic security primitives like Message Authentication Codes (MACs) [20] to ensure that no false data can be injected at all.

However, the use of MAC in CPSs is often limited by the available compute resources in the on-board platforms, e.g. on a 96 MHz ARM Cortex-M3-based Electronic Control Unit (ECU), a control law computation takes approximately $5\mu s$ while a 128-bit MAC computation for a single message takes $100\mu s$ [13]. Hence, sporadic MAC verification-based Intrusion Detection Systems (IDS) has been proposed for resource constrained CPSs[13]. In such IDSs, MAC is periodically verified once in a fixed number of control loop iterations to avoid the introduction of unbounded state estimation error in the system. This guarantees that the system state variables are strictly operational within a *safe* operating region. Hence the *activation interval* of the IDS is the *key parameter* to guarantee system safety under FDI attacks. The primary objective of this work is *to improve the sporadicity of such IDSs even further by increasing this activation interval so that its computational, as well as communication footprint can be reduced without compromising the security.*

A sporadic IDS (shown in Fig. 1) can be specified by a pair of intervals (n_{up}, n_{down}) which denotes that the IDS is active for

n_{up} consecutive control iterations, inactive for n_{down} consecutive control iterations, and this behavior repeats in a cycle. Consider that there exists an *initial region* C for a control system which is composed of the initial range of plant state values. Starting from C, let us consider that the preferred operating region for the system is given by an *inner safety region* $C_1 (C \subseteq C_1)$ in the absence of any external attacks. The safety guarantee offered by a sporadic IDS is based on the existence of an *outer safety region* C_2 ($C_1 \subset C_2$) which meets the safety requirements of the system, but may not be a preferable operating region due to unsatisfactory control performance. The IDS parameters, n_{up} and n_{down} are defined as,

$x[k] \in C_1 \Rightarrow \forall i \leq n_{down}\ x[k+i] \in C_2$ when IDS is off and

$x[k] \in C_2 \Rightarrow \forall i \geq n_{up}\ x[k+i] \in C_1$ when IDS is on

where $x[k]$ is the plant state at any time instant k. If an IDS is not available for n_{down} consecutive control cycles, stealthy FDI attacks are possible. To keep the system *secure* against stealthy FDI attacks, this IDS *activation interval* n_{down} should be small enough to ensure that such attacks do not bring the system outside the outer safety region C_2. When the IDS is active for n_{up} consecutive control iterations, no FDI is possible. The period n_{up} needs to be large enough to ensure that the system is steered back inside C_1 starting from anywhere $\in C_2$, thus nullifying the effect of FDIs during n_{down}. The sporadic MAC verification-based IDS proposed in [13] has $n_{up} = 1$ and n_{down} is the derived periodicity of their MAC computation.

In the present work, we adapt the well-known concept of *non-uniform control execution* to improve the down-time of such sporadic IDSs even further while guaranteeing the same level of security. Such a control policy *intentionally skips* some of the control executions without compromising the desired performance guarantee [5, 15, 22]. This reduces the load on the computation/communication resources and also increases the resilience of a system against FDI. Because the system is unaffected by any malicious data injected by the attacker into the communication channel at those sampling instants when the control executions are skipped. The overall contributions of the present work can be summarized as follows:

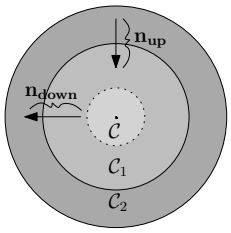


Figure 1: Sporadic IDS

(1) This is the first work motivating the use of intentional control execution skips as a formally verified control-theoretic security measure against FDI.
(2) We propose an SMT-based framework to formally analyze the attack resilience of skipped control executions against FDI.
(3) We leverage this framework to further increase the activation interval of sporadic IDS (i.e. reduce resource usage) with a formal security guarantee.
(4) We establish the usefulness of our approach by considering automotive system examples where our sporadic IDS results in a significant reduction in resource consumption.

In the next section, we formalize our control theoretic system model and the FDI threat model.

2 SYSTEM AND ATTACK MODELING

System Modeling: We represent physical plant P as a linear discrete-time invariant (LTI) system as given by $x[k+1] = Ax[k] + Bu[k]$ where $x[k]$ and $u[k]$ denotes the plant state vector and the control input at the k-th iteration. The output measurement vector in the k-th iteration $y[k]$ is given by $y[k] = Cx[k]$. It is used to estimate the plant state vector in the $(k+1)$-th iteration given by $\hat{x}[k+1] = A\hat{x}[k] + Bu[k] + L(y[k] - C\hat{x}[k])$ where L is the Kalman Filter gain. The control input $u[k+1]$ for the $(k+1)$-th iteration is calculated as $u[k+1] = K\hat{x}[k+1]$ where K is the Linear Quadratic Regulator (LQR) based optimal control gain.

To analyze the effect of execution skips on the closed plant-control loop (P, K), we define $X[k] = [x^{\mathsf{T}}[k]\ \hat{x}^{\mathsf{T}}[k]\ u[k]^{\mathsf{T}}]^{\mathsf{T}}$ as state vector for the augmented system comprising both the plant and estimator states along with control inputs. If the control execution is intentionally skipped in the k-th iteration, then 1) the measurement vector $y[k]$ is not communicated to the controller and 2) in the $(k+1)$-th iteration the last control input $u[k]$ is repeated i.e., $u[k+1] = u[k]$. The evolution of the augmented system under skipped execution is given by $X[k+1] = A_0\ X[k]$, while for periodic execution A_0 is replaced by A_1. Here A_0 and A_1 are given by

$$\begin{bmatrix} A & 0 & B \\ LC & A-LC-BK & 0 \\ 0 & 0 & I \end{bmatrix}, \begin{bmatrix} A & 0 & B \\ LC & A-LC-BK & 0 \\ KLC & KA-KLC-KBK & 0 \end{bmatrix}$$

respectively. An l-length *control skipping pattern* for a given control loop (P, K), is an l length sequence $\rho \in \{0,1\}^l$ which can be used to define an infinite length control schedule $\pi = \rho^{\omega}$, repeating with period l, i.e., $\pi[k] = \pi[k+l] = \rho[k\%l], \forall k \in \mathbb{Z}^+$. The evolution of the closed-loop system according to a control skipping pattern can be exemplified as: for $\rho = 110010$, we have, $X[6] = A_1X[5] = A_1A_1A_0X[3] = \ldots = A_1A_1A_0A_0A_1A_0X[0]$.

Control Performance Metrics: The control performance metric that we use in this work is *settling time*, i.e. the time needed by the system output to fall and stay around the reference value (e.g., within 2 % error band). The control strategy is designed to always meet the settling time requirement. A significant amount of work exists on achieving desired control design and performance in the presence of execution skips [5, 15, 22]. Given the settling time requirement, T_s, we follow Theorem 4.1 of [5] to calculate the *minimum execution rate*, r_{min}. This means, to maintain T_s, the controller has to be executed at least $\lceil l \times r_{min} \rceil$ times in l-length consecutive control samples, i.e., in an l-length control skipping pattern, ρ, there has to be at least $\lceil l \times r_{min} \rceil$ number of '1's.

Control-theoretic countermeasure: Following existing detection techniques [7], we assume that our system has a threshold-based intrusion detector to prevent FDI attacks as shown in Fig. 2. The threshold-based detector flags an attack whenever the residue $r[k] = y[k] - C\hat{x}[k]$ (i.e., the estimation error) surpasses the constant detector threshold Th i.e., $||r[k]|| > Th$, ($||.||$ denotes vector 2-norm). This generalizes any deterministic or probabilistic (e.g., χ^2-based detection) state-of-the-art threshold-based detectors [13].

Attack Modeling: We assume that the attacker has full knowledge of the system dynamics and threshold-based detectors present in the system. The attacker can compromise the plant-controller communication medium (Ref. Fig. 2) to inject false data in the following ways **(i)** tamper the control input sent to the actuator resulting

in $\tilde{u}[k] = u[k] + \triangle u[k]$ and **(ii)** provide false sensor measurements to the controller, denoted by $\tilde{y}[k] = y[k] + \triangle y[k]$. Here, $\triangle y[k]$ and $\triangle u[k]$ are bounded by the system supported sensor range and actuator saturation limit. For the attacker to remain stealthy, the FDI is also subject to the constraint that the residue ($r[k] = \tilde{y}[k] - C\hat{x}[k]$) must remain under the threshold of the attack detector. *The goal of the attacker* is to drive the operating point of the system to an unsafe state $x \notin C_2$ while remaining stealthy. The FDI in a single iteration is expressed as $\mathcal{A}[k] = [\triangle u^\top[k] \ \triangle y^\top[k]]^\top$. An attack vector of length d can be defined as $\mathcal{A}_d = [\mathcal{A}[1]\mathcal{A}[2] \cdots \mathcal{A}[d]]$. We can express the evolution of the plant state under FDI as, $\tilde{X}[k+1] = A_1 \tilde{X}[k] + B_1 \ \mathcal{A}[k]$ where, $B_1^\top = \begin{bmatrix} 0 & L^\top & L^\top K^\top \\ 0 & 0 & I \end{bmatrix}$. $B_0^\top = \begin{bmatrix} 0 & L^\top & 0 \\ 0 & 0 & 0 \end{bmatrix}$, replaces $B_1^\top$ in the presence of *control execution skip*, denoting less effect of FDI during skips. So an attack vector $\mathcal{A}_d$ launched on a protected control system at k-th iteration is deemed *i) stealthy* if $||r[i]|| \leq Th, \forall i \in [k+1, k+d+n_{up}]$ (n_{up} is the up-time of the IDS), and *ii) successful* if $\exists j \in [k+1, k+d+n_{up}]$ such that, $x[j] \notin C_2$, i.e., it violates the safety criteria of the system in any iteration. Note that we define this *stealthiness* and *success* of an attack of length d over a window of $d + n_{up}$ control samples because, **(i)** the threshold-based detector is always active, **(ii)** an attack of d-iterations can drive the system to an unsafe state even after the attack is over, and **(iii)** it also ensures to bring the system back to C_1 within n_{up} iterations canceling the effect of $\mathcal{A}_d$. We further assume that the attacker can always launch a successful and stealthy attack, as long as it exists, irrespective of the control skipping pattern followed by the system.

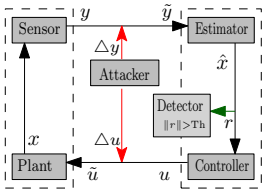


Figure 2: FDI attack

3 PROPOSED METHODOLOGY

Given a closed-loop system that runs following a fully periodic execution pattern and is protected with a threshold-based detector and a sporadic IDS, our objective is to improve resource-awareness of an existing sporadic IDS even further without compromising the system performance and security. To achieve this, **(i)** we first formally analyze the security level of the existing IDS against FDI (in lines of [10]) and **(ii)** we derive the most attack resilient execution patterns that satisfy the desired performance criteria of the system. Using this set of most attack resilient patterns we can reduce resource consumption of the existing sporadic IDS while ensuring the same level of security.

1. Attack Vector Synthesis : We design the function AttVecSyn() (in Algo. 1) to synthesize a *successful* and *stealthy* attack vector of length d (if it exists) for a system with a threshold-based detector of threshold Th under periodic or skipped execution. Given the IDS up-time n_{up} for the closed loop system (P, K), this function captures the system evolution starting from any initial state $x[0] \in C_1$, for $d + n_{up}$ closed control loop iterations (lines 2-11). In every k-th ($k \in [0, d + n_{up}]$) iteration of the function, we introduce two non-deterministic variables $\triangle u[k]$ and $\triangle y[k]$ to model the actuation and measurement error introduced by the adversary

Algorithm 1 Attack Vector Synthesis for Pattern-based Execution

Require: Attack length: d, pattern: ρ, IDS up-time: n_{up}, detector threshold: Th, inner safety region: C_1, outer safety region: C_2
Ensure: Attack vector $\mathcal{A}_d$ of length d (if it exists, otherwise NULL)

1: **function** AttVecSyn(d, ρ, n_{up}, Th)
2: $x[0] \in C_1$; $\hat{x}[0] \leftarrow 0$; $u[0] \leftarrow K\hat{x}[0] \leftarrow 0$; $y[0] \leftarrow Cx[0]$;
3: $r[0] \leftarrow y[0] - C\hat{x}[0]$; $\tilde{u}[0] \leftarrow u[0]$; $\tilde{y}[0] \leftarrow y[0]$;
4: **for** $k = 1$ to $d + n_{up}$ **do**
5: $x[k] \leftarrow Ax[k-1] + B\tilde{u}[k-1]$;
6: $\hat{x}[k] \leftarrow A\hat{x}[k-1] + Bu[k-1] + Lr[k-1]$;
7: **if** $k \leq d$ **then** $\triangle u[k] \leftarrow$ **nondet()**; $\triangle y[k] \leftarrow$ **nondet()**;
8: **else** $\triangle u[k] \leftarrow 0$; $\triangle y[k] \leftarrow 0$;
9: **if** $\rho[k] = 1$ **then** $u[k] \leftarrow K\hat{x}[k]$; $\tilde{u}[k] \leftarrow u[k] + \triangle u[k]$;
10: **else** $u[k] \leftarrow u[k-1]$; $\tilde{u}[k] \leftarrow \tilde{u}[k-1]$; ▷ Skip Execution
11: $\tilde{y}[k] \leftarrow Cx[k] + \triangle y[k]$; $r[k] \leftarrow \tilde{y}[k] - C\hat{x}[k]$;
12: $\Phi \leftarrow$**assert**$((||r[1]|| \leq Th \wedge .. ||r[d+n_{up}]|| \leq Th) \wedge (x[1] \notin C_2 \vee .. \vee x[d+n_{up}] \notin C_2)$);
13: **if** Φ is $unsatisfiable$ **then return** NULL;
14: **else return** $\mathcal{A}_d \leftarrow \begin{bmatrix} \triangle u[1] & \cdots & \triangle u[d] \\ \triangle y[1] & \cdots & \triangle y[d] \end{bmatrix}$;

(line 7). We bound the length of the attack to d by setting these variables to zero $\forall k \in (d, d+n_{up}]$ (line 8). For the attacker to remain stealthy, the residue $r[k]$ must not cross the detector threshold Th i.e., $||r[k]|| < Th$ in any k-th iteration (line 12). As explained in Sec 2, when the k-th control execution is skipped (i.e. $\rho[k] = 0$), $x[k], r[k], y[k]$ are calculated as usual (line 5), but $u[k], \tilde{u}[k]$ are updated using the last calculated $u[k-1], \tilde{u}[k-1]$ (line 10). The safety criterion of the system defined over the plant state values in each iteration k is given by $x[k] \in C_2$ (line 12). So to verify if there exists a *successful* attack of length d, that is *stealthy* over $d + n_{up}$ iterations (i.e., further activation of IDS) we formulate an assertion Φ (line 12) and input it to the SMT solver z3. Algo. 1 returns a successful attack vector $\mathcal{A}_d$ of length d if the assertion (Φ) is satisfiable. Otherwise, it returns *NULL* which guarantees that no *successful and stealthy* attack of length d exists.

The minimum length d_{min}, for which Algo. 1 finds a successful and stealthy attack vector is termed as the *minimum attack-length*. It denotes the *security level* of a system. To prevent a successful attack, the IDS down-time (maximum allowable attack-length) must be less than d_{min} i.e. $n_{down} = d_{min} - 1$ (Ref. MinAttLen() in Algo. 2).

2. Attack Resilient Pattern Synthesis: We now present our formal methodology towards deriving the set of most attack resilient *control skipping patterns* in Algo. 2. For a closed-loop system (P, K) with **(i)** minimum execution rate r_{min} derived from desired settling time specifications for the system (Ref. Sec. 2), **(ii)** inner and outer safety regions given by C_1 and C_2 respectively, **(iii)** threshold Th of the existing threshold-based detector, **(iv)** the existing sporadic IDS specifications (with periodic execution pattern $\rho^* = 1$) i.e. *minimum attack length* $d_{min}^{\rho^*}$ and IDS *up-time* $n_{up}^{\rho^*}$, and **(v)** a set of patterns $\mathcal{P} = \{\rho | \rho \in \{0,1\}^n, n > 0\}$ generated by skipping one or more control executions (Ref. Sec. 2); Algo. 2 prunes $\mathcal{P}$ such that each pattern $\rho \in \mathcal{P}$ having a sporadic IDS with up-time n_{up}^{ρ} and minimum attack length d_{min}^{ρ}, satisfies the following conditions. **(i)** The rate of '1's in ρ should be at least r_{min} [5] so that any control schedule (ρ^ω) designed by cyclically

repeating ρ will abide by the desired performance requirement. **(ii)** Starting from anywhere inside the outer safety region C_2, the system, following ρ^ω will reach a given inner safety region C_1 (Fig. 1) with no stealthy FDI attacks as guaranteed by the IDS within n^ρ_{up} iterations. **(iii)** The ratio $(n^\rho_{down}/n^\rho_{up})$ for ρ is maximum among all patterns $\mathcal{P}$ thereby ensuring minimum IDS execution rate $(n^\rho_{up}/(n^\rho_{down}+n^\rho_{up}))$ where the maximum IDS down-time n^ρ_{down} for ρ is given by $n^\rho_{down} = (d^\rho_{min} - 1)$. We use the function

Algorithm 2 Sporadic IDS Design

Require: $\mathcal{P}$, closed-loop system (P, K), r_{min}, $d^{\rho^*}_{min}$, $n^{\rho^*}_{up}$, Th, C_1 and C_2
Ensure: Pruned set $\mathcal{P}$ with most *attack resilient* patterns

1: $n^{\rho^*}_{down} \leftarrow d^{\rho^*}_{min} - 1$;
2: $rate_{\rho^*} \leftarrow n^{\rho^*}_{up}/(n^{\rho^*}_{down} + n^{\rho^*}_{up})$; $rate_{min} \leftarrow rate_{\rho^*}$;
3: **for** each pattern $\rho \in \mathcal{P}$ **do**
4: **if** CountOnes$(\rho) < \lceil |\rho| * r_{min} \rceil$ **then** $\mathcal{P} \leftarrow \mathcal{P} \setminus \rho$
5: **else**
6: $n^\rho_{up} \leftarrow$FindOnTime(ρ, C_1, C_2);
7: $d^\rho_{min} \leftarrow$ MinAttLen$(\rho, d^{\rho^*}_{min}, n^\rho_{up}, Th)$-1;
8: **if** $d^\rho_{min} \geq d^{\rho^*}_{min}$ **then** $n^\rho_{down} \leftarrow d^\rho_{min} - 1$;
9: $rate_\rho \leftarrow n^\rho_{up}/(n^\rho_{down} + n^\rho_{up})$;
10: **if** $rate_\rho > rate_{min}$ **then** $\mathcal{P} \leftarrow \mathcal{P} \setminus \rho$
11: **else** $rate_{min} \leftarrow rate_\rho$
12: **else** $\mathcal{P} \leftarrow \mathcal{P} \setminus \rho$
13: **return** $\mathcal{P}$
14: **function** MinAttLen$(\rho, d^{\rho^*}_{min}, n^\rho_{up}, Th)$
15: $d \leftarrow d^{\rho^*}_{min}$;
16: **repeat** $d \leftarrow d + 1$
17: **for** $i = 0$ to $|\rho| - 1$ **do**
18: $\rho' \leftarrow i$-times left cyclic shift of pattern ρ;
19: **if** AttVecSyn$(d, \rho', n^\rho_{up}, Th) \neq NULL$ **then return** d;
20: **until** AttVecSyn$(d, \rho', n^\rho_{up}, Th) = NULL$
21: **function** FindOnTime(ρ, C_1, C_2)
22: $n \leftarrow 1$
23: **for** $i = 0$ to $|\rho| - 1$ **do**
24: $\rho' \leftarrow i$-times left cyclic shift of pattern ρ;
25: **repeat**
26: $x[0] \in C_2$; $u[0] = 0$; $y[0] = 0$;
27: **for** $k = 1$ to n **do**
28: $r[k-1] \leftarrow y[k-1] - C\hat{x}[k-1]$;
29: $\hat{x}[k] \leftarrow A\hat{x}[k-1] + Bu[k-1] + Lr[k-1]$;
30: $x[k] \leftarrow Ax[k-1] + Bu[k-1]$;
31: **if** $\rho'[k] = 1$ **then** $u[k] = K\hat{x}[k]$;
32: **else** $u[k] \leftarrow u[k-1]$; ▷ Skip Execution
33: $\Phi \leftarrow$ **assert**$(|r[1]| \leq Th \wedge \cdots \wedge |r[n]| \leq Th \wedge x[n] \notin C_1)$;
34: $n \leftarrow n + 1$
35: **until** Φ is $unsatisfiable$
36: **return** $n - 1$

FindOnTime() to find the minimum number of closed-loop iterations required by the system to return to the inner safety region C_1 from any state $x[0]$ in the outer safety region C_2 while following the control schedule $(\rho)^\omega$ under no FDI (lines 21-36). To this end, we symbolically simulate attack-free closed-loop iterations of the system starting from an initial state $x[0] \in C_2$ according to the pattern ρ' (where ρ' represents any left cyclic shift of the pattern ρ) (lines 24-26). We assert the negation of the design requirement for the IDS up-time n_{up} i.e. $x[n] \notin C_1$ to verify if the system is not inside the inner safety region C_1 even after n attack-free iterations (line 33). If this assertion is found to be unsatisfiable for all possible ρ' (lines 25-34) then our design requirement is valid and $n-1$ is a safe up-time of the sporadic IDS designed using an attack resilient control skipping pattern ρ (line 35-36). Else we infer that n iterations are not sufficient to bring the system to the inner safety region C_1 starting from any point in the outer safety region C_2 and increase n from 1 until Φ becomes unsatisfiable (line 34).

The function MinAttLen() *analyzes the security level of any protected system against FDI when following the control schedule* $(\rho)^\omega$. It uses the function AttVecSyn (Ref. Algo. 1) to find the minimum attack length for all possible cyclic shifts (ρ') of any control skipping pattern ρ (line 18-19). Starting from $d^{\rho^*}_{min}$ (line 15) it increases the attack length by 1 (line 16) until a successful and stealthy attack vector is found (line 19).

In Algo. 2, we first prune the patterns from $\mathcal{P}$ which do not meet the desired performance requirement (line 4). Next, we calculate the up and down-time for any pattern ρ using FindOnTime() and MinAttLen() respectively (line 6-7). We now prune all patterns whose minimum attack-length is less than periodic execution as they offer lower attack resilience. Next, to minimize the resource requirements of the IDS, we need to find the set of control skipping patterns with minimum IDS execution rate (given by $n_{up}/(n_{up} + n_{down})$) and prune the rest. To this end, we initialize $rate_{min}$ with given sporadic IDS execution rate for periodic execution (lines 1-2). We start by removing the patterns which have a higher IDS execution rate compared to $rate_{min}$ (line 10). For every remaining pattern $\rho \in \mathcal{P}$ we compute their IDS execution rate $rate_\rho$ as the ratio $n^\rho_{up}/(n^\rho_{down} + n^\rho_{up})$ (line 9) and compare it with $rate_{min}$ (line 10) to prune ρ if $rate_\rho > rate_{min}$. Otherwise, $rate_{min}$ is updated with $rate_\rho$ (line 11) to find the patterns with the least IDS execution rate. Finally, Algo. 2 returns a pruned set of control skipping patterns $\mathcal{P}$ (line 13) such that, $\forall \rho \in \mathcal{P}$, **(i)** performance criteria is met and **(ii)** a more sporadic IDS (less IDS activation) can be designed with a formal guarantee of the security against FDI.

4 RESULTS

We demonstrate the efficacy of the proposed approach by designing sporadic IDSs for automotive systems like, **(i)** Vehicle Dynamic Controller (VDC), which regulates side slip (β) and yaw rate (γ) by controlling steering angle [27] and **(ii)** Trajectory Tracking Controller (TTC), that regulates deviation of a vehicle from a given trajectory (D) and a reference velocity (V) by applying proper acceleration [13]. Following [5], the settling time criterion of 5 s allows maximum 50% execution skips, i.e., $r_{min} = 0.5$ for both of these systems. The protection and attacker model is as described in Sec. 2. System matrices (A, B, C), sampling period (h), outer (C_2), inner (C_1) safety regions of the state variables [18, 19] and detector thresholds (Th) for the systems are given in Tab. 1. We now propose a sporadic IDS for both VDC and TTC using pattern-based execution to minimize the IDS execution rate while adhering to the safety requirement mentioned in Tab. 1. In presence of the given sporadic IDSs for the systems following fully periodic control schedule $1^\omega((\rho^*)^\omega)$, the minimum attack lengths required to drive the

Table 1: System Specifications

Sys.	Specifications	C_2	C_1	Th
VDC	A = [0.4450,-0.0458;1.2939,0.4402]; B = [0.0550;4.5607]; C = [0,1]; h = 0.1sec; K = [-0.0987;0.1420]; L = [-0.0390;0.4339]	$\beta \in$ [-1, 1] $\gamma \in$ [-2, 2]	$\beta \in$ [-0.1, 0.1] $\gamma \in$ [-0.2, 0.2]	0.003
TTC	A = [1.0000, 0.1000;0, 1.0000]; B = [0.0050;0.1000]; C = [1 0]; h = 0.1sec; K = [16.0302, 5.6622]; L = [1.8721;9.6532]	$D \in$ [-25, 25] $V \in$ [-30, 30]	$D \in$ [-15, 15] $V \in$ [-18, 18]	2

systems with VDC and TTC to an unsafe state while remaining stealthy are $d_{min} = 3, 11$ respectively. IDS on-time for both systems is 3. The down-time (= $d_{min} - 1$), up-time pair $\langle n_{down}, n_{up} \rangle$ and execution rate of the sporadic IDSs using 1^ω are $\langle 2, 3 \rangle$,0.6 and $\langle 10, 3 \rangle$, 0.2308 respectively for VDC and TTC (Tab. 2 Row 1). We generate

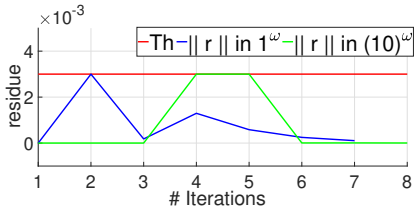
Table 2: Designed Sporadic IDSs for VDC and TTC

VDC				TTC			
Pattern	n_{up}	n_{down}	IDS rate	Pattern	n_{up}	n_{down}	IDS rate
1	3	2	0.6	1	3	10	0.2308
1100	3	4	0.4286	1101001010	3	13	0.1875
11100	3	4	0.4286	1101011100	3	14	0.1765
10	**3**	**5**	**0.375**	**1010011111**	**3**	**15**	**0.1667**
110100	**3**	**5**	**0.375**	**11010111100**	**3**	**15**	**0.1667**
110010	**3**	**5**	**0.375**	-	-	-	-
100011	**3**	**5**	**0.375**	-	-	-	-

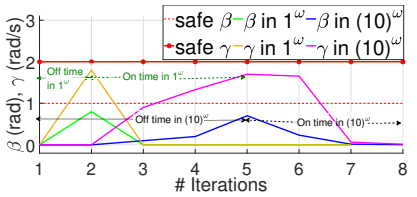
a set of all possible control skipping patterns (up to $l = 12$) and input this set to Algo. 2. It first selects the patterns with minimum $6(= \lceil 12 \times 0.5 \rceil)$ '1's, then returns only those patterns among them (Tab. 2) that exhibit the lowest IDS rate. For VDC, our methodology returns the patterns 10, 110100, 110010 and 100011 as the most resilient ones with 37.5% reduction on IDS execution rate (Row 4-7 in the left half of Tab. 2). The patterns 1010011111 and 11010111100 show similar qualities for TTC (Row 4-5 in the right half of Tab. 2), with a promising reduction of 27.78% in IDS rate. These output patterns and their corresponding IDSs with $\langle n_{down}, n_{up} \rangle$ values ($\langle 5, 3 \rangle$ for VDC and $\langle 15, 3 \rangle$ for TTC) are reported in Tab. 2 in bold fonts, along with other example patterns. For comparison, we consider the effect of a stealthy and successful FDI attack on VDC when it is executing the closed-loop following 1^ω (periodic) and $(10)^\omega$ (best pattern returned by Algo. 2). Fig. 3a shows under stealthy FDI, the residue of the VDC ($||r||$) stays always below Th for both 1^ω, $(10)^\omega$ and the states (β, γ) take longer to become unsafe for $(10)^\omega$ leading to a higher IDS down-time (from 2 to 5). This validates our principal claim of potential increment in system attack resilience provably guaranteeing the security by judiciously skipping some control executions. Next, we demonstrate the usefulness of our sporadic IDS in intra-vehicular networks.

Effect of Intentional Skips on CAN Bandwidth: Controller Area Network (CAN) [3] is a lightweight broadcast protocol used to connect automotive domain Electronic Control Units (ECUs). CAN sends messages without source or destination information and lacks any security mechanisms. Attackers have exploited the lack of security primitives in CAN to inject false data, manipulate denial of service, or launch zero-day like attacks on automotive[8, 11]. Hence the security of intra-vehicular network is an important issue due to the safety-critical nature of automotive systems[2, 14, 26]. As a result, the use of MAC has been made mandatory as per AUTOSAR standards[25]. So we explore the efficacy of our proposed sporadic IDS design approach towards reducing the computational and communication overhead in intra-vehicular communication network protocols like CAN.

Let us consider such an automotive system where the CAN messages are communicated through the bus with a speed of B bps at periodicity $p_1, p_2, \ldots, p_k$ such that $p_1 > p_2 > \cdots > p_k$. The number of message types with rate p_i is given by m_i, $i \in [1, k]$. Assume that IDS is implemented for messages with periodicity $p_{k'}$ and there are $m_{k'} > 0$ number of such types of messages. Similar to [3], we consider a p_1-length observation window ($\geq$ the largest period) and compute bandwidth consumption in CAN bus for the aforementioned setup through the following steps.



(a) Stealthy attack on VDC



(b) Higher n_{down} for control skips

Figure 3: VDC under FDI

(i) We find out the number of messages communicated over the observation window p_1. For any m_i it is $c_i = \lceil p_1/p_i \rceil \forall i \in [1, k]$. We consider maximum CAN payload for each message, i.e. 64 bits.
(ii) For each of the m'_k different type of messages, the IDS rate is $rate_i, i \in [1, m'_k]$. If we design the IDS with CMAC/AES-128 (with a-bit CMAC) [25] encryption to provide confidentiality and authenticity, payload will be of size (64+a) bits. This will convert to $\lceil$(64+a)/128$\rceil$ AES blocks or b= ($\lceil$(64+a)/128$\rceil \times 128$)/64 CAN frames (CAN payload size=64). In such an arrangement, each CAN frame will be replaced by b CAN frames when IDS is active (refer Fig. 4a where $b = 4$). Hence, over the observation window, each of the $m_{k'}$ messages is transmitted $(1 - rate_i) \times c_{k'}$ times without IDS active and $b \times rate_i \times c_{k'}$ times with IDS active giving a total count of $(1 + (b-1)rate_i) \times c_{k'}$.

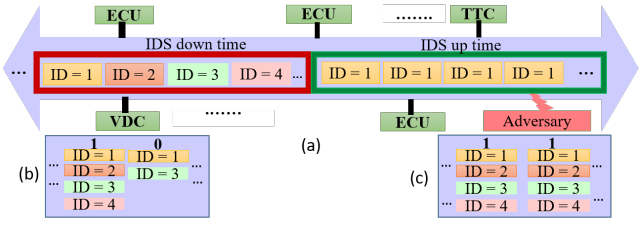


Figure 4: a) CAN Transmissions with sporadic IDS under FDI, b) Message flow for $(1)^\omega$, c) Message flow for $(10)^\omega$

(iii) Additional 47 bits are added to the payload to form one CAN frame (SOF + Arbitration + RTR + Control + CRC + Acknowledgment + EOF + Interframe Space = 1 + 11 + 1 + 6 + 16 + 2 + 7 + 3 = 47 bits)[3]. Thus, in our consideration, size of each CAN frame is (64+47) bits = 111 bits. Following this, total bandwidth consumption over *observation window* is computed as $T = 111\times[m_1+m_2\times c_2+..+\sum_{i=1}^{m_{k'}}(1+(b-1)rate_i)\times c_{k'}+..+m_k\times c_k]/B$. Let the IDS rates for some control skipping pattern, output by Algo. 2 be $rate'_i, \forall i \in [1, m_{k'}]$. Since Algo. 2 ensures if proposed patterns are used $rate'_i < rate_i (\forall i \in [1, m_{k'}])$, the improvement in bandwidth consumption when executing a pattern-based schedule compared to a periodic schedule is given as, $(T - T')/T = 111 \cdot \sum_{i=1}^{m_{k'}}((1+(b-1)(rate_i - rate'_i)) \cdot c_3)/T$ considering T' as the bandwidth consumed by pattern-based schedule.

Example: Let us consider the following setup of (#message, periodicity): $\langle m_1, p_1\rangle = \langle 10, 1\rangle$, $\langle m_2, p_2\rangle = \langle 20, 0.2\rangle$, $\langle m_3, p_3\rangle = \langle 2, 0.1\rangle$ (VDC), $\langle m_4, p_4\rangle = \langle 2, 0.1\rangle (TTC)$ in CAN bus. So, the VDC and TTC both require two types of messages (sensor o/p, control i/p) of period p_3 and p_4 respectively. These are denoted by CAN IDs $1 \cdots 4$(Fig. 4a). During skips in the control execution, actuation signals are not communicated as we can see in Fig. 4c, which also frees the bandwidth. If the IDS in place uses 128 bit CMAC (i.e. $a = 128$), each CAN frame is replaced with $b = 4$ CAN frames when IDS is active (refer Fig. 4a). Following the derived formula for the aforementioned setup, we get *16.25% net improvement in CAN bandwidth consumption* using the secure control schedule 10^ω for VDC and 1010011111^ω for TTC. Considering our methodology to design such pattern-based secure control schedules for a significant number of control loops will have an additive effect on the bandwidth saving. Thus our methodology helps to design pattern-based sporadic IDSs that promise better resource utilization.

5 RELATED WORK

In [16], the authors discuss suitable conditions under which a control system with χ^2-based detectors is stealthily attackable. The performance degradation of such χ^2 detector enabled systems in the presence of stealthy attacks has been quantified in [4]. In [17], the authors report such 'fake disturbance attacks' and their implications in network control systems in the presence of deterministic monitoring algorithms. The idea of stealthy attacks on both sensor and actuator sides being able to destabilize automated power generation systems with threshold-based detectors has been discussed in [24]. Authors in [1, 26] also discuss security vulnerabilities in the automotive CPS domain. Designing resilient control implementations by leveraging secure state estimation techniques, more specifically in the automotive context has been reported in [21]. The idea of sporadically using IDSs like MAC computation has been investigated in a different line of works [9, 12, 13]. In [6], the authors explore the advantage of employing lightweight periodic authentication schemes like Physically Unclonable Functions (PUFs) in the context of a sporadically available IDS for CPS security. In the current work, we assume that the IDS security primitive is available for n_{up} consecutive iterations followed by an off time for which we can establish a guarantee that the performance degradation due to stealthy attacks is inside recoverable limits.

6 CONCLUSION

The present work formally analyzes the attack resilience of *intentionally* skipped control executions without compromising stability. The safe and resilient patterns generated by the method helps to reduce the computation and communication overhead of existing IDSs when employed in automotive CPS. Although it is an offline approach, integrating this SMT-based technique with safe but approximate analysis (e.g. using 'Barrier functions') can help increase the scalability of the approach for applicability in complex industrial test cases. This along with synthesizing optimal control strategies with joint objectives of optimizing performance, minimizing resource usage, and guaranteeing uncompromised security are important future extensions possible for this work.

ACKNOWLEDGMENTS

We thank Dr. Majid Zamani for helping us to formalize the problem.

REFERENCES

[1] Paul Carsten and et al. 2015. In-vehicle networks: Attacks, vulnerabilities, and proposed solutions. In *CISRC*. ACM.
[2] Kyong-Tak Cho et al. 2016. Fingerprinting electronic control units for vehicle intrusion detection. In *USENIX Security)*. 911–927.
[3] JA Cook et al. 2007. Controller Area Network (CAN). *EECS* 461 (2007), 1–5.
[4] Benjamin Gerard et al. 2018. Cyber Security and Vulnerability Analysis of Networked Control System subject to False-Data injection. In *ACC*. IEEE.
[5] Sumana Ghosh et al. 2017. A structured methodology for pattern based adaptive scheduling in embedded control. *ACM TECS* 16, 5s (2017), 189.
[6] Saurav K. Ghosh et al. 2018. Performance, Security Trade-offs in Secure Control. *IEEE ESL* (2018).
[7] Jairo Giraldo et al. 2018. A survey of physics-based attack detection in cyber-physical systems. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 76.
[8] Andy Greenberg. 2015. Hackers remotely kill a jeep on the highway—with me in it. *Wired* 7 (2015), 21.
[9] Ilija Jovanov et al. 2018. Secure State Estimation with Cumulative Message Authentication. In *CDC*. IEEE.
[10] Ipsita Koley et al. 2020. Formal synthesis of monitoring and detection systems for secure CPS implementations. In *DATE*. IEEE, 314–317.
[11] Karl Koscher et al. 2010. Experimental security analysis of a modern automobile. In *S & P*. IEEE, 447–462.
[12] Vuk Lesi et al. 2017. Security-Aware Scheduling of Embedded Control Tasks. *ACM TECS* 16, 5 (2017).
[13] Vuk Lesi et al. 2020. Integrating Security in Resource-Constrained Cyber-Physical Systems. *ACM TCPS* 4, 3 (2020), 1–27.
[14] Stefano Longari et al. 2019. CopyCAN: An Error-Handling Protocol based Intrusion Detection System for Controller Area Network. In *CPS-SPC*. 39–50.
[15] Rupak Majumdar et al. 2011. Performance-aware scheduler synthesis for control systems. In *Proc. Embedded Software*. 299–308.
[16] Yilin Mo et al. 2010. False data injection attacks in control systems. In *SCS*.
[17] Yilin Mo et al. 2016. On the Performance Degradation of Cyber-Physical Systems Under Stealthy Integrity Attacks. *IEEE TAC* 61, 9 (2016), 2618–2624.
[18] Bosch Motorsport. 2018. Acceleration Sensor MM5. 10.
[19] Bosch Motorsport. 2020. Steering Wheel Angle Sensor LWS. http://www.bosch-motorsport.de/content/downloads/Raceparts/en-GB/54425995191962507.html
[20] Arslan Munir et al. 2018. Design and analysis of secure and dependable automotive CPS: A steer-by-wire case study. *IEEE TDSC* (2018).
[21] Miroslav Pajic et al. 2017. Design and Implementation of Attack-Resilient Cyberphysical Systems: With a Focus on Attack-Resilient State Estimators. *IEEE Control Systems Magazine* 37, 2 (April 2017), 66–81.
[22] Damoon Soudbakhsh et al. 2013. Co-design of control and platform with dropped signals. In *ICCPS*. ACM.
[23] Andre Teixeira et al. 2015. A secure control framework for resource-limited adversaries. *Automatica* 51 (2015), 135–148.
[24] Andre Teixeira et al. 2015. Secure control systems: A quantitative risk management approach. *IEEE Control Systems Magazine* 35, 1 (2015), 24–45.
[25] Springer Fachmedien Wiesbaden. 2013. AUTOSAR — The Worldwide Automotive Standard for E/E Systems. *ATZextra worldwide* 18, 9 (Oct 2013), 5–12.
[26] Clinton Young et al. 2019. Survey of automotive controller area network intrusion detection systems. *IEEE Design & Test* 36, 6 (2019), 48–55.
[27] Shuibo Zheng et al. 2006. Controller design for vehicle stability enhancement. *Control Engineering Practice* 14, 12 (2006), 1413–1421.