Cross-Layer Codesign for Secure Cyber-Physical Systems

Bowen Zheng, *Student Member, IEEE*, Peng Deng, *Student Member, IEEE*, Rajasekhar Anguluri, *Student Member, IEEE*, Qi Zhu, *Member, IEEE*, and Fabio Pasqualetti, *Member, IEEE*

***Abstract*—Security attacks may have disruptive consequences on cyber-physical systems, and lead to significant social and economic losses. Building secure cyber-physical systems is particularly challenging due to the variety of attack surfaces from the cyber and physical components, and often to limited computation and communication resources. In this paper, we propose a cross-layer design framework for resource-constrained cyber-physical systems. The framework combines control-theoretic methods at the functional layer and cybersecurity techniques at the embedded platform layer, and addresses security together with other design metrics such as control performance under resource and real-time constraints. We use the concept of interface variables to capture the interactions between control and platform layers, and quantitatively model the relation among system security, performance, and schedulability via interface variables. The general codesign framework is customized and refined to the automotive domain, and its effectiveness is demonstrated through an industrial case study and a set of synthetic examples.**

***Index Terms*—Codesign, control performance, cross-layer, cyber-physical systems, schedulability, security.**

I. Introduction

CYBER-PHYSICAL systems, where embedded computing and control techniques are integrated together for the interaction with physical processes, have wide applicability in various domains such as advanced automotive systems, reliable medical devices, and energy management systems [1], [2]. Recently, security for cyber-physical systems has become a pressing issue, as evidenced, for example, by the Stuxnet worm attack [3], the experiments on automotive security [4], [5], and the Maroochy water breach [6].

Ensuring security for cyber-physical systems is increasingly challenging, particularly because attacks may come from a variety of cyber and physical interfaces. While traditional cybersecurity approaches (such as encryption and authentication) may protect a system from attacks against cyber components and data, they are ineffective against insider and physical attacks [6]. On the other hand, control-theoretic approaches may be applied to detect attacks based on the analysis of the system state and dynamics [7]. We argue that to provide comprehensive and effective cyber-physical protection, it is important to address security issues across multiple layers of control design and embedded systems.

Another important aspect is that, for many cyber-physical systems, the adoption of security techniques introduces overhead on computation and communication. In turn, this may affect the system performance, safety, reliability, and other timing related metrics. To build correct, efficient, and secure cyber-physical systems, it is crucial to quantitatively model the impacts of security techniques on other related metrics, and address them together in a codesign environment.

In this paper, we propose a cross-layer codesign framework to combine control-theoretic methods at the functional layer and cybersecurity techniques at the embedded platform layer. Furthermore, we address security together with other design metrics, in particular the control performance, under resource and real-time constraints.

In the literature for cyber-physical system security, researchers have proposed various control-oriented approaches for attacks on cyber-physical systems [7]–[10]. Pasqualetti *et al.* [7] designed attack detection and identification monitors from a control-theoretic perspective. In [8], an optimal control approach is proposed to address jamming in the communication channel between the controller and the plant. In [9], a recursive networked predictive control method is proposed to deal with denial of service attacks. In [10], a minimax control approach is presented to address network packet scheduling attacks. Resource and real-time constraints are not considered in these approaches, and there is no guarantee of schedulability and control performance.

Lin *et al.* [11], [12] modeled the impact of message authentication techniques on real-time constraints in automotive systems. However, they do not consider the impact of these messages on sampling periods and control performance, and do not model their relation with system security (rather it is assumed that authentication requirements are directly given at the message level, which may not be practical in many design processes).

Manuscript received June 16, 2015; revised November 1, 2015; accepted January 11, 2016. Date of publication February 7, 2016; date of current version April 19, 2016. This work was supported by the Office of Naval Research under Grant N00014-14-1-0815 and Grant N00014-14-1-0816. This paper was recommended by Associate Editor S. Hu, S. Hu, and A. Y. Zomaya. *(Corresponding author: Qi Zhu.)*

B. Zheng, P. Deng, and Q. Zhu are with the Department of Electrical and Computer Engineering, University of California at Riverside, Riverside, CA 92507 USA (e-mail: bzheng@ece.ucr.edu; pdeng002@ucr.edu; qzhu@ece.ucr.edu).

R. Anguluri and F. Pasqualetti are with the Department of Mechanical Engineering, University of California at Riverside, Riverside, CA 92507 USA (e-mail: rangu003@ucr.edu; fabiopas@engr.ucr.edu).

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

Digital Object Identifier 10.1109/TCAD.2016.2523937

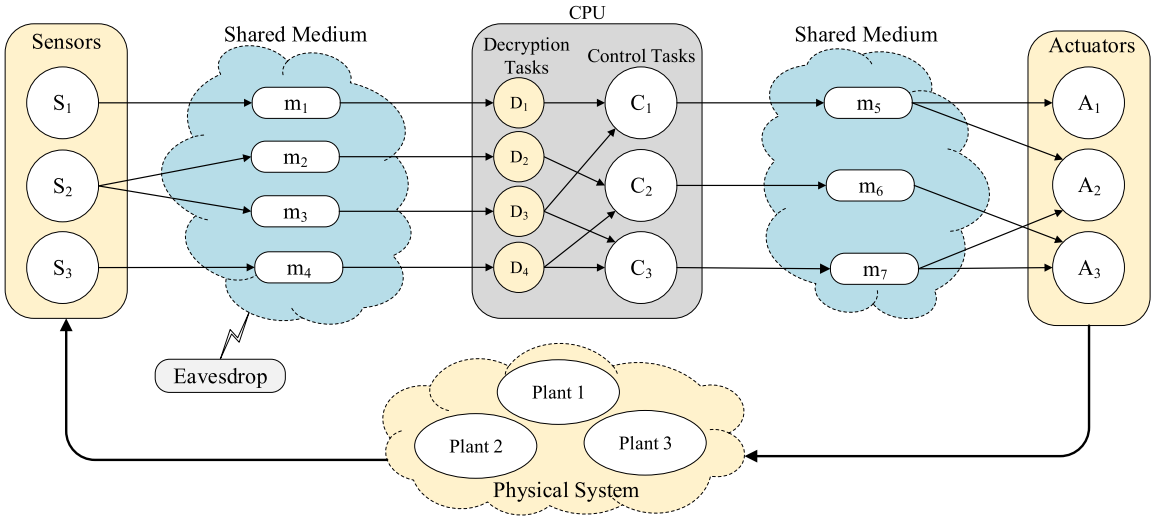


Fig. 1. Cyber-physical system with multiple control loops sharing an embedded platform. Attackers may eavesdrop on the communication medium and apply various attacks.

More recently Pasqualetti and Zhu [13] proposed an integrated framework to address control performance, security, and platform schedulability together. While outlining some of the initial ideas of cross-layer codesign for security, the paper only considers a simple single-task platform and consequently straightforward schedulability model and simplified notion of security level. In this paper, we consider a more complex and realistic model of cyber-physical systems, where multiple control loops share the computation and communication platform. Furthermore, we introduce a notion of security level based on the estimation error of an optimal Kalman filter [14], a general codesign formulation, and its application to automotive systems with control performance and schedulability models.

Our framework quantitatively models the impact of security techniques on control performance and platform schedulability, and explores tradeoffs between security level and control performance while guaranteeing real-time constraints for cyber-physical systems. The main contributions of this paper include the following.

1) Models that capture the relation between system security level and applied security techniques, and models that quantify the impact of security techniques on control performance and platform schedulability under the proposed attack model.
2) Codesign formulation and algorithm that explores the configuration of security techniques to address both system security and control performance while guaranteeing system schedulability.
3) A customized and refined codesign formulation for automotive systems and case studies derived from an industrial experimental vehicle and synthetic examples.

The rest of this paper is organized as follows. In Section II, we introduce our general cross-layer codesign framework, including a motivation example, system model, attack model, and codesign formulations. In Section III, a customized and refined codesign formulation for automotive systems is introduced. In Section IV, case studies in automotive domain are demonstrated to show the effectiveness of our approach. Section V concludes this paper.

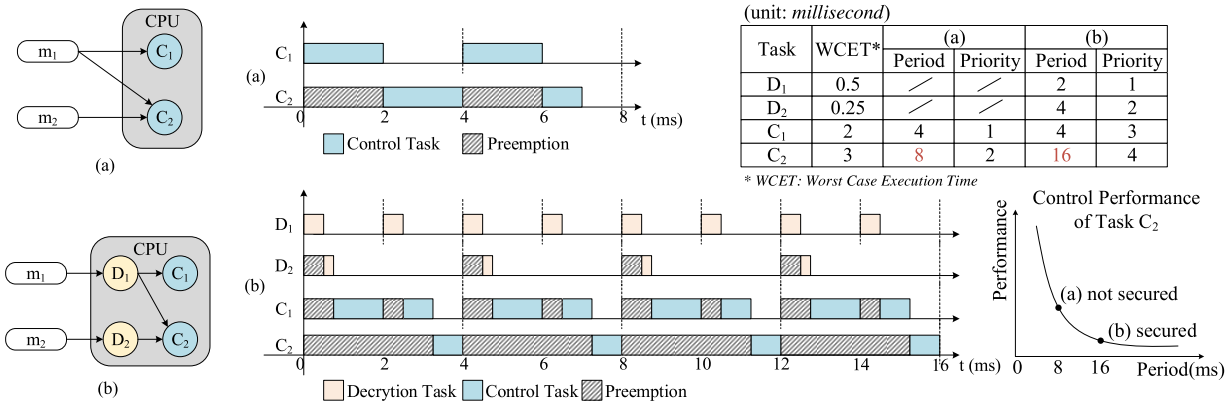
II. General Codesign Framework

This paper addresses a typical cyber-physical system, where multiple control loops share an embedded platform, with messages transmitted from sensors (vision sensors, global positioning system, ultrasound, etc.) to controllers and from controllers to actuators, as shown in Fig. 1 and similarly considered in [10]. Each controller (implemented as a control task) collects the sensed information, processes it on a shared computation unit (e.g., a single-core CPU),[1] and sends commands to various actuators. In our model, a message from a sensor may be sent to multiple control tasks for sharing information (which is common in many domains such as automotive systems). If a message is encrypted for security measurement, a dedicated decryption task is used for decrypting the message and send it to the receiving tasks (this approach reduces overhead, compared with carrying out the decryption of the same message within each receiving task).[2]

The attackers may be able to eavesdrop on the communication medium and further reconstruct the system state. This results not only in a loss of privacy, but can further be used as the basis for other malicious attacks. The system is resource-constrained, as control tasks compete for computation resources and messages compete for communication resources. Applying security techniques such as message encryption will introduce computation and communication overhead, through the elongation of message transmission time, the additions of decryption tasks, and consequently the elongation of control task execution time due to resource contention. This will in turn have a significant impact on system schedulability and control performance, as demonstrated in the following motivating example.

[1] We assume all the control tasks are implemented on a single computation unit in this paper. Our formulation can be extended to address multicore and multiprocessor platforms with more complex models for schedulability and security level measurement, as planned in the future work.

[2] In this paper, we assume the message is encrypted with the same key for all the receiving tasks. A more complex strategy with different keys may be used for higher level of security with significantly more overhead.



Task	WCET*	(a)		(b)	
		Period	Priority	Period	Priority
D_1	0.5	/	/	2	1
D_2	0.25	/	/	4	2
C_1	2	4	1	4	3
C_2	3	8	2	16	4

Fig. 2. Motivating example. (a) No message is encrypted. (b) Both messages are encrypted and the period of C_2 has to be increased, which leads to higher security level but lower control performance.

A. Motivating Example

A motivating example is shown in Fig. 2. In this example, there are two control tasks C_1 and C_2, two messages m_1 and m_2, and potentially two decryption tasks D_1 and D_2 if the corresponding messages are encrypted. We show only the control and decryption tasks here for simplicity, and will model sensing and actuation tasks later in problem formulation. All the tasks are implemented on a single-core CPU under the pre-emptive fixed-priority scheduling policy (commonly used in cyber-physical systems, such as automotive systems with OSEK standard [15]). In order to guarantee the correct execution order, the priorities of decryption tasks are set higher than the control tasks. We further assume C_1 has higher priority than C_2. The initial periods of C_1 and C_2 are set to 4 ms and 8 ms, respectively.

We consider two scenarios: 1) no message is encrypted and the system is not protected and 2) both messages are encrypted and the system is protected. In scenario 1), no decryption task is needed and there is no security overhead. Tasks C_1 and C_2 can be completed within their periods, i.e., before their next periodic activation. In scenario 2), the overhead of two decryption tasks elongates the time it takes to complete task C_2, and consequently the period of C_2 cannot be smaller than 16 ms (otherwise C_2 will not complete within its period and the system functionality may be incorrect). As discussed in [16] and [17], control performance typically decreases significantly when the control task period increases. Therefore, this motivating example has clearly shown the additions of security measurements may have a negative impact on system timing, and consequently control performance and system schedulability. It is essential to quantitatively analyze the tradeoff among these metrics. In the following, we will introduce our general codesign formulation for this purpose.

B. General Formulation

Our codesign framework addresses three design metrics: 1) control performance; 2) system security level; and 3) platform schedulability. Control performance and system security level are measured at the functional layer, while schedulability is analyzed at the embedded platform layer. As shown in Fig. 3, to bridge these metrics, a set of interface variables are introduced, specifically the sampling period of every control task and the selection of messages to be encrypted. Intuitively, when the sampling period of a control task increases, its control performance decreases, and platform schedulability becomes easier with less frequent activation of the control task. On the other hand, when the number of messages being encrypted increases, the system security level increases, and platform schedulability becomes harder because of the increased overhead—the sampling periods may have to increase for schedulability concern thereby worsening the control performance. These relations are quantitatively modeled in our codesign formulation as introduced below.

First, we define the following notation for the cyber-physical system in Fig. 1. Tasks are represented by $\mathcal{T} = \mathcal{T}_\mathcal{S} \cup \mathcal{T}_\mathcal{C} \cup \mathcal{T}_\mathcal{D} \cup \mathcal{T}_\mathcal{A}$, where $\mathcal{T}_\mathcal{S} = \{\tau_s^1, \tau_s^2, \ldots, \tau_s^l\}$ denotes the set of sensing tasks, $\mathcal{T}_\mathcal{C} = \{\tau_c^1, \tau_c^2, \ldots, \tau_c^m\}$ denotes the control tasks, $\mathcal{T}_\mathcal{D} = \{\tau_d^1, \tau_d^2, \ldots, \tau_d^p\}$ denotes the decryption tasks, and $\mathcal{T}_\mathcal{A} = \{\tau_a^1, \tau_a^2, \ldots, \tau_a^q\}$ denotes the actuation tasks. Each task τ_i is associated with an activation period T_τ^i and worst case execution time C_τ^i. The worst-case execution time for sensing tasks include the time for processing the sensor data from crude input form. The relative deadline of each task is set equal to its period as in typical real-time systems. Messages are represented by $\mathcal{M} = \{m_1, m_2, \ldots, m_p\}$. $\text{src}(m_i)$ denotes the source task of message m_i, and the set $\{\text{dst}(m_i)\}$ includes all destination tasks of message m_i. Each message m_i is associated with a period T_{m_i} and an original transmission time C_{m_i} (without encryption). The relative deadline of each message is set equal to its period.

A control loop consists of a control task connected with a set of sensors, a set of actuators, and the corresponding plants. A control path p is a sequence of tasks (including sensing, control, and actuation tasks) connected through messages.

In our codesign formulation, we explore the selection of messages for encryption and the assignment of periods to control tasks to address both control performance and system security while guaranteeing platform schedulability. In the following formulation from (1) to (5), we set control performance as the optimization objective and assume constraints on the system security level. Then, by varying the requirements

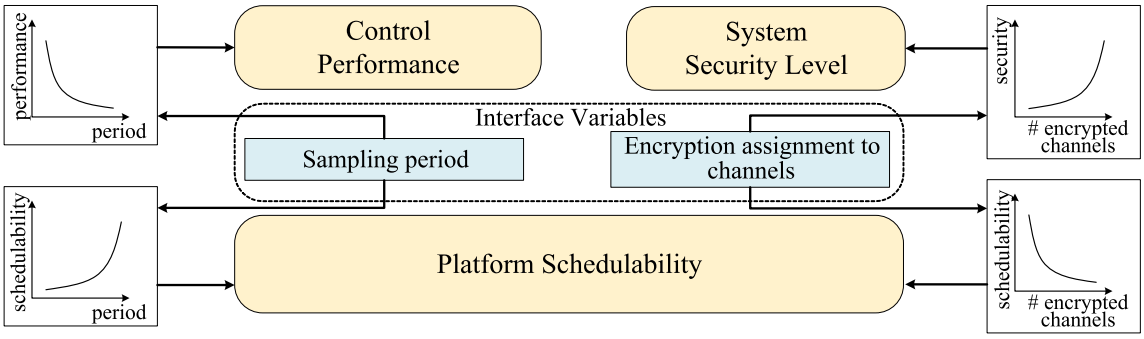


Fig. 3. Illustration of the interface variables and their relations to design metrics.

on the system security level, we can obtain the Pareto front between control performance and security level

$$\text{maximize: } J\left(\overrightarrow{T_{\tau_c}}\right) \text{ (control performance)} \quad \text{s.t.} \tag{1}$$

$$S\left(\overrightarrow{o_m}\right) \geq S_0 \text{ (security constraint)} \tag{2}$$

$$U_e\left(\overrightarrow{o_m}, \overrightarrow{T_{\tau_c}}\right) \leq U_{e0} \text{ (computational resource)} \tag{3}$$

$$U_c\left(\overrightarrow{o_m}\right) \leq U_{c0} \text{ (communication resource)} \tag{4}$$

$$l_p\left(\overrightarrow{o_m}, \overrightarrow{T_{\tau_c}}\right) \leq D_p \text{ (end to end latency).} \tag{5}$$

As stated before, the design variables are the interface variables, which include the control task periods $\overrightarrow{T_{\tau_c}} = \{T^1_{\tau_c}, T^2_{\tau_c}, \ldots, T^m_{\tau_c}\}$, and the selection of messages for encryption, denoted by $\overrightarrow{o_m} = \{o_{m_1}, o_{m_2}, \ldots, o_{m_k}\}$. The binary variable o_{m_i} is 1 if message m_i is encrypted, and 0 otherwise. The variables in (1)–(5) are defined as follows and the details of these equations will be introduced in the rest of the section. J represents the control performance, which is a function of control task period $\overrightarrow{T_{\tau_c}}$ and works as the objective of this problem. S denotes the security level of the system, which is a function of the selection of messages for encryption $\overrightarrow{o_m}$. S_0 is the minimum security level set in the design requirements. U_e represents the utilization of the computation unit, and U_{e0} is the required maximum utilization. U_c denotes the utilization of the shared communication medium, and similarly, U_{c0} is the required upper bound on communication utilization. l_p represents the end to end latency of path p, and D_p is the required deadline for path p. In what follows, we introduce how each of the equations above are refined for modeling control performance, security level, and schedulability, with respect to the design variables.

1) Control Performance Modeling: We consider linear continuous-time dynamics for each physical plant

$$\begin{aligned} \dot{x} &= Ax + Bu + w \\ y &= Cx + v \end{aligned} \tag{6}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $x : \mathbb{R} \to \mathbb{R}^n$ is the map describing the system state, $u : \mathbb{R} \to \mathbb{R}^m$ is the control input, and $y : \mathbb{R} \to \mathbb{R}^p$ is the measured output. Finally, $w : \mathbb{R} \to \mathbb{R}^n$ and $v : \mathbb{R} \to \mathbb{R}^p$ represent process and measurement noise, which we assume to be zero-mean, Gaussian, and white.

Each continuous-time physical plant is controlled by a digital controller τ^i_c with a sampling period $T^i_{\tau_c}$, which is also the activation period of the corresponding control task. Thus, intuitively, the longer the sampling period, the worse the performance of the control system. For controller τ^i_c, we assume the control performance $J^i_{\tau_c}$ decreases exponentially as the sampling period $T^i_{\tau_c}$ increases, as in [16]–[18].[3] As in these works, we let the performance $J^i_{\tau_c}$ be an exponentially decaying function of $T^i_{\tau_c}$ defined as

$$J^i_{\tau_c} = \alpha^{-\beta T^i_{\tau_c}} \tag{7}$$

for appropriate constants $\alpha \in \mathbb{R}$, $\beta \in \mathbb{R}$, and $\alpha > 0$, $\beta > 0$. For each control task, α and β can be obtained by fitting the relation of control performance and sampling period with an exponential decay function, as described in Section III-A.

For a system consisting of multiple control tasks, the overall control performance is calculated as the weighted average in (8), where $\omega^i_{\tau_c}$ is the weight for each control task and $|\mathcal{T}_\mathcal{C}|$ is the number of control tasks

$$J = \frac{1}{|\mathcal{T}_\mathcal{C}|} \sum_{\tau^i_c \in \mathcal{T}_\mathcal{C}} \omega^i_{\tau_c} J^i_{\tau_c}. \tag{8}$$

2) Security Level Modeling: We consider attackers with knowledge of the system dynamics [i.e., the matrices A, B, and C in (6)], and attempt to reconstruct the system state by eavesdropping messages containing sensor measurements. It should be noticed that reconstructing the system state results in not only a loss of privacy, but also in vulnerabilities to feedback attacks.

To protect against such attacks, a key-based encryption technique is adopted in our framework. As stated before, for simplicity and efficiency, we only consider the case where a message is encrypted with the same key for all receiving tasks. Our formulation can be extended to address multiple-key distribution scenarios, using techniques similar to the ones in [11].

The security level of the control task τ^i_c is defined as the complexity to eavesdrop the messages with sensor measurements and observe the system state. It is modeled in (9). Specifically, $P(n^i_{\tau_c}, \mu)$ represents the probability for an attacker

[3] In general, the relation between control performance and sampling period could be quite complex and may require simulations for accurate capturing. In those cases, we may approximate the control performance with a closed-form representation (if possible) and apply our codesign formulation, or we can combine our codesign formulation and simulated annealing algorithm directly with simulations for exploring control performance.

to eavesdrop μ encrypted sensing messages for task τ_c^i, where $n_{\tau_c}^i$ is the number of encrypted sensing messages. $\xi(\Omega_\mu(\rho))$ represents the complexity for the attacker to estimate the system state from the ρth element of Ω_μ, which is the set containing all possible combinations of the eavesdropped messages, i.e., μ encrypted messages together with all unencrypted messages [its cardinality is $\binom{n_{\tau_c}^i}{\mu}$]. More about $\xi(\Omega_\mu(\rho))$ is discussed later. $P(n_{\tau_c}^i, \mu)$ can be further defined as in (10), where $\mathcal{D}(l_{\text{key}})$ denotes the probability for an attacker to decrypt one message with encryption key length l_{key}

$$S_{\tau_c}^i = \sum_{\mu=0}^{n_{\tau_c}^i} \sum_{\rho=1}^{\binom{n_{\tau_c}^i}{\mu}} P\big(n_{\tau_c}^i, \mu\big)\xi\big(\Omega_\mu(\rho)\big) \tag{9}$$

$$P\big(n_{\tau_c}^i, \mu\big) = \mathcal{D}(l_{\text{key}})^\mu \big(1 - \mathcal{D}(l_{\text{key}})\big)^{n_{\tau_c}^i - \mu}. \tag{10}$$

We use the example in Fig. 1 to illustrate the parameters shown above. Let us consider the solution in which only messages m_1 and m_3 are encrypted. In this case, the number of sensing messages of task C_1 is 2, i.e., m_1 and m_3, and the number of encrypted sensing messages of task C_1 is 2 as both m_1 and m_3 are encrypted. The number of sensing message of task C_2 is 2, i.e., m_2 and m_4, and the number of encrypted sensing messages of task C_2 is 0, as neither m_2 nor m_4 is encrypted. The number of sensing messages of task C_3 is 2, and the number of encrypted sensing messages of task C_3 is 1. Using C_3 as an example for (10), the number of encrypted sensing messages $n_{\tau_c}^3 = 1$. The attacker may eavesdrop 0 or 1 encrypted sensing message (i.e., μ is 0 or 1), with the probability defined as $P(1, 0)$ and $P(1, 1)$ in (10). In addition, the attacker can always eavesdrop the unencrypted message m_4 to learn about C_3. Set Ω_μ contains m_4 and m_3 if $\mu = 1$ (i.e., encrypted message m_3 is eavesdropped), and only contains m_4 if $\mu = 0$. Equation (9) considers all possible situations for C_3 and computes its overall security level.

We simultaneously consider multiple control loops. A successful reconstruction of the system state of any control loop may lead to the whole system being attacked, therefore the system level security is defined as the minimum security level among all control tasks as

$$S = \min_{\tau_c^i \in \mathcal{T}_\mathcal{C}} S_{\tau_c}^i. \tag{11}$$

The function $\xi(\Omega_\mu(\rho))$ can be defined in different ways. For deterministic systems (that is, without process and measurement noise), as shown in [13], $\xi(\Omega_\mu(\rho))$ can be defined based on the observability Gramian [14]. This measure of observability quantifies the relative importance of different measurement channels based on the system dynamics only, and independently of any particular estimation scheme. For stochastic systems driven by process and measurements noise, $\xi(\Omega_\mu(\rho))$ can be defined based on the estimation error of an optimal Kalman filter. This measure of observability, which is inherently dependent on the Kalman estimation procedure, allows us to highlight the role of system noise with respect to the system security level. We now present these two metrics.

a) Option 1—Observability Gramian: Let $\mathcal{K} \subseteq \{1, \ldots, p\}$ be the set of measurements decrypted by the attacker, and let $y_\mathcal{K}$ be the decrypted measurements. The observability Gramian is defined as

$$\mathcal{O}_\mathcal{K} \coloneqq \sum_{\tau=0}^{\infty} \big(A^\intercal\big)^\tau C_\mathcal{K}^\intercal C_\mathcal{K} A^\tau \tag{12}$$

where $C_\mathcal{K}$ is the output matrix associated with the decrypted measurements ($y_\mathcal{K} = C_\mathcal{K} x$). The energy associated with the decrypted measurements $\mathcal{K}$ and due to the free evolution of the system from the x is

$$\begin{aligned} E(x) &\coloneqq \sum_{\tau=0}^{\infty} \|y_\mathcal{K}(\tau)\|^2 \sum_{\tau=0}^{\infty} y_\mathcal{K}^T y_\mathcal{K} = \sum_{\tau=0}^{\infty} x^T(\tau) C^T C x(\tau) \\ &= \sum_{\tau=0}^{\infty} x^T \big(A^T\big)^\tau C^T C A^\tau x = x^T \mathcal{O}_\mathcal{K} x \geq \lambda_{\min}(\mathcal{O}_\mathcal{K}) \end{aligned} \tag{13}$$

where $\lambda_{\min}(\mathcal{O}_\mathcal{K})$ denotes the smallest modulus of the eigenvalues of $\mathcal{O}_\mathcal{K}$. The following facts can be formally proven with standard methods [14]. First, the larger $\lambda_{\min}(\mathcal{O}_\mathcal{K})$, the easier the reconstruction of the system state from measurements. Thus, the eigenvalue $\lambda_{\min}(\mathcal{O}_\mathcal{K})$ measures the information of the system state contained in the measurements $y_\mathcal{K}$. Second, the eigenvalue $\lambda_{\min}(\mathcal{O}_\mathcal{K})$ is a function of both the cardinality and the decrypted messages. Third, the inequality (13) holds with equality for certain system states and for an infinite observation horizon. Otherwise, $\lambda_{\min}(\mathcal{O}_\mathcal{K})$ is a lower bound on the information retrieved by the attacker from the decrypted measurements $y_\mathcal{K}$. Thus, for deterministic systems we select

$$\xi(\Omega_\mu(\rho)) = \lambda_{\min}^{-1}(\mathcal{O}_{\Omega_\mu(\rho)}). \tag{14}$$

b) Option 2—Kalman filter: Let $y_\mathcal{K}$ be the measurements decrypted by the attacker, and define the Kalman filter as

$$\hat{x}_{k+1} = A\hat{x}_k + K_k\big(y_k - C\hat{x}_k\big) + Bu_k$$

where the Kalman gain K_k and the error covariance matrix $P_{k+1} \triangleq \mathbb{E}[(\hat{x}_{k+1} - x_{k+1})(\hat{x}_{k+1} - x_{k+1})^T]$ can be calculated with the recursions

$$\begin{aligned} K_k &= AP_kC^T\big(CP_KC^T + \Sigma_v\big)^{-1} \\ P_{k+1} &= AP_kA^T - AP_kC^T\big(CP_kC^T + \Sigma_v\big)^{-1}CP_kA^T + \Sigma_w \end{aligned}$$

with initial conditions $\hat{x}_1 = \mathbb{E}[x_1]$ and $P_1 = \mathbb{E}[x_1 x_1^T]$. The matrices Σ_w and Σ_v are the process and measurements noise covariance matrices, respectively.

If the system is detectable, the above recursion converges to the steady state $\lim_{k\to\infty} P_k = P$, where P can be obtained as the solution to an algebraic Riccati equation [14]. For the ease of presentation, we assume that the attacker uses a steady state Kalman filter, and we adopt trace(P) to evaluate the complexity of the attacker's reconstruction of the state. Thus

$$\xi(\Omega_\mu(\rho)) = \text{trace}(P). \tag{15}$$

3) Platform Schedulability: The encryption/decryption of messages puts overhead on computation and communication, and may have significant impact on platform schedulability as modeled below.

a) Decryption tasks: Every encrypted message m_i requires a decryption task ${\tau_d}^i$, with worst case execution time denoted by $C^i_{\tau_d}$. We use d_{m_i} to denote the decryption time of message m_i. For un-encrypted messages, we simply set $C^i_{\tau_d}$ to 0. The computation of $C^i_{\tau_d}$ thus can be modeled as in (16), where o_{m_i} denotes whether the message is encrypted and d_{m_i} is a function of encryption key length l_{key} and message length l_{m_i}. The period of ${\tau_d}^i$, denoted as $T^i_{\tau_d}$, is equal to the message period T_{m_i} as shown in the following equation:

$$C^i_{\tau_d} = d_{m_i} o_{m_i} \tag{16}$$

$$d_{m_i} \sim (l_{\text{key}}, l_{m_i}) \tag{17}$$

$$T^i_{\tau_d} = T_{m_i}. \tag{18}$$

System scheduling has to ensure the functional dependencies among tasks, which include the dependencies between decryption tasks and corresponding control tasks. Later in our automotive domain formulation, where fixed-priority preemptive scheduling is assumed, we achieve this by setting the priorities of decryption tasks higher than the priorities of control tasks.

b) Computation resource utilization: The original control tasks and the added decryption tasks are all allocated to a single computation unit. The constraint for computation resource utilization U_e is shown in (19). As defined earlier, $\mathcal{T}_\mathcal{C}$ represents the set of control tasks, where each control task τ^i_c has a worst case execution time $C^i_{\tau_c}$ and a period $T^i_{\tau_c}$. $\mathcal{T}_\mathcal{D}$ represents the set of decryption tasks, with $C^j_{\tau_d}$ and $T^j_{\tau_d}$ similarly defined. U_{e0} (between 0 and 1) represents the utilization bound set by design requirement

$$U_e = \sum_{\tau^i_c \in \mathcal{T}_\mathcal{C}} \frac{C^i_{\tau_c}}{T^i_{\tau_c}} + \sum_{\tau^j_d \in \mathcal{T}_\mathcal{D}} \frac{C^j_{\tau_d}}{T^j_{\tau_d}} \le U_{e0}. \tag{19}$$

c) Communication resource utilization: The constraint on communication resource utilization U_c is shown in (20), where the message transmission time C_{m_i} depends on the message length l_{m_i}, selection for encryption o_{m_i}, encryption key length l_{key}, and link data rate R

$$U_c = \sum_{m_i \in \mathcal{M}} \frac{C_{m_i}}{T_{m_i}} \le U_{c0} \tag{20}$$

$$C_{m_i} \sim \left(l_{m_i}, l_{\text{key}}, R, o_{m_i}\right). \tag{21}$$

d) End-to-end path latency: The end-to-end latency along a control path p (from sensor s_i to control task c_j to actuator a_k) is modeled in (22). In the formulation, $t^i_{\tau_s}$, $t^m_{\tau_d}$, $t^j_{\tau_c}$, and $t^k_{\tau_a}$ represent the maximum latency for sensing, decryption, control execution, and actuation, respectively. $t_{m_{s\to c}}$ represents the message transmission latency between sensor s_i and control task c_j, and $t_{m_{c\to a}}$ represents the message transmission latency between control task c_j and actuator a_k

$$l_{p\left(\tau^i_s, \tau^j_c, \tau^k_a\right)} = t^i_{\tau_s} + t_{m_{s\to c}} + t^m_{\tau_d} + t^j_{\tau_c} + t_{m_{c\to a}} + t^k_{\tau_a}. \tag{22}$$

III. Automotive Domain Codesign

To demonstrate the effectiveness of our approach, we customize and refine the general codesign formulation introduced in Section II to automotive systems.

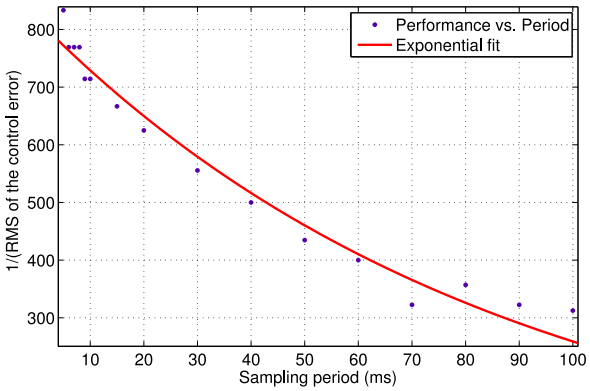


Fig. 4. Fitting the control performance of the Simulink electrohydraulic servo example for different sampling periods with an exponential decay function.

A. Customization and Refinement of the General Formulation

1) Control Performance Refinement: In this paper, we adopt exponential decay functions as control performance formulation in the automotive domain refinement. This is because for many automotive systems, the relation between control performance and sampling period can be captured by these functions with sufficient accuracy for our codesign.

As an example, we studied the model of an automotive electrohydraulic servomechanism controlled by a pulse-width modulated solenoid in the Simulink library [19]. It is a feedback control loop with a high-level controller collecting data from sensors and sending commands to actuators. The control loop may conceivably be implemented in a distributed fashion, sharing sensors, actuators, and computation node with other control loops. We can change the sampling period of the control loop, and measure its performance as the reciprocal of the root mean square (RMS) of the difference between the actuator position and the reference position (i.e., the error in the actuator position) of the electrohydraulic servo over the simulation process.

Fig. 4 shows this control performance measurement for different sampling periods. When the period increases, the performance decreases with larger error in the actuator position (i.e., larger RMS value). An exponential decay function can be used to approximate (fit) the functional dependency of the control performance on the sampling period. As shown in the figure, even when an exponential control cost function cannot be determined analytically, it is still possible to determine the parameters by fitting the cost values obtained through simulation runs for different sampling period values. In this case, the exponential fitting is very close to the simulation data, with an R-squared value of 0.972 (1 is a perfect fit). We have also conducted experiments on a fuel control system example and an engine speed control example in the Simulink library, and observed similar exponential decay trend between control performance and sampling period (with R-squared values of 0.994 and 0.996 in the exponential fittings, respectively).

We further normalize the performance of each control task with respect to its performance under an initial period $T^{i0}_{\tau_c}$

(which is obtained by solving the entire problem without encryption)

$$J^i_{\tau_c} = \alpha^{-\beta\left(T^i_{\tau_c} - T^{i0}_{\tau_c}\right)}. \tag{23}$$

The normalization is for fair consideration when performances of multiple control tasks are averaged to get overall system control performance, as shown in (8).

2) Security Level Refinement: For automotive systems, the presence of measurement noise motivates the use of the Kalman filter based approach to quantify the system security level.[4] For simplicity, we assume that all sensing messages for a control task reveal equal amount of information of the system state, that is, yield the same estimation error covariance matrix. Our approach can be extended to the case of inhomogeneous measurement channels at the cost of a more involved notation. We use Kal(n) to represent the Kalman filter performance with any n measurement messages, i.e., how easy it is to reconstruct the system state from the information of n messages. The security level defined in (9) can be refined to (24) in below. $N^i_{\tau_c}$ denotes the total number of sensing messages for task τ^i_c, out of which $n^i_{\tau_c}$ messages are encrypted. $P(n^i_{\tau_c}, \mu)$ denotes the probability that μ encrypted messages (out of $n^i_{\tau_c}$) are hacked by an attacker. We assume the attacker uses brute-force attack by randomly guessing the key with probability $2^{-l_{\text{key}}}$ to decrypt one message in one try.[5] Kal$(N^i_{\tau_c} - n^i_{\tau_c} + \mu)$ is the Kalman filter performance with the observation of μ hacked messages and $N^i_{\tau_c} - n^i_{\tau_c}$ un-encrypted messages (all un-encrypted messages are assumed as observable to the attacker)

$$S^i_{\tau_c}\left(n^i_{\tau_c}\right) = \sum_{\mu=0}^{n^i_{\tau_c}} \binom{n^i_{\tau_c}}{\mu} P\left(n^i_{\tau_c}, \mu\right) \text{Kal}\left(N^i_{\tau_c} - n^i_{\tau_c} + \mu\right) \tag{24}$$

$$P\left(n^i_{\tau_c}, \mu\right) = 2^{-\mu l_{\text{key}}} \left(1 - 2^{-l_{\text{key}}}\right)^{n^i_{\tau_c} - \mu}. \tag{25}$$

For fair comparison, the security level of each control task is normalized with respect to the maximum security level the control task may obtain

$$S'^i_{\tau_c}\left(n^i_{\tau_c}\right) = S^i_{\tau_c}\left(n^i_{\tau_c}\right) \Big/ \max_{n^i_{\tau_c} \in \left[0, N^i_{\tau_c}\right]} \left\{S^i_{\tau_c}\left(n^i_{\tau_c}\right)\right\}. \tag{26}$$

The system security level is denoted by the task that has the lowest security level as show in (11).

3) Platform Schedulability Refinement: In this paper, we consider a single controller area network (CAN) [20] or controller area network with flexible data-rate (CAN-FD) [21] bus as the communication medium. We use Bus_speed to denote the bus speed (bits/ms). Similar to [22] and [23], we adopt the KASUMI encryption algorithm with encryption speed at En_speed bits/ms, decryption speed De_speed bits/ms, and block size at B_size.

[4]In the experiments, we also evaluate the security level using observability Gramian. However, we think Kalman filter is a more suitable measurement for security level in automotive systems as it directly addresses measurement noises.

[5]Depending on the systems, more sophisticated attacks may be applied with different success probability. We plan to study some of those cases in future work, and our codesign formulation will still apply.

Based on the KASUMI algorithm, the size of an encrypted message should be elongated to $n * \text{B\_size}$ (n is an integer), thus the modified transmission time of an encrypted message m_i, denoted as c_{m_i}, should be calculated as (27), where C_{m_i} denotes the original transmission time and o_{m_i} denotes whether m_i is encrypted

$$c_{m_i} = (1 - o_{m_i}) C_{m_i} + o_{m_i} \frac{\left\lceil \frac{C_{m_i} \cdot \text{Bus\_speed}}{\text{B\_size}} \right\rceil \text{B\_size}}{\text{Bus\_speed}}. \tag{27}$$

The encryption time of message m_i, denoted as e_{m_i}, is computed according to (28). Similarly, the decryption time of message m_i, denoted as d_{m_i}, is computed according to (29). The worst case execution time of decryption task τ^i_d, denoted as $c^i_{\tau_d}$, is calculated as

$$e_{m_i} = \frac{C_{m_i} \cdot \text{Bus\_speed}}{\text{En\_speed}} \tag{28}$$

$$d_{m_i} = \frac{C_{m_i} \cdot \text{Bus\_speed}}{\text{De\_speed}} \tag{29}$$

$$c^i_{\tau_d} = o_{m_i} d_{m_i}. \tag{30}$$

The execution time of sensing task τ^i_s, denoted as $c^i_{\tau_s}$, is elongated as (31), where every encrypted message sent by sensor τ^i_s introduces its encryption overhead

$$c^i_{\tau_s} = C^i_{\tau_s} + \sum_{\tau^i_s \in \text{src}(m_i)} o_{m_i} e_{m_i} \le T^i_{\tau_s}. \tag{31}$$

We use task set $\mathcal{T}_{\mathcal{CD}} = \mathcal{T}_{\mathcal{C}} \cup \mathcal{T}_{\mathcal{D}}$ to denote all the tasks allocated on the shared computation node, including control tasks and decryption tasks. For automotive systems, we assume fixed-priority preemptive scheduling. We assign priorities following: 1) the priorities of decryption tasks are higher than control tasks to guarantee the correct execution order and 2) for tasks with the same type, the ones with shorter periods are assigned with higher priorities following the commonly-used rate-monotonic scheduling [24].

We conduct response time analysis to check platform schedulability, using techniques similarly as in [25] and [26]. Task response time denotes the longest time it may take to complete the task, and should be less or equal to task period. For system with pre-emptive fixed-priority scheduling, task response time contains the computation time requirement from the task itself and the interference from higher priority tasks. Specifically, the task response time $r^i_{\tau_c}$ of control task τ^i_c is shown below in (32). The set hp(τ^i_c) contains all the higher priority tasks compared to task τ^i_c. The first term $C^i_{\tau_c}$ is the worst case execution time of task τ^i_c. The second term is the summation of all the time τ^i_c being pre-empted by higher priority tasks on the same computation unit. The constraint between task response time and task period is shown in the following equation:

$$r^i_{\tau_c} = C^i_{\tau_c} + \sum_{\tau^k \in \text{hp}(\tau^i_c) \cap \mathcal{T}_{\mathcal{CD}}} \left\lceil \frac{r^i_{\tau_c}}{T^k_\tau} \right\rceil C^k_\tau \tag{32}$$

$$r^i_{\tau_c} \le T^i_{\tau_c}. \tag{33}$$

The task response time of decryption task τ^i_d is shown below. If message m_i is not encrypted, the corresponding

decryption task τ_d^i should not exist, thus the task response time of task τ_d^i should be 0

$$r^i_{\tau_d} = o_{m_i} \cdot \left[C^i_{\tau_d} + \sum_{\tau^k \in \text{hp}(\tau_d^i) \cap \mathcal{T}_{\mathcal{CD}}} \left\lceil \frac{r^i_{\tau_d}}{T^k_\tau} \right\rceil C^k_\tau \right] \le T^i_{\tau_d}. \quad (34)$$

For messages transmitted through the CAN or CAN-FD bus, non-preemptive fixed-priority scheduling is applied. The response time r_{m_i} of message m_i is shown in (35), where c_{m_i} denotes the worst-case transmission time of message m_i. Because message transmitted on CAN bus is not pre-emptable, a message may have to wait for a blocking time B_{m_i}, which is calculated as $\max_{j \in \text{lp}(i)} c_{m_j}$, where $\text{lp}(i)$ is the set of all lower priority messages that are allocated on the same bus with m_i. Similarly, message m_i itself is not subject to pre-emption from higher priority messages therefore the inferences from higher priority messages can only occur within $r_{m_i} - c_{m_i}$ time intervals

$$r_{m_i} = c_{m_i} + B_{m_i} + \sum_{m_k \in \text{hp}(m_i)} \left\lceil \frac{r_{m_i} - c_{m_i}}{T_{m_k}} \right\rceil c_{m_k} \le T_{m_i}. \quad (35)$$

Finally, the end-to-end latency is shown in (36), where path p is represented by the link $s_i \to c_j \to a_k$. Message m_{s_i,c_j} is transmitted between sensing task τ_s^i and control task τ_c^j, and decryption task τ_d^i for decrypting m_{s_i,c_j} may be added to the path. Message m_{c_j,a_k} is transmitted between control task τ_c^j and actuation task τ_a^k. Because of the asynchronous nature of the automotive embedded systems, in the worst case, when any task/message on the path completes its execution/transmission, the receiving message/task might have just been activated and will need to wait for the next activation to continue processing, where the wait time can be arbitrarily close to its period. For the sensing task, the arrival of the external event has the similar effect, i.e., it may just have missed the activation of the sensing task. Thus, in the worst case scenario, the periods of all the tasks and messages on the path should be added into the latency. For more detailed discussion (and the cases where such worst case bound can be reduced), please refer to [25]

$$\begin{aligned} l_{p\left(\tau_s^i \to \tau_c^j \to \tau_a^k\right)} = {} & c^i_{\tau_s} + T^i_{\tau_s} + r_{m_{s_i,c_j}} + T_{m_{s_i,c_j}} + r^i_{\tau_d} \\ & + o_{m_i} T^i_{\tau_d} + r^j_{\tau_c} + T^j_{\tau_c} + r_{m_{c_j,a_k}} + T_{m_{c_j,a_k}} \\ & + c^k_{\tau_a} + T^k_{\tau_a}. \end{aligned} \quad (36)$$

B. Optimization With Simulated Annealing

The final optimization formulation for this automotive system is shown in below, refined from the general formulation in Section II

$$\text{maximize: } J = \frac{1}{|\mathcal{T}_\mathcal{C}|} \sum_{\tau_c^i \in \mathcal{T}_\mathcal{C}} \omega^i_{\tau_c} \alpha^{-\beta\left(T^i_{\tau_c} - T^{i0}_{\tau_c}\right)} \quad \text{s.t.} \quad (37)$$

$$\forall \tau_c^i \in \mathcal{T}_\mathcal{C},\ S^i_{\tau_c} \ge S_0 \text{ (security constraint)} \quad (38)$$

$$\forall \tau_c^i \in \mathcal{T}_\mathcal{C},\ r^i_{\tau_c}/T^i_{\tau_c} \le 1 \text{ (computational constraint)} \quad (39)$$

$$\forall \tau_s^i \in \mathcal{T}_\mathcal{S},\ c^i_{\tau_s}/T^i_{\tau_s} \le 1 \text{ (computational constraint)} \quad (40)$$

$$\forall m_i \in \mathcal{M},\ r_{m_i} \le T_{m_i} \text{ (communicational constraint)} \quad (41)$$

Algorithm 1: Simulated Annealing()

Input: task graph, task execution time, message length, message period, priority, key length, encryption_speed, bus_speed
Output: sampling period $\overrightarrow{T_{\tau_c}}$, encryption assignment $\overrightarrow{o_m}$, best control performance found

1 $curSol \leftarrow initial\_solution()$; $heat \leftarrow heat_0$; $nIter \leftarrow 0$;
2 **while** $nIter < maxIter \wedge heat > heat_{final}$ **do**
3 $nTry \leftarrow 0$;
4 **while** $nTry < maxTry$ **do**
5 $nTry{+}{+}$; $i \leftarrow randIdx$;
6 **if** $move = randomMove1$ **then**
7 $tmpSol.o_{m_i} \leftarrow !curSol.o_{m_i}$;
8 **end**
9 **else if** $move = randomMove2$ **then**
10 $tmpSol.T^i_{\tau_c} \leftarrow curSol.(T^i_{\tau_c} + \delta(r^i_{\tau_c} - T^i_{\tau_c}))$;
11 **end**
12 $tmpCost \leftarrow \alpha_1 \cdot obj^{-1} + \alpha_2 \cdot timeVio + \alpha_3 \cdot secuVio$;
13 **if** $tmpCost < minCost$ **then**
14 Accept $tmpSol$;
15 **end**
16 **else if** $randNum < e^{\gamma(curCost - tmpCost)/heat}$ **then**
17 Conditionally accept $tmpSol$;
18 **end**
19 **end**
20 $heat \leftarrow heat * coolFactor$; $nIter{+}{+}$;
21 **end**

$$\forall p \in \mathcal{P},\ l_p \le D_p \text{ (end-to-end latency)} \quad (42)$$

where the computation of variables such as security level $S^i_{\tau_c}$, response time $r^i_{\tau_c}$, and end-to-end latency l_p can be referred to the formulations in Section III-A.

The above formulation is complex, and direct use of a generic nonlinear solver may be intractable for industrial size problems. Instead, we implement a simulated annealing (SA) algorithm to explore acceptable feasible solutions.

The algorithm shown in Algorithm 1 is based on the standard SA procedure. We first set every message not to be encrypted and obtain the initial period by solving the problem without encryption (line 1). Then we start from an initial temperature heat_0 to iteratively search the design space until the number of iterations nIter exceeds a preset limit maxIter or the temperature falls below a preset final temperature $\text{heat}_{\text{final}}$. During each iteration, we randomly explore changes to the current solution curSol by considering either: 1) selecting a message and changing its encryption status $\text{curSol}.o_{m_i}$ or 2) selecting a control task and changing its period $\text{curSol}.T^i_{\tau_c}$ (lines 5–8). We evaluate the cost of such changes tmpCost, which is based on the objective value obj, the penalty proportional to the number of schedulability violations timeVio and to the number of security violations secuVio (line 12). If the new cost is smaller than the previous minimum cost minCost, the new solution tmpSol will be accepted immediately; otherwise it will be accepted with a transition probability $P = \exp^{\gamma(\text{curCost} - \text{tmpCost})/\text{heat}}$, where γ is a parameter and heat is the current temperature. After each iteration, heat is lowered with a cooling factor coolFactor.

We properly tune the values of parameters α_1, α_2, α_3, δ, γ, and coolFactor to improve the SA performance.

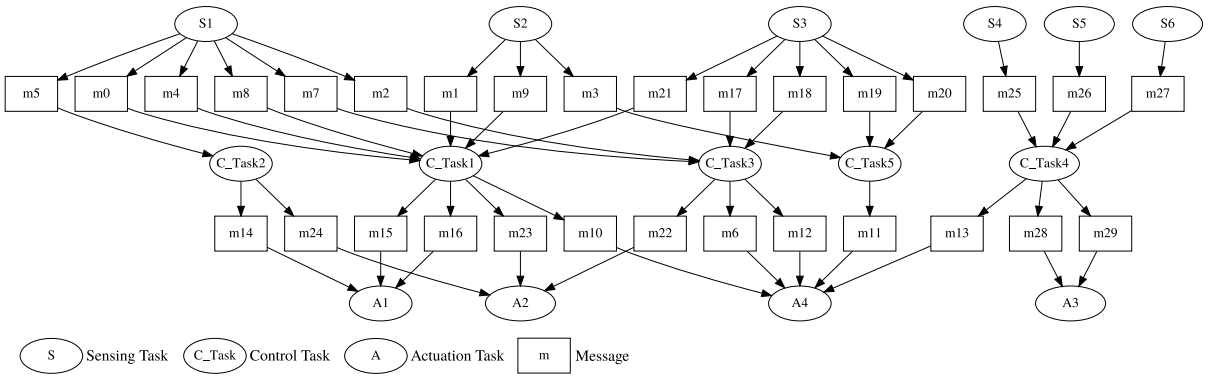


Fig. 5. Modified automotive subsystem used in the case study.

IV. Automotive Case Study Results

To evaluate the effectiveness of our codesign methodology and its refinement in automotive domain, we conducted experiments for an industrial automotive system example and a set of synthetic examples. All experiments are run on an Intel Core i7 CPU with 12 GB memory. The results are discussed in below.

A. Industrial Example

We first conduct a case study that is derived from a subsystem of an experimental vehicle with active safety functions, similarly as the one used in [25] and [26]. The vehicle supports distributed functions with end-to-end computations collecting data from 360° sensors and sending commands to the actuators, consisting of the throttle, brake, and steering subsystems and of advanced human–machine interface devices. Examples of active safety functions include adaptive cruise control, lane departure warning or lane keeping systems. These functions are deployed together in a car electronics system, sharing the sensing and actuation layers and possibly also intermediate processing stages, such as the sensor fusion and object detection functions or the actuator arbitration layers. The result is a complex graph of functions (programmed as tasks) with a high degree of communication dependency and deadlines on selected pairs of endpoints. In this case study, we select a subsystem of those functions, including their tasks and communication signals.[6] The example consists of 14 tasks (including 6 sensing tasks, 5 control tasks, and 4 actuation tasks), 17 messages from sensing tasks to control tasks, and 13 messages from control tasks to actuation tasks. As we explore the encryption of sensing messages, up to 17 additional encryption tasks may be added. The structure of the example is shown in Fig. 5. The task execution times are in the range of 0.2 to 20 ms, and the initial task periods are in the range of 10 to 100 ms. The message lengths are in the range of 1–64 bits, and the message periods are in the range of 10–100 ms. In this paper, we derived system dynamics from two automotive examples in the Simulink library (these systems and their derivations are used for the control loops in the industrial example and the synthetic examples). The first system dynamics is linearized from Simulink Vehicle Suspension Model, and the equation is

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -50.4 & -4.671 & 7.105 * 10^{-15} & -0.1429 \\ 0 & 0 & 0 & 1 \\ 7.105 * 10^{-15} & -0.25 & -81.67 & -7.5 \end{bmatrix} x(t) + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -21.6 & -1.929 & 24 & 2.143 & -28.8 & -2.743 & -24 & -2.286 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 42 & 3.75 & -46.67 & -4.167 & -42 & -4 & -35 & -3.333 \end{bmatrix} u(t) + w(t). \quad (43)$$

The second dynamics is linearized from Simulink Engine Speed model. The equation is

$$\dot{x}(t) = \begin{bmatrix} -7.146 & -0.02545 & 0 & 0 \\ 592.9 & 0.4642 & -2.323 * 10^{6} & 0 \\ 0.2043 & 0.0001861 & -400.1 & -5.335 * 10^{4} \\ 0 & 0 & 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 & 0.6771 \\ -7.143 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} u(t) + w(t). \quad (44)$$

There are five control loops in the system (corresponding to five control tasks). The system control performance, calculated as in Section III-A, is in the range of [0, 1] (where 1 represents the best possible performance obtained without encryption). The α value in (23) is set as the Euler's number e and β is set as 1.

The attacker conducts eavesdropping on 17 sensing messages. The security level of every control loop is measured by the complexity for the Kalman filter to retrieve information, and normalized to be within [0, 1] as shown in Section III-A, where 1 represents the best possible security level, i.e., every sensing message of the control loop is encrypted. The system security level is the minimum of all control loops' security level, and is within the range of [0, 1]. System security level reaches 1 when all 17 messages are encrypted, and is 0 when none is encrypted.

In our experiments, we explore the selection of messages for encryption and the assignment of control sampling periods, to address security level together with control performance while guaranteeing platform schedulability.

[6] Addressing the entire system available requires models for more complex functional graph and multiple computation units (planned in the future work).

TABLE I
SECURITY LEVEL OF CONTROL LOOP 3 WITH DIFFERENT NUMBER OF ENCRYPTED MESSAGES

# encr. msgs	0	1	2	3	4
security level	0	0.2902	0.4349	0.4371	1

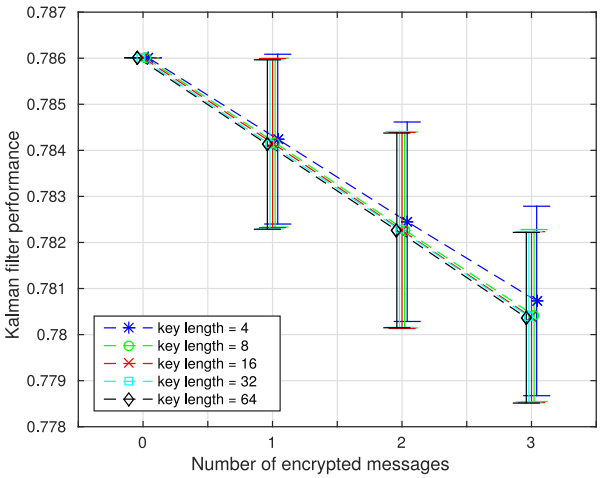


Fig. 6. Mean and standard deviation of the Kalman filter performance under different number of encrypted messages and different key lengths (simulations of control loop 3).

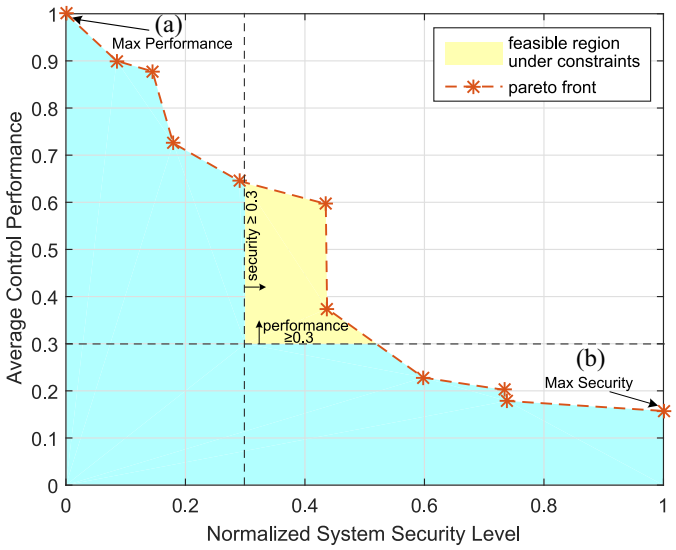


Fig. 7. Pareto front between normalized control performance and security level for the industrial example. An example feasible region denotes all feasible solutions under requirement that control performance ≥0.3 and security level ≥0.3.

1) Tradeoff Between Control Performance and System Security Level: As we explore the selection of sensing messages for encryption, the security levels of control loops are significantly affected. For instance, Table I shows the security level of control loop 3 (the control loop associated with control task C_Task 3 in Fig. 5) when different number of sensing messages is encrypted, as directly computed from (26). The system dynamics with regard to control loop 3 is shown in (43). The measurement noise is set to $0.01 * \text{eye}(4, 4)$.

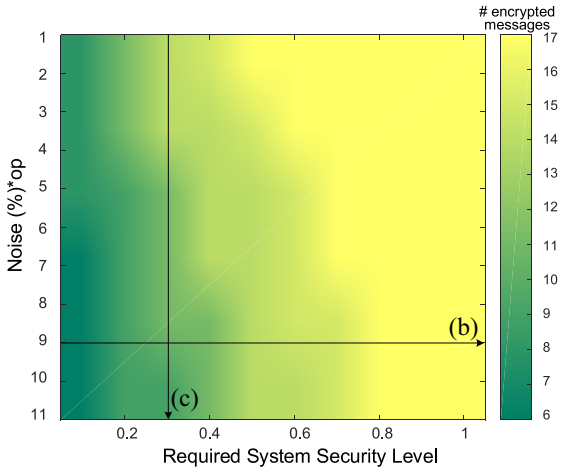
We further conduct simulations to report the actual mean and variance of the Kalman filter performance, i.e., the attacker performance, as a function of the number of encrypted messages and key length (varied from 4 to 64). For each configuration for control loop 3, we run 1000 simulations to compute the mean and the variance of the Kalman filter performance, measured as the inverse of the trace of the Kalman covariance error. As shown in Fig. 6, the means are noted with markers while the standard deviations (square roots of the variances) are shown as the vertical bars. We can see from the figure that as the number of encrypted messages increases or the key length increases, the attacker performance decreases, i.e., the control loop has a higher security level.

On the other hand, encrypting and decrypting messages introduces significant timing overhead, which may cause the increase of control sampling periods due to schedulability constraints and thus reduce the control performance. Based on our formulation introduced in Section III-A and using the SA algorithm shown in Section III-B, we are able to explore the design space while quantitatively analyzing the tradeoffs between control performance and security level. Fig. 7 shows the Pareto front between the two normalized metrics for the automotive case study. The relative noise of each sensing message (i.e., how much the measurement deviates from the true value) is set as 10% of the operation point—the noise impact on security will be discussed later in this section.

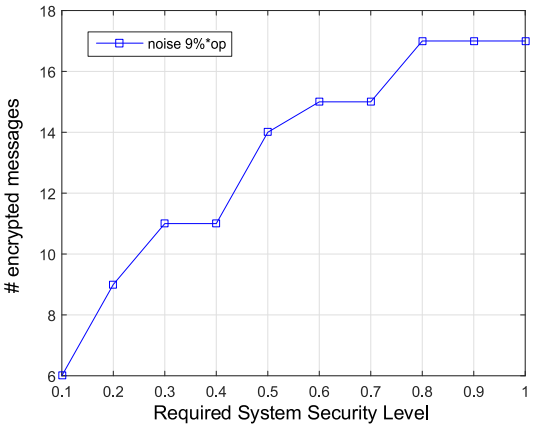
From Fig. 7, we can clearly see the tradeoffs between control performance and security level. During design, constraints on these two metrics may be set according to system requirements. The Pareto front generated by our approach will provide a feasible region that is important for making decision choices. For instance, an example feasible region is shown in the figure, under the requirements that the system control performance should be no less than 0.3 and the system security level should be no less than 0.3. Without our codesign approach, it is impossible to identify the feasible designs under such requirements. Instead, the designers might get a solution that violates security requirement if they only optimize for control performance [point (a) in Fig. 7], or a solution that violates performance requirement if they simply choose to encrypt all messages [point (b) in Fig. 7]. This shows the importance of our codesign framework.

2) Impact of Sensing Noise on System Security Level and the Number of Encrypted Messages: In our experiments, we observe that the noise on sensor measurements has a significant impact on system security level. Intuitively, as the attacker tries to reconstruct the system state from hacked sensing messages and un-encrypted messages, the lower the sensing noise, the easier it is for the attacker (thus the system is less secure). To quantitatively analyze this relation, we conduct a series of experiments: we set the sensing noise to different levels [measured by the relative error of the sensing measurements with respect to the control operating point (op)], and evaluate how many messages need to be encrypted for certain system security level requirement, while maximizing control performance.

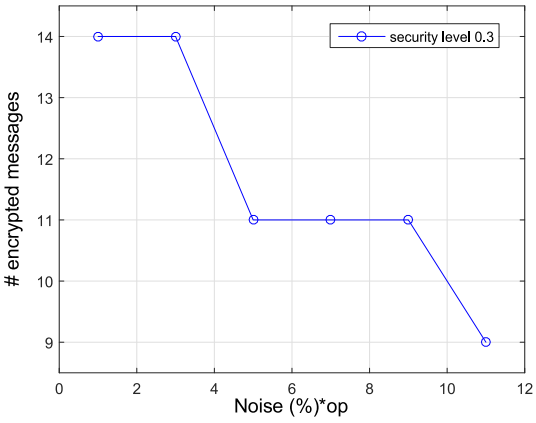
A heat map demonstrating the relation among noise level, system security level, and the number of encrypted messages is



(a)



(b)



(c)

Fig. 8. Impact of sensing noise on security level and encrypted messages for the industrial example. (a) Number of messages encrypted under different noise levels and different security requirements. (b) When the noise level is fixed at 9%*op: as the required security level increases, the required number of encrypted messages increases. (c) When the security level is 0.3, as the noise level increases, the required number of encrypted messages decreases.

shown in Fig. 8(a). We can clearly see the trend that when the noise level increases, we need fewer messages to be encrypted to reach certain security level. Fig. 8(b) and (c) further extract one horizontal line and one vertical line from the heat map, respectively. For (b), we can see that as the security level

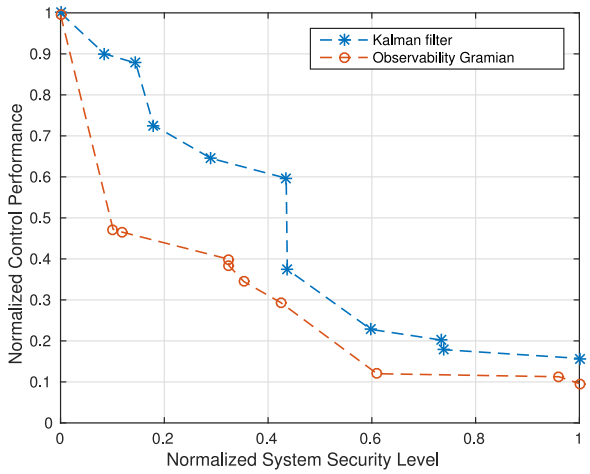


Fig. 9. Pareto front between normalized control performance and security level using observability Gramian and Kalman filter for the industrial example.

requirement increases, we need to encrypt more messages. More interestingly, for (c), we can see that as the noise level increases, we may encrypt fewer messages.

3) Security Evaluation Using Observability Gramian: As stated in Section II-B, we may also use the observability Gramian to measure the complexity for an attacker to estimate the system states, and further compute the system security level. In this experiment, we conduct experiments with observability Gramian as the security level measurement, and evaluate its tradeoff with the control performance for the industrial example. Fig. 9 shows such tradeoff, along with the tradeoff using Kalman filter (which is the same as the one from Fig. 7). The two security level measurements, observability Gramian and Kalman filter, are based on different perspectives and consequently provide different values. Despite this, the two Pareto fronts show similar tradeoff trend between control performance and security level. Furthermore, for automotive systems with measurement noise, we think Kalman filter-based approach is a more suitable measurement with its consideration of measurement noises.

B. Synthetic Examples

For more comprehensive study of our codesign approach, we conduct a set of experiments with synthetic examples that have varying number of tasks, messages, execution times, and periods.

We randomly generate task graphs that have the same structure as the system model in Fig. 1. Specifically, we first vary the number of control tasks from 5 to 25, and then randomly generate a number of sensing tasks and actuation tasks as well as their connections with the control tasks (the numbers of sensing tasks and actuation tasks are proportional to the number of control tasks in average). The period of each task is randomly generated between 10 to 100 ms, and the execution time is randomly generated within the period. When randomly generating the task periods and execution times, we keep the total utilization of the computation unit around 60% (before adding the decryption tasks). Each message may have multiple successive control tasks, and each message length is randomly

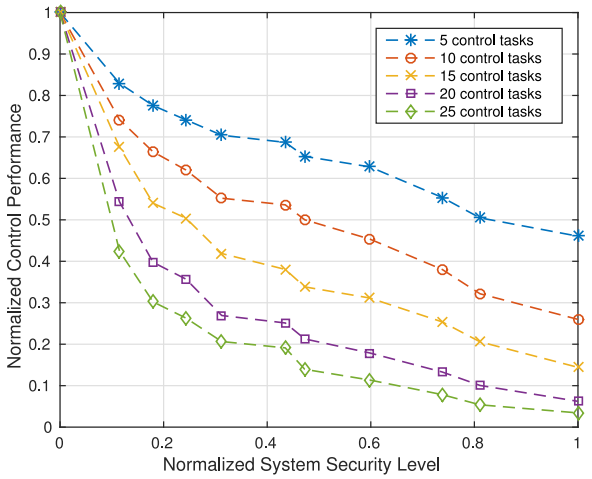


Fig. 10. Pareto front between normalized control performance and security level for synthetic examples with different number of control tasks (using Kalman filter-based security level measurement).

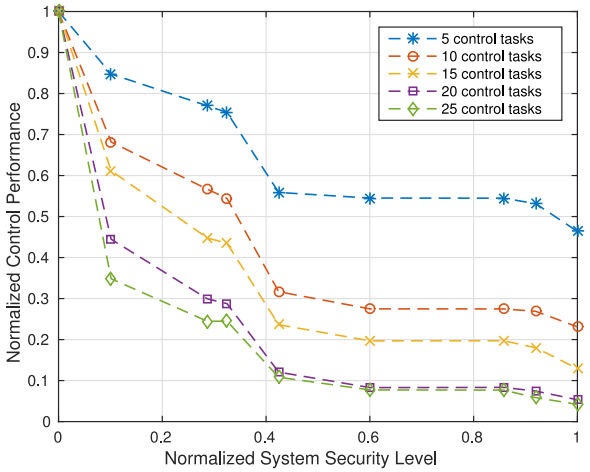


Fig. 11. Pareto front between normalized control performance and security level for synthetic examples with different number of control tasks (using observability Gramian-based security level measurement).

generated between 1 to 32 bits. We set the message transmission speed at 1000 bit/ms, and message decryption speed at 250 bit/ms.

Similarly as for the industrial example, we evaluate the tradeoff between control performance and security level in codesign for various synthetic examples. Fig. 10 shows the Pareto fronts for synthetic examples with different number of control tasks, using Kalman filter as the security level measurement. Every point in this figure is the average of ten randomly generated examples. We can see that the tradeoff between control performance and security level is similar to the industrial example. Furthermore, as the number of control tasks increases, the control performance decreases faster with respect to increasing security level (i.e., the Pareto front curve is lower as shown in the figure). This is because to achieve the same system security level, more messages need to be encrypted for a larger task set. This leads to more decryption tasks added to the computation unit, and consequently harder scheduling (even for similar level of utilization), and eventually longer periods for control tasks and worse control performance.

We also conduct experiments using observability Gramian as the security level measurement for the synthetic examples. The results in Fig. 11 demonstrate the similar trend as using Kalman filter.

The runtime of our algorithm depends on the problem size (in particular the number of control tasks) and the tuning parameters in SA. In Table II, we record the runtime of our algorithm for the synthetic examples under different sizes of control task set and the same tuning parameters. The time we record is the average time for running the algorithm once, i.e., for obtaining one point in Figs. 10 and 11. As the number of control tasks increases, we let the numbers of messages, sensing tasks and actuation tasks increase proportionally. Note that the observability Gramian and Kalman filter performance for each control loop under different number of encrypted sensing messages are calculated and stored in arrays before running the SA algorithm. The calculation time of both metrics is small compared to SA. Therefore, we only record the average runtime of the SA algorithm, which is almost the same for the two metrics.

TABLE II
ALGORITHM RUNTIME OF SA UNDER DIFFERENT NUMBER OF CONTROL TASKS

control tasks #	5	10	15	20	25
runtime (s)	43	126	181	482	745

V. CONCLUSION

We propose a cross-layer codesign framework that addresses security and control performance together while guaranteeing platform schedulability for cyber-physical systems. In the framework, we identify the key interface variables among security, performance, and schedulability, in particular the sampling periods and the selection of sensing messages for encryption. We quantitatively model the relation between message encryption and system security level, and model the impact of message encryption and sampling periods on control performance and platform schedulability. The general framework is refined to automotive systems, and an SA algorithm is developed for exploring the design space based on the refined codesign formulation. An automotive case study and synthetic examples demonstrate the effective of our approach in facilitating design space exploration. Future work includes considering other types of attacks, other security techniques, and other types of architecture platforms and functional graph. We also plan to develop efficient heuristic optimization algorithms to complement the SA algorithm.

REFERENCES

[1] E. A. Lee, "Cyber physical systems: Design challenges," in *Proc. IEEE 11th Int. Symp. Object Oriented Real-Time Distrib. Comput. (ISORC)*, Orlando, FL, USA, 2008, pp. 363–369.

[2] R. Poovendran *et al.*, "Special issue on cyber-physical systems [scanning the issue]," *Proc. IEEE*, vol. 100, no. 1, pp. 6–12, Jan. 2012.

[3] S. Karnouskos, "Stuxnet worm impact on industrial cyber-physical system security," in *Proc. IEEE 37th Annu. Conf. Ind. Electron. Soc. (IECON)*, Melbourne, VIC, Australia, 2011, pp. 4490–4494.

[4] K. Koscher *et al.*, "Experimental security analysis of a modern automobile," in *Proc. IEEE Symp. Security Privacy (SP)*, Oakland, CA, USA, 2010, pp. 447–462.

[5] S. Checkoway *et al.*, "Comprehensive experimental analyses of automotive attack surfaces," in *Proc. USENIX Security Symp.*, San Francisco, CA, USA, 2011, p. 6.

[6] J. Slay and M. Miller, "Lessons learned from the maroochy water breach," in *Critical Infrastructure Protection*, D. E. Goetz and P. S. Shenoi, Eds. Boston, MA, USA: Springer, 2008, pp. 73–82.

[7] F. Pasqualetti, F. Dorfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE Trans. Autom. Control*, vol. 58, no. 11, pp. 2715–2729, Nov. 2013.

[8] A. Gupta, C. Langbort, and T. Basar, "Optimal control in the presence of an intelligent jammer with limited actions," in *Proc. IEEE 49th Conf. Decis. Control (CDC)*, Atlanta, GA, USA, 2010, pp. 1096–1101.

[9] Z.-H. Pang, G. P. Liu, and Z. Dong, "Secure networked control systems under denial of service attacks," in *Proc. 18th IFAC World Congr.*, Milan, Italy, 2011, pp. 8908–8913.

[10] Y. Shoukry, J. Araujo, P. Tabuada, M. Srivastava, and K. H. Johansson, "Minimax control for cyber-physical systems under network packet scheduling attacks," in *Proc. 2nd ACM Int. Conf. High Confidence Netw. Syst.*, Berlin, Germany, 2013, pp. 93–100.

[11] C.-W. Lin, Q. Zhu, C. Phung, and A. Sangiovanni-Vincentelli, "Security-aware mapping for CAN-based real-time distributed automotive systems," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, San Jose, CA, USA, Nov. 2013, pp. 115–121.

[12] C.-W. Lin, Q. Zhu, and A. Sangiovanni-Vincentelli, "Security-aware mapping for TDMA-based real-time distributed systems," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Austin, TX, USA, Nov. 2014, pp. 24–31.

[13] F. Pasqualetti and Q. Zhu, "Design and operation of secure cyber-physical systems," *IEEE Embedded Syst. Lett.*, vol. 7, no. 1, pp. 3–6, Mar. 2015.

[14] T. Kailath, *Linear Systems*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1980.

[15] G. C. Buttazzo, *Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications*, vol. 24. New York, NY, USA: Springer, 2011.

[16] D. Seto, J. P. Lehoczky, L. Sha, and K. G. Shin, "On task schedulability in real-time control systems," in *Proc. 17th IEEE Real-Time Syst. Symp.*, Los Alamitos, CA, USA, 1996, pp. 13–21.

[17] L. Sha, X. Liu, M. Caccamo, and G. Buttazzo, "Online control optimization using load driven scheduling," in *Proc. 39th IEEE Conf. Decis. Control*, vol. 5. Sydney, NSW, Australia, 2000, pp. 4877–4882.

[18] X. Liu *et al.*, "ORTEGA: An efficient and flexible online fault tolerance architecture for real-time control systems," *IEEE Trans. Ind. Informat.*, vol. 4, no. 4, pp. 213–224, Nov. 2008.

[19] (Feb. 12, 2016). *Simulink*. [Online]. Available: http://www.mathworks.com/products/simulink/

[20] K. W. Tindell, H. Hansson, and A. J. Wellings, "Analysing real-time communications: Controller area network (CAN)," in *Proc. Real-Time Syst. Symp.*, San Juan, PR, USA, Dec. 1994, pp. 259–263.

[21] F. Hartwich, "CAN with flexible data-rate," in *Proc. 3th Int. CAN Conf. (ICC)*, Hambach, Germany, 2012, pp. 14-1–14-9.

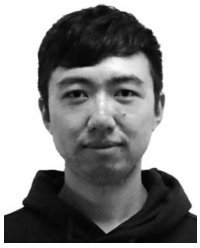
[22] D. K. Nilsson, U. E. Larson, and E. Jonsson, "Efficient in-vehicle delayed data authentication based on compound message authentication codes," in *Proc. IEEE 68th Veh. Technol. Conf. (VTC)*, Calgary, AB, Canada, Sep. 2008, pp. 1–5.

[23] O. Hyncica, P. Kucera, P. Honzik, and P. Fiedler, "Performance evaluation of symmetric cryptography in embedded systems," in *Proc. IEEE 6th Int. Conf. Intell. Data Acquis. Adv. Comput. Syst. (IDAACS)*, vol. 1. Prague, Czech Republic, 2011, pp. 277–282.

[24] C. L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *J. ACM*, vol. 20, no. 1, pp. 46–61, 1973.

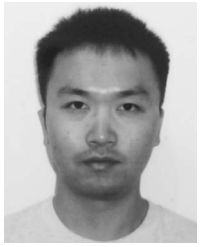
[25] A. Davare *et al.*, "Period optimization for hard real-time distributed automotive systems," in *Proc. 44th Annu. Design Autom. Conf.*, San Diego, CA, USA, 2007, pp. 278–283.

[26] Q. Zhu, H. Zeng, W. Zheng, M. D. Natale, and A. Sangiovanni-Vincentelli, "Optimization of task allocation and priority assignment in hard real-time distributed systems," *ACM Trans. Embedded Comput. Syst.*, vol. 11, no. 4, 2012, Art. ID 85.



Bowen Zheng (S'16) received the B.E. degree in communication engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2012. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of California at Riverside, Riverside, CA, USA.

His current research interests include cyber-physical system security, computer-aided design, fault-tolerance design, and embedded software synthesis.

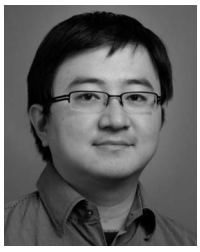


Peng Deng (S'13) received the B.E. degree in information engineering from Beihang University, Beijing, China, in 2010. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of California at Riverside, Riverside, CA, USA.

His current research interests include computer-aided design and software synthesis for cyber-physical systems.

Rajasekhar Anguluri (S'10) received the B.Tech. degree from the National Institute of Technology at Warangal, Warangal, India, in 2013. He is currently pursuing the Ph.D. degree with the Department of Mechanical Engineering, University of California at Riverside, Riverside, CA, USA.

His current research interests include estimation and decision theory with applications to cyber-physical systems security.



Qi Zhu (M'12) received the B.E. degree in computer science from Tsinghua Univerisity, Beijing, China, in 2003, and the Ph.D. degree in electrical engineering and computer science from the University of California at Berlekey, Berlekey, CA, USA, in 2008.

He is an Assistant Professor with the Department of Electrical and Computer Engineering, University of California at Riverside (UCR), Riverside, CA, USA. Prior to joining UCR, he was a Research Scientist with the Strategic CAD Labs, Intel, Santa Clara, CA, USA, from 2008 to 2011. His current research interests include model-based design and software synthesis for cyber-physical systems, cyber-physical system security, energy-efficient buildings and infrastructures, and system-on-chip design.

Fabio Pasqualetti (M'07) received the Laurea degree in computer engineering and the Laurea Magistrale degree in automation engineering from the University of Pisa, Pisa, Italy, in 2004 and 2007, respectively, and the Ph.D. degree in mechanical engineering from the University of California at Santa Barbara, Santa Barbara, CA, USA, in 2012.

He is an Assistant Professor with the Department of Mechanical Engineering, University of California at Riverside, Riverside, CA, USA. His current research interests include secure control systems, with application to multiagent networks, distributed computing, and power networks, and vehicle routing and combinatorial optimization, with application to distributed area patrolling and persistent surveillance.