2020 59th IEEE Conference on Decision and Control (CDC)
Jeju Island, Republic of Korea, December 14-18, 2020

Privacy Against Adversarial Classification in Cyber-Physical Systems

Carlos Murguia and Paulo Tabuada

***Abstract*— For a class of Cyber-Physical Systems (CPSs), we address the problem of performing computations over the cloud without revealing private information about the structure and operation of the system. We model CPSs as a collection of input-output dynamical systems (the system operation modes). Depending on the *mode* the system is operating on, the output trajectory is generated by one of these systems in response to driving inputs. Output measurements and driving inputs are sent to the cloud for processing purposes. We capture this "processing" through some function (of the input-output trajectory) that we require the cloud to compute accurately – referred here as the *trajectory utility*. However, for privacy reasons, we would like to keep the mode private, i.e., we do not want the cloud to correctly identify what mode of the CPS produced a given trajectory. To this end, we *distort* trajectories before transmission and send the corrupted data to the cloud. We provide mathematical tools (based on output-regulation techniques) to properly design distorting mechanisms so that: 1) the original and distorted trajectories lead to the *same utility*; and the distorted data leads the cloud to *misclassify the mode*.**

I. Introduction

Scientific and technological advances have led to an overwhelming amount of user data being collected and processed by hundreds of companies over the cloud. Companies mine and classify this data to provide personalized services and advertising. However, these new technologies have also led to an alarming widespread loss of privacy in society. Depending on the adversaries' resources, they may infer sensitive (private) information about the operation of systems from public data available on the internet and unsecured/public servers and communication networks. A motivating example of this privacy loss is the data collection, classification, and sharing by the Internet-of-Things (IoT) [1], which is, most of the time, done without the user's informed consent. Another example of privacy loss is the potential use of data from smart electrical meters by criminals, advertising agencies, and governments, for monitoring the presence and activities of occupants, [2]-[3]. These privacy concerns show that there is an acute need for privacy preserving mechanisms capable of handling the new privacy challenges induced by a hyperconnected world. That is why researchers from different fields (e.g., computer science, information theory, and control theory) have been attracted to the broad research area of privacy and security of Cyber-Physical Systems (CPSs), see, e.g., [4]-[28].

Carlos Murguia is with the Department of Mechanical Engineering, Eindhoven University of Technology, The Netherlands; and Paulo Tabuada is with the Department of Electrical Engineering, University of California, Los Angeles, USA. Emails: c.g.murguia@tue.nl, & tabuada@ee.ucla.edu.

This work was partially supported by the NSF awards 1740047 and 1705135, and by the UC-NL grant LFR-18-548554

In this manuscript, for a class of Cyber-Physical Systems (CPSs), we address the problem of performing computations over the cloud without revealing private information about the structure and operation of the system. That is, the objective is to have the cloud provide a service by processing system data while preventing it from learning private information. The setting that we consider is the following. The underlying physical part of the system (the system dynamics) consists of a finite collection of N *input-output dynamical systems*, Σ_i, $i \in \{1, \ldots, N\}$. Depending on the *mode* the system is operating on, sensor measurements are generated by one of these dynamical systems in response to driving inputs. Each subsystem characterizes an operation mode of the CPS. For instance, the operation of fitness trackers is based on different modes (i.e., different dynamical systems) indicating our activity level, e.g., depending whether we are walking, running, or resting, sensors/actuators embedded in the device would provide different data and this data would be consistent with the corresponding dynamical system. That is, we have a dynamical system explaining the data for walking, one for running, and one for resting. Under normal operating conditions, sensor measurements and driving inputs are sent to the cloud for monitoring or processing purposes. However, for privacy reasons, we would like to keep the mode private. To accomplish this, we use knowledge of the system dynamics to appropriately modify sensor measurements and driving inputs generated by/for system Σ_i so that the distorted data appears to have been generated by a different *target system*, Σ_j, $j \neq i$, within the operation modes of the CPS, and we send the distorted data to the cloud. The idea is that if the target system is sufficiently different (in some appropriate sense) from the mode that generated the data, the cloud would incorrectly classify the mode.

Note, however, that we do not want to overly distort the data. The main reason for sharing system data is to have the cloud provide a service by processing it. Usually, there is some function of the sensor data that we would like the cloud to compute accurately–referred here as *the utility function*. The utility function imposes a constraint on the class of systems that we can use as target systems. Concretely, we aim at modifying input-output data so that: (1) the utility function evaluated at the distorted data equals its value on the original data; and (2) the output trajectory seems to have been generated by the target system in response to driving inputs, i.e., the provided input-output data is consistent with the target system dynamics. We remark that we do not make any assumption on the classification algorithm employed by the cloud. It is unrealistic to assume we know how data is being classified. However, if one is only concerned about misclassifying specific modes of the system dynamics, arguably, mapping output trajectories and driving inputs to a different mode would lead to incorrect classification if the model of the modes that we use to design the distorting mechanism is accurate enough to capture the true dynamic behavior of the system.

Most of the work related to privacy of dynamical systems deals with keeping the system state private when output measurements and the system model are disclosed for processing purposes, see, e.g., [21]-[27]. All these manuscripts follow stochastic formulations where the objective is twofold: 1) To quantify the potential information leakage given a privacy metric (e.g., based on differential privacy [29] or information-theoretic [30]); and 2) To design randomizing mechanisms to distort data so that the distorted disclosed data provides prescribed privacy guarantees. In this manuscript, we address a fundamentally different problem. First, we consider fully deterministic systems and thus stochastic privacy metrics do not make sense in our setting. Secondly, we are not concerned with privacy of the system state per se, but it is the mode the system is operating on what we want to keep private. Because we consider LTI dynamics for each mode, all the input-output data that a mode can generate forms a linear subspace (referred here as the mode *behaviour*). So, instead of looking for the probability distribution of the noise to inject (as it is usually done in stochastic formulations), we seek distorting mechanisms, based on system-theoretic tools (output regulation), that maps data from the actual mode behaviour into the behaviour of a different *target mode*, while maintaining its utility invariant.

Notation: The notation $\text{col}(x_1, \ldots, x_n)$ stands for the column vector composed of the elements $x_1, \ldots, x_n$. This notation is also used in case the components x_i are vectors. The $n \times n$ identity matrix is denoted by I_n or simply I if n is clear from the context. Similarly, $n \times m$ matrices composed of only ones and only zeros are denoted by $\mathbf{1}_{n\times m}$ and $\mathbf{0}_{n\times m}$, respectively, or simply $\mathbf{1}$ and $\mathbf{0}$ when their dimensions are clear. Finite sequences of vectors are written as $x^N := (x(1)^\top, \ldots, x(N)^\top)^\top \in \mathbb{R}^{Nn}$ with $x(i) \in \mathbb{R}^n$, and $n, N \in \mathbb{N}$. We denote powers of matrices as $(A)^K = A \cdots A$ (K times) for $K > 0$, $(A)^0 = I$, and $(A)^K = \mathbf{0}$ for $K < 0$. Matrix $Q^+ \in \mathbb{R}^{m\times n}$ denotes the Moore–Penrose inverse of $Q \in \mathbb{R}^{n\times m}$.

II. System Description and Problem Formulation

We consider a class of cyber-physical systems whose physical part can be modeled by switching discrete-time linear systems of the form:

$$\Sigma_\rho := \begin{cases} x_\rho(k+1) = A_\rho x_\rho(k) + B_\rho u(k), \\ \quad y_\rho(k) = C_\rho x_\rho(k), \\ \quad \rho \in \mathcal{N} := \{1, 2, \ldots, N\}, \end{cases} \tag{1a}$$

$$y(k) = y_\rho(k), \tag{1b}$$

with time index $k \in \mathbb{N}$, state $x_\rho \in \mathbb{R}^{n_\rho}$, $n_\rho \in \mathbb{N}$, output $y \in \mathbb{R}^m$, $m \in \mathbb{N}$, input $u \in \mathbb{R}^l$, $l \in \mathbb{N}$, and matrices $A_\rho \in \mathbb{R}^{n_\rho \times n_\rho}$, $B_\rho \in \mathbb{R}^{n_\rho \times l}$, and $C_\rho \in \mathbb{R}^{m \times n_\rho}$. It is assumed that, for all $\rho \in \mathcal{N}$, A_ρ, C_ρ, and B_ρ are known, (A_ρ, C_ρ) is observable, (A_ρ, B_ρ) is controllable, $\text{Im}[C_\rho] = \mathbb{R}^m$, and $\text{Ker}[B_\rho] = \{\mathbf{0}\}$. Depending on the *operation mode* of the system, output data is generated by one of the N subsystems in (1), i.e., the output of the system at time k, $y(k) \in \mathbb{R}^m$, is given by $y(k) = y_\rho(k)$ if the ρ-th mode (system Σ_ρ) is active, $\rho \in \mathcal{N}$. Although $y(k)$ might switch among different modes, we assume (for trajectory classification to actually make sense) that during a window of observations, $k \in \mathcal{K} := \{1, 2, \ldots, K\}$, $K \in \mathbb{N}$, the trajectory $Y^K = \text{col}[y(1), y(2), \ldots, y(K)] \in \mathbb{R}^{Km}$ is generated by a single mode, i.e., $Y^K = \text{col}[y_\rho(1), y_\rho(2), \ldots, y_\rho(K)]$, for some $\rho \in \mathcal{N}$, in response to some driving sequence $U^{K-1} = \text{col}[u(1), u(2), \ldots, u(K-1)] \in \mathbb{R}^{(K-1)l}$. With slight abuse of notation, we often write Y^K as Y_ρ^K to remark that the trajectory has been generated by subsystem Σ_ρ.

Each operation mode $\rho \in \mathcal{N}$ characterizes a *behaviour* of the system. For instance, in smart devices, we may have modes indicating our activity level. Depending whether we are walking, running, or idle, sensors embedded in the device provide different output trajectories Y_ρ^K. Each trajectory would be *consistent* with the dynamical system Σ_ρ that produced it. That is, we have a dynamical system Σ_ρ explaining the data for walking, one for running, and one for idle. Thus, when we say that an input-output trajectory (U^{K-1}, Y^K) is being classified into a mode $\rho \in \mathcal{N}$, we refer to identifying which system Σ_ρ in (1) produced it. To classify the mode, we characterize the set of all input-output trajectories that Σ_ρ could produce, over all possible initial conditions $x_\rho(1) \in \mathbb{R}^\rho$, and then we test if (U^{K-1}, Y^K) belongs to this set – if so, we say that (U^{K-1}, Y^K) is classified into mode ρ. We refer to this set of input-output trajectories as the *behaviour* of mode ρ.

Definition 1 (Behaviour) *The behaviour $\mathcal{B}_\rho \subseteq \mathbb{R}^{Km}$ of system Σ_ρ, over $k \in \mathcal{K} = \{1, \ldots, K\}$, is the set of all input-output trajectories (U^{K-1}, Y^K) satisfying* (1a) *over all possible initial conditions $x_\rho(1) \in \mathbb{R}^{n_\rho}$.*

Hence, classification could be accomplished by identifying to which behaviour $\mathcal{B}_\rho$, $\rho \in \mathcal{N}$, the trajectory (U^{K-1}, Y^K) belongs. Note, however, that if $\mathcal{B}_\rho \cap \mathcal{B}_{\rho'} \neq \emptyset$, for some $\rho, \rho' \in \mathcal{N}$, $\rho \neq \rho'$, trajectories might belong to multiple behaviours, i.e., trajectories in the intersection are not classifiable. This limitation is inherent to the system dynamics and cannot be surpassed by any classifier. In this manuscript, we are interested in forcing the cloud to misclassify trajectories. To induce this, we modify the input-output data that we provide to the cloud so that it appears to have been generated by a different mode. Concretely, given (U^{K-1}, Y^K) generated by some mode $\rho \in \mathcal{N}$, and a behaviour $\mathcal{B}_{\rho'}$, $\rho' \in \mathcal{N}$, $\rho \neq \rho'$, we seek a map, $g_{\rho,\rho'} : \mathcal{B}_\rho \to \mathcal{B}_{\rho'}$, referred here as a *distorting map*. That is, the map $(U^{K-1}, Y^K) \mapsto g_{\rho,\rho'}(U^{K-1}, Y^K)$ takes trajectories from $\mathcal{B}_\rho$ and maps them into $\mathcal{B}_{\rho'}$. By passing (U^{K-1}, Y^K) through $g_{\rho,\rho'}(\cdot)$ before transmission, we are forcing the cloud to classify $g_{\rho,\rho'}(U^{K-1}, Y^K)$ and since $g_{\rho,\rho'}(U^{K-1}, Y^K) \in \mathcal{B}_{\rho'}$, the cloud would (ideally) classify the mode as ρ'.

Definition 2 (Distorting Map and Target Mode) *Given two behaviours, $\mathcal{B}_\rho$ and $\mathcal{B}_{\rho'}$, $\rho, \rho' \in \mathcal{N}$, $\rho \neq \rho'$, we say that a function $g_{\rho,\rho'}(\cdot)$ is a distorting map if $g_{\rho,\rho'} : \mathcal{B}_\rho \to \mathcal{B}_{\rho'}$. We refer to ρ' as the target mode.*

Note that, because the dynamics of the modes in (1) is linear, each behaviour $\mathcal{B}_\rho$ is an linear subspace. The latter implies that there might exist distorting maps that make the Euclidian distance between (U^{K-1}, Y^K) and $g_{\rho,\rho'}(U^{K-1}, Y^K)$ arbitrarily large. We do not want to overly distort trajectories. Usually, there is some sensitive information (function of the input-output trajectory (U^{K-1}, Y^K)) that we would like the cloud to compute accurately. For instance, in intelligent transportation systems, we might want the cloud to accurately compute the average speed of vehicles so that it can send the highway capacity or the shortest route to a destination back to us. To this end, we introduce the notions of *utility* and *utility function* of the trajectory (U^{K-1}, Y^K).

Definition 3 (Utility and Utility Function) *The utility of a trajectory (U^{K-1}, Y^K) refers to some sensitive information, denoted as $z(U^{K-1}, Y^K) \in \mathbb{R}^q$, $q \in \mathbb{N}$, $z : \mathbb{R}^{(K-1)l} \times \mathbb{R}^{Km} \to \mathbb{R}^q$, the cloud must compute accurately. We refer to the function $z(\cdot)$ as the utility function.*

Then, to maintain the utility of the trajectory after distortion, we require that the utility function evaluated at the distorted data equals the utility of the trajectory (U^{K-1}, Y^K). This imposes a constraint on the modes that we can select as target systems and the class of distorting functions that we can use. Concretely, we seek distorting mechanisms, $g_{\rho,\rho'}(\cdot)$, that satisfy $z \circ g_{\rho,\rho'}(U^{K-1}, Y^K) = z(U^{K-1}, Y^K)$ and map the input-output trajectory (U^{K-1}, Y^K_ρ) into the behaviour $\mathcal{B}_{\rho'}$ – leading to incorrect classification.

In some applications, it might not be realistic to assume that we know the utility function exactly. If $z(\cdot)$ is completely unknown, we would not know how to select $g_{\rho,\rho'}(\cdot)$ to avoid overly distorting the trajectory. So, in the problem formulation introduced above, we are implicitly assuming that $z(\cdot)$ is known. To relax this, we work with utility functions that can be written as the composition of two other functions, an *unknown* function $h : \mathbb{R}^r \to \mathbb{R}^q$ and a *known* function $f : \mathbb{R}^{(K-1)l} \times \mathbb{R}^{Km} \to \mathbb{R}^r$, $r \in \mathbb{N}$, i.e., $z(\cdot) = h \circ f(\cdot)$. We formulate the problem in terms of the known part of $z(\cdot)$, the function $f(\cdot)$. This is without loss of generality as if the complete $z(\cdot)$ is known, $z(\cdot) = f(\cdot)$ and $h(\cdot) = \text{id}(\cdot)$, where $\text{id}(\cdot)$ denotes the identity map. From a different perspective, some utility functions, even if they are fully known, might be too complicated to work with. Then, *factorising* $z(\cdot)$ as $h \circ f(\cdot)$ and working with a lower complexity function $f(\cdot)$ might make the problem more tractable. Then, if $z(\cdot) = h \circ f(\cdot)$, for some lower (or equal) complexity known function $f(\cdot)$, the aforementioned utility constraint, $z \circ g_{\rho,\rho'}(U^{K-1}, Y^K) = z(U^{K-1}, Y^K)$, takes the form $h \circ f \circ g_{\rho,\rho'}(U^{K-1}, Y^K) = h \circ f(U^{K-1}, Y^K)$, which is satisfied if (and only if when $h(\cdot)$ is an injection) $f \circ g_{\rho,\rho'}(U^{K-1}, Y^K) = f(U^{K-1}, Y^K)$.

Next, we formally pose the problem we seek to address.

Problem 1 (Misclassification-Utility Problem) *Given an input-output trajectory (U^{K-1}, Y^K_ρ), a target mode ρ', $\rho, \rho' \in \mathcal{N}$, $\rho \neq \rho'$, and a utility function $z(\cdot) = h \circ f(\cdot)$, find a distorting map $g_{\rho,\rho'} : \mathcal{B}_\rho \to \mathcal{B}_{\rho'}$ satisfying: $f \circ g_{\rho,\rho'}(U^{K-1}, Y^K_\rho) = f(U^{K-1}, Y^K_\rho)$.*

Thus, Problem 1 seeks distorting mechanisms for which the distorted data leads to the same utility as (U^{K-1}, Y^K_ρ) – since $f \circ g_{\rho,\rho'}(U^{K-1}, Y^K_\rho) = f(U^{K-1}, Y^K_\rho)$ implies $h \circ f \circ g_{\rho,\rho'}(U^{K-1}, Y^K_\rho) = h \circ f(U^{K-1}, Y^K_\rho)$ – and that map the trajectory (U^{K-1}, Y^K_ρ) into the behaviour $\mathcal{B}_{\rho'}$.

In many applications, the known part of $z(\cdot)$, $f(\cdot)$, is either an average of some sensor measurements or a weighted sum of them over a period of time, i.e., $f(\cdot)$ *is a linear transformation of the output trajectory* Y^K. Motivated by this, we consider affine functions for $f(\cdot)$ that only depend on output trajectories Y^K. Also, because behaviours are linear subspaces and we consider an affine function $f(\cdot)$, we consider affine distorting maps $g_{\rho,\rho'}(\cdot)$ too.

III. Affine Distorting Maps and Utility Functions

Consider $g_{\rho,\rho'}(\cdot)$ and $f(\cdot)$ of the form:

$$g_{\rho,\rho'}(Y, U) := \begin{pmatrix} Y + \Delta_Y \\ U + \Delta_U \end{pmatrix}, \tag{2a}$$

$$f(Y) := FY + \mu, \tag{2b}$$

with $Y, \Delta_Y \in \mathbb{R}^{Km}$, $U, \Delta_U \in \mathbb{R}^{(K-1)l}$, $\Delta := \text{col}[\Delta_Y, \Delta_U]$, $F \in \mathbb{R}^{q \times Km}$, and $\mu \in \mathbb{R}^q$. It follows that Problem 1 amounts to finding $\Delta \in \mathbb{R}^{K(m+l)-l}$ such that $\text{col}[U^{K-1} + \Delta_U, Y^K_\rho + \Delta_Y] \in \mathcal{B}_{\rho'}$, $\rho, \rho' \in \mathcal{N}$, $\rho \neq \rho'$, for some target mode ρ', and $F(Y^K_\rho + \Delta_Y) + \mu = FY^K_\rho + \mu$ (i.e., $\Delta_Y \in \text{Ker}[F]$).

In most real-time applications, input-output data is sent to the cloud immediately after it is generated. Thus, a realistic configuration is to modify $(u(k), y(k))$ recursively and in real-time so that the modified data, say $(\bar{u}(k), \bar{y}(k))$, $k \in \mathcal{K}$, satisfies $F\bar{Y}^K = FY^K$ (same utility) and $(\bar{U}^{K-1}, \bar{Y}^K) \in \mathcal{B}_{\rho'}$ (belongs to the target mode behaviour), for some target mode $\rho' \in \mathcal{N}$, $\bar{Y}^K = \text{col}[\bar{y}(1), \bar{y}(2), \ldots, \bar{y}(K)]$, and $\bar{U}^{K-1} = \text{col}[\bar{u}(1), \bar{u}(2), \ldots, \bar{u}(K-1)]$.

The target mode dynamics, $\Sigma_{\rho'}$, is characterized by the triple $(A_{\rho'}, B_{\rho'}, C_{\rho'})$, $\rho' \in \mathcal{N}$, as introduced in (1). By Definition 1, any input-output trajectory $(\bar{U}^{K-1}, \bar{Y}^K)$ in $\mathcal{B}_{\rho'}$ satisfies the difference equations:

$$\begin{cases} \bar{x}(k+1) = A_{\rho'}\bar{x}(k) + B_{\rho'}\bar{u}(k), \\ \bar{y}(k) = C_{\rho'}\bar{x}(k), \end{cases} \tag{3}$$

for some initial condition $\bar{x}(1) \in \mathbb{R}^{n_{\rho'}}$. Therefore, we can generate trajectories from $\mathcal{B}_{\rho'}$ by fixing the input sequence $\bar{u}(k) \in \mathbb{R}^l$, $k \in \{1, 2, \ldots, K-1\}$, and passing it through (3) for some initial condition, to obtain an output sequence $\bar{y}(k) \in \mathbb{R}^m$, $k \in \mathcal{K}$. By construction, the corresponding trajectory $(\bar{U}^{K-1}, \bar{Y}^K)$ belongs to $\mathcal{B}_{\rho'}$. Thus, we can use (3) to recursively generate trajectories from the target mode behaviour. The idea is that if we send these trajectories through the network (instead of the actual (U^{K-1}, Y^K)), the cloud would classify the mode as ρ'. However, we cannot just send any trajectory. We need Y^K and $\bar{Y}^K$ to lead to the same utility, i.e., $F\bar{Y}^K = FY^K$ (see (2)). Note that if $\bar{Y}^K = Y^K + \Delta_Y$, $F\bar{Y}^K = FY^K$, if and only if $\Delta_Y \in \text{Ker}[F]$. Let $\Delta_Y = \text{col}[\delta_Y(1), \ldots, \delta_Y(K)]$, $\delta_Y(i) \in \mathbb{R}^m$, $i \in \mathcal{K}$; then, $\bar{Y}^K = Y^K + \Delta_Y$ can be written as $\bar{y}(k) = y(k) + \delta_Y(k)$, $k \in \mathcal{K}$. Hence, we can address Problem 1 recursively and in real-time by designing an artificial input sequence $\bar{u}(k)$, $k \in \{1, 2, \ldots, K-1\}$, and an initial condition $\bar{x}(1) \in \mathbb{R}^{n_{\rho'}}$ such that $\bar{y}(k)$ in (3) satisfies $\bar{y}(k) = y(k) + \delta_Y(k)$ for some $\Delta_Y \in \text{Ker}[F]$.

Let $\bar{u}(k) = \bar{u}^1(k) + \bar{u}^2(k)$, and write the state, $\bar{x}(k)$, and output, $\bar{y}(k)$, of (3) as $\bar{x}(k) = \bar{x}^1(k) + \bar{x}^2(k)$ and $\bar{y}(k) = \bar{y}^1(k) + \bar{y}^2(k)$, where $(\bar{x}^1(k), \bar{y}^1(k))$ denotes the part of $(\bar{x}(k), \bar{y}(k))$ driven by $\bar{u}^1(k)$ and $(\bar{x}^2(k), \bar{y}^2(k))$ the part driven by $\bar{u}^2(k)$. Using this new notation and superposition of linear systems, we can write (3) as

$$\begin{cases} \bar{x}^1(k+1) = A_{\rho'}\bar{x}^1(k) + B_{\rho'}\bar{u}^1(k), \\ \quad \bar{y}^1(k) = C_{\rho'}\bar{x}^1(k), \end{cases} \tag{4a}$$

$$\begin{cases} \bar{x}^2(k+1) = A_{\rho'}\bar{x}^2(k) + B_{\rho'}\bar{u}^2(k), \\ \quad \bar{y}^2(k) = C_{\rho'}\bar{x}^2(k), \end{cases} \tag{4b}$$

$$\bar{y}(k) = \bar{y}^1(k) + \bar{y}^2(k), \tag{4c}$$

with corresponding initial conditions $\bar{x}^1(1), \bar{x}^2(1) \in \mathbb{R}^n$ satisfying $\bar{x}(1) = \bar{x}^1(1) + \bar{x}^2(1)$. Then, an approach to enforce $\bar{y}(k) = y(k) + \delta_Y(k)$, $k \in \mathcal{K}$, is to design $\bar{u}^1(k)$ in (4a) such that $\bar{y}^1(k) = C_{\rho'}\bar{x}^1(k) = y(k)$ (*output regulation*), $k \in \mathcal{K}$, and $u^2(k)$ in (4b) to enforce $\bar{y}^2(k) = C_{\rho'}\bar{x}^2(k) = \delta_Y(k)$ (*utility invariance*), $k \in \mathcal{K}$, and apply the combined $\bar{u}(k) = \bar{u}^1(k) + \bar{u}^2(k)$ to the virtual target system (3). By construction, the resulting $\bar{y}(k)$ satisfies $\bar{y}(k) = y(k) + \delta_Y(k)$ and the input-output trajectory $(\bar{U}^{K-1}, \bar{Y}^K)$ belongs to $\mathcal{B}_{\rho'}$.

Using output-regulation techniques [31]-[32], we design input $\bar{u}^1(k)$ and the initial condition $\bar{x}^1(1)$ to regulate the error, $r(k) := \bar{y}^1(k) - y(k)$, given $(u(k), y(k))$, the true mode dynamics Σ_ρ, and the target mode ρ'. Input $u^2(k)$ is used to steer $\bar{y}^2(k)$, $k \in \mathcal{K}$, to an element, Δ_Y, in the kernel of F. Since we know F a priori (before starting the system operation), we can design $u^2(k)$ off-line, i.e., without using real-time data $(u(k), y(k))$. In particular, we lift the target system dynamics (4b) over $k \in \mathcal{K}$ and cast the problem of finding $\bar{x}^2(1)$ and $\bar{u}^2(k)$, $k \in \{1, \ldots, K-1\}$, in terms of the solution of some linear equations.

A. Virtual Output Regulation

Consider the trajectory (U^{K-1}, Y^K) generated in real-time by system Σ_ρ, $\rho \in \mathbb{N}$. At every $k \in \mathcal{K}$, the input-output data available to design $\bar{u}^1(k)$ is (U^{k-1}, Y^k). For ease of presentation, we assume that the state $x_\rho(k)$ of system Σ_ρ is available for feedback. However, when this is not true, given observability of (A_ρ, C_ρ), we can recover the state $x_\rho(k)$ from (U^{k-1}, Y^k) after n time-steps (note that, in general, $n \ll K$). In this case, we would have to wait for n time-steps before we start sending the corrupted data to the cloud. An alternative would be to use *internal model principle* techniques to synthesize *dynamic regulators* [32] for $\bar{u}^1(k)$. In this manuscript, however, we assume that $x_\rho(k)$ is available at every $k \in \mathcal{K}$ and work with *static regulators* to enforce $\bar{y}^1(k) = y(k)$, $k \in \mathcal{K}$.

Consider the following state controller for system (4a):

$$\bar{u}^1(k) = R\bar{x}^1(k) + Lx_\rho(k) + Su(k), \tag{5}$$

with true system state $x_\rho(k) \in \mathbb{R}^{n_\rho}$ and input $u(k) \in \mathbb{R}^l$ of (1), virtual state $\bar{x}^1(k) \in \mathbb{R}^{n_{\rho'}}$ of (4a), and matrices $R \in \mathbb{R}^{l \times n_{\rho'}}$, $L \in \mathbb{R}^{l \times n_\rho}$, and $S \in \mathbb{R}^{l \times l}$. The feedback term, $R\bar{x}^1(k)$, is used to enforce internal stability of (4a)-(5) only, i.e., matrix R is selected so that $(A_{\rho'} + B_{\rho'}R)$ is Schur stable. Such an R always exists due to controllability of $(A_{\rho'}, B_{\rho'})$. We need internal stability to prevent $\bar{u}^1(k)$ from growing unbounded. The remaining terms in (5), $Lx_\rho(k)$ and $Su(k)$, are used to enforce $r(k) = \bar{y}^1(k) - y(k) = \mathbf{0}$, for all $u(k) \in \mathbb{R}^l$, $x_\rho(k) \in \mathbb{R}^{n_\rho}$, and $k \in \mathcal{K}$.

Problem 2 (Virtual Output Regulation) *Given input-output real-time data $(u(k), y(k))$ generated by mode Σ_ρ, the target mode dynamics* (4a)*, controller* (5)*, and a matrix R so that $(A_{\rho'} + B_{\rho'}R)$ is Schur, find (if possible) matrices (L, S) in* (5) *and initial condition $\bar{x}^1(1)$ of the virtual system* (4a) *such that $\bar{y}^1(k) = y(k)$ for all $u(k) \in \mathbb{R}^l$, $x_\rho(k) \in \mathbb{R}^{n_\rho}$, and $k \in \mathcal{K}$.*

Theorem 1 *Problem 2 is solvable if and only if there exist matrices $\Pi \in \mathbb{R}^{n_{\rho'} \times n_\rho}$, $\Gamma \in \mathbb{R}^{l \times n_\rho}$, and $\Theta \in \mathbb{R}^{l \times l}$ that are a solution to the regulator equations:*

$$\begin{cases} A_{\rho'}\Pi - \Pi A_\rho + B_{\rho'}\Gamma = \mathbf{0}, \\ C_{\rho'}\Pi - C_\rho = \mathbf{0}, \\ B_{\rho'}\Theta - \Pi B_\rho = \mathbf{0}. \end{cases} \tag{6}$$

The proof of Theorem 1 is omitted here due to limited space.

Corollary 1 *Consider Problem 2 and let $\Pi \in \mathbb{R}^{n_{\rho'} \times n_\rho}$, $\Gamma \in \mathbb{R}^{l \times n_\rho}$, and $\Theta \in \mathbb{R}^{l \times l}$ solve the regulator equations* (6)*. Then, $L = \Gamma - R\Pi$, $S = \Theta$, and $\bar{x}^1(1) = \Pi x_\rho(1)$ are a solution to Problem 2.*

Theorem 1 provides necessary and sufficient conditions for Problem 2 to have a solution in terms of the solution of the regulator equations (6). Once we have a solution (this solution might not be unique), for given R so that $(A_{\rho'} + B_{\rho'}R)$ is Schur, we can compute matrices (L, S) and initial condition $\bar{x}^1(1)$ to realize the controller $\bar{u}^1(k)$ in (5) using Corollary 1.

Remark 1 *Note that any controller $\bar{u}^1(k)$ in* (5) *and initial condition $\bar{x}^1(1)$ of* (4a) *solving Problem 2 provide already a solution to Problem 1 (for the class of $f(\cdot)$ and $g_{\rho,\rho'}(\cdot)$ introduced above). That is, these $\bar{u}^1(k)$ and $\bar{y}^1(k)$, $k \in \mathcal{K}$, belong to $\mathcal{B}_{\rho'}$ and, because $\bar{y}^1(k) = y(k)$, Y^K and $\bar{Y}^K = \text{col}[\bar{y}^1(1), \ldots, \bar{y}^1(k)]$ have the same utility, i.e., $FY^K = F\bar{Y}^K$. However, if we share $\bar{u}^1(k)$ and $\bar{y}^1(k)$, the cloud would still get the true output data – with a different input sequence though. In the next subsection, we provide tools for properly distorting output data to avoid sharing the true output sequence with the cloud.*

B. Utility Invariance (Batch Approach)

In this subsection, we provide tools for designing $\bar{u}^2(k)$ and $\bar{x}^2(1)$ in (4b) so that $\bar{y}^2(k)$, $k \in \mathcal{K}$, is steered to an element, $\Delta_Y \neq \mathbf{0}$, in the kernel of F. We use the resulting $\bar{u}^2(k)$ and a $\bar{u}^1(k)$ synthesized using Corollary 1 to construct the actual input $\bar{u}(k) = \bar{u}^1(k) + \bar{u}^2(k)$ and initial condition $\bar{x}(1) = \bar{x}^1(1) + \bar{x}^2(1)$ for the virtual system (3). Because we know F a priori (before starting the system operation) and $\bar{u}^1(k)$ is already being used to handle real-time data, we can actually design $u^2(k)$ off-line, i.e., independent of $(u(k), y(k))$. To do so, we lift the target system dynamics (4b), over $k \in \mathcal{K}$, and cast the problem of finding $\bar{x}^2(1)$ and $\bar{u}^2(k)$, $k \in \{1, \ldots, K-1\}$, in terms of the solution of some linear equations.

We aim at enforcing that the sequence of virtual outputs $\bar{y}^2(k)$, $k \in \mathcal{K}$, is contained in the kernel of F. If Ker$[F]$ is trivial, the only vector to which we can drive $\bar{y}^2(k)$ is the zero vector. In this case, $\bar{u}^1(k)$ solving Problem 2 and $\bar{y}^1(k) = y(k)$ are the only option for solving Problem 1 (see Remark 1). Therefore, a necessary condition for Problem 1 to have a different solution ($\bar{y}(k) \neq y(k)$) is that F has a nontrivial kernel.

Assumption 1 *The kernel of $F \in \mathbb{R}^{q \times Km}$ is nontrivial.*

Consider $\tilde{Y}^K := \text{col}[\bar{y}^2(1), \ldots, \bar{y}^2(K)]$. The stacked vector $\tilde{Y}^K$ can be written explicitly in terms of $\bar{x}^2(1)$ and $\tilde{U}^{K-1} := \text{col}[\bar{u}^2(1), \ldots, \bar{u}^2(K-1)]$ as follows

$$\begin{cases} \tilde{Y}^K = \mathcal{O}_K \bar{x}^2(1) + \mathcal{T}_K \tilde{U}^{K-1}, \\ \mathcal{T}_K := \begin{bmatrix} \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ C_{\rho'} B_{\rho'} & \mathbf{0} & \cdots & \mathbf{0} \\ C_{\rho'} A_{\rho'} B_{\rho'} & C_{\rho'} B_{\rho'} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ C_{\rho'} (A_{\rho'})^{K-2} B_{\rho'} & C_{\rho'} (A_{\rho'})^{K-3} B_{\rho'} & \cdots & C_{\rho'} B_{\rho'} \end{bmatrix}, \\ \mathcal{O}_K := \begin{bmatrix} C_{\rho'} \\ C_{\rho'} A_{\rho'} \\ \vdots \\ C_{\rho'} (A_{\rho'})^{K-1} \end{bmatrix}, \end{cases} \tag{7}$$

Problem 3 (Utility Invariance) *Given the utility function* (2b) *and the target mode dynamics* (4b)*, find (if possible) an initial condition $\bar{x}^2(1) \in \mathbb{R}^{n_{\rho'}}$ of* (4b) *and a sequence of inputs $\tilde{U}^{K-1} = \text{col}[\bar{u}^2(1), \ldots, \bar{u}^2(K-1)]$ such that $\mathcal{O}_K \bar{x}^2(1) + \mathcal{T}_K \tilde{U}^{K-1} \in \text{Ker}[F]$, with $\mathcal{O}_K$ and $\mathcal{T}_K$ as defined in* (7).

Lemma 1 *Consider the target mode behaviour $\mathcal{B}_{\rho'}$. Problem 3 is solvable if and only if $\mathcal{B}_{\rho'} \cap \text{Ker}[F] \neq \emptyset$.*

The proof of Lemma 1 is omitted here due to limited space.

Theorem 2 *Problem 3 is solvable if and only if there exist vectors $x \in \mathbb{R}^{n_{\rho'}}$, $U \in \mathbb{R}^{(K-1)l}$, and $\theta \in \mathbb{R}^{Km}$ solution to the linear equations:*

$$\begin{pmatrix} \mathcal{O}_K & \mathcal{T}_K & (F^+ F - I_{Km}) \end{pmatrix} \begin{pmatrix} x \\ U \\ \theta \end{pmatrix} = \mathbf{0}. \tag{8}$$

The proof of Theorem 2 is omitted here due to limited space.

Corollary 2 *Consider Problem 3 and let $x \in \mathbb{R}^{n_{\rho'}}$, $U \in \mathbb{R}^{(K-1)l}$, and $\theta \in \mathbb{R}^{Km}$ solve* (8)*. Then, $\bar{x}^2(1) = x$ and $\tilde{U}^{K-1} = U$ are a solution to Problem 3.*

Theorem 2 provides necessary and sufficient conditions for Problem 3 to have a solution in terms of the solution of (8). If there exists a solution, using Corollary 1, we can compute the initial condition $\bar{x}^2(1)$ and the sequence of controllers $\tilde{U}^{K-1} = \text{col}[\bar{u}^2(1), \ldots, \bar{u}^2(K-1)]$ to drive system (4b) so that $\tilde{Y}^K \in \text{Ker}[F]$.

Remark 2 *For given mode $\rho' \in \mathcal{N}$ and utility matrix F, there either do not exist solutions to* (8) *or the solution is unique or there exist an infinite number of solutions. In the latter case, the set of solutions form a linear subspace, which implies that $\tilde{Y}^K$ can be chosen arbitrarily large. This is the most appealing case to us as we can induce arbitrarily large distortion without affecting the utility of the trajectory.*

In the next subsection, we provide a synthesis procedure to summarize the results presented above.

C. Synthesis Procedure:

Synthesis:

1) Given the mode dynamics Σ_ρ in (1), $\rho \in \mathcal{N}$, that will generate the input-output trajectory, select a target mode $\Sigma_{\rho'}$, $\rho' \in \mathcal{N}$, $\rho \neq \rho'$.

2) Using the true system matrices (A_ρ, B_ρ, C_ρ) and the target mode matrices $(A_{\rho'}, B_{\rho'}, C_{\rho'})$, seek a solution $\Pi \in \mathbb{R}^{n_{\rho'} \times n_\rho}$, $\Gamma \in \mathbb{R}^{l \times n_\rho}$, and $\Theta \in \mathbb{R}^{l \times l}$ to the regulator equations (6).

3) Select any matrix R so that $(A_{\rho'} + B_{\rho'} R)$ is Schur, and compute matrices (L, S) of controller $\bar{u}^1(k)$ in (5) and the initial condition $\bar{x}^1(1)$ of (4a) using Corollary 1.

4) Consider the virtual target system (4a), with initial $\bar{x}^1(1)$, and close it with controller $\bar{u}^1(k)$ in (5).

5) Given the trajectory length $K \in \mathbb{N}$ and the utility matrix F in (2b), compute matrices $\mathcal{O}_K$ and $\mathcal{T}_K$ in (7) and seek a solution $x \in \mathbb{R}^{n_{\rho'}}$, $U \in \mathbb{R}^{(K-1)l}$, and $\theta \in \mathbb{R}^{Km}$ to the linear equations (8).

6) Compute the initial condition $\bar{x}^2(1)$ of (4b) and the sequence of controllers $\{\bar{u}^2(1), \ldots, \bar{u}^2(K-1)\}$ using Corollary 2.

7) Consider the virtual target system (4b), with initial $\bar{x}^2(1)$, and close it with controller $\bar{u}^2(k)$, $k \in \mathcal{K}$.

8) Compute the combined input $\bar{u}(k) = \bar{u}^1(k) + \bar{u}^2(k)$ and corresponding output $\bar{y}(k) = \bar{y}^1(k) + \bar{y}^2(k)$ in (4c), and send $(\bar{u}(k), \bar{y}(k))$ to the cloud in real-time.

IV. Conclusion

We have proposed a new formulation for dealing with privacy problems in cyber-physical systems. In particular, for a class of CPSs, we have addressed the problem of performing computations over the cloud without revealing private information about the structure and operation of the system. A distorting mechanism (based on output regulation techniques) that ensure CPSs data privacy and utility invariance has been proposed.

REFERENCES

[1] R. H. Weber, "Internet of things - new security and privacy challenges," *Computer Law and Security Review*, vol. 26, pp. 23–30, 2010.

[2] S. R. Rajagopalan, L. Sankar, S. Mohajer, and H. V. Poor, "Smart meter privacy: A utility-privacy framework," in *2011 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, 2011, pp. 190–195.

[3] O. Tan, D. Gunduz, and H. V. Poor, "Increasing smart meter privacy through energy harvesting and storage devices," *IEEE Journal on Selected Areas in Communications*, vol. 31, pp. 1331–1341, 2013.

[4] F. Farokhi and H. Sandberg, "Optimal privacy-preserving policy using constrained additive noise to minimize the fisher information," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 2017.

[5] N. Hashemi, E. V. German, J. P. Ramirez, and J. Ruths, "Filtering approaches for dealing with noise in anomaly detection," in *arXiv:1909.01477*, 2019.

[6] N. Hashemi and J. Ruths, "Generalized chi-squared detector for lti systems with non-gaussian noise," in *2019 American Control Conference (ACC)*, 2019.

[7] C. M. Ahmed, C. Murguia, and J. Ruths, "Model-based attack detection scheme for smart water distribution networks," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ser. ASIA CCS '17, 2017, pp. 101–113.

[8] F. Farokhi, H. Sandberg, I. Shames, and M. Cantoni, "Quadratic Gaussian privacy games," in *2015 54th IEEE Conference on Decision and Control (CDC)*, 2015, pp. 4505–4510.

[9] F. Farokhi and G. Nair, "Privacy-constrained communication," *IFAC-PapersOnLine*, vol. 49, pp. 43 – 48, 2016.

[10] C. Murguia and J. Ruths, "On reachable sets of hidden cps sensor attacks," in *proceedings of the American Control Conference* (ACC), 2018.

[11] F. Pasqualetti, F. Dorfler, and F. Bullo, "Attack detection and identification in cyber-physical systems," *IEEE Transactions on Automatic Control*, vol. 58, pp. 2715–2729, 2013.

[12] C. Murguia and J. Ruths, "Cusum and chi-squared attack detection of compromised sensors," in *proceedings of the IEEE Multi-Conference on Systems and Control (MSC)*, 2016.

[13] L. H. Ozarow and A. D. Wyner, "Wire-tap channel ii," in *Advances in Cryptology*, T. Beth, N. Cot, and I. Ingemarsson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985, pp. 33–50.

[14] C. Murguia and J. Ruths, "Characterization of a cusum model-based sensor attack detector," in *proceedings of the 55th IEEE Conference on Decision and Control (CDC)*, 2016.

[15] F. du Pin Calmon and N. Fawaz, "Privacy against statistical inference," in *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2012, pp. 1401–1408.

[16] N. Hashemil, C. Murguia, and J. Ruths, "A comparison of stealthy sensor attacks on control systems," in *proceedings of the American Control Conference (ACC), 2018*, 2018.

[17] C. Murguia, N. van de Wouw, and J. Ruths, "Reachable sets of hidden cps sensor attacks: Analysis and synthesis tools," in *proceedings of the IFAC World Congress*, 2016.

[18] S. H. Kafash, J. Giraldo, C. Murguia, A. A. Cardenas, and J. Ruths, "Constraining attacker capabilities through actuator saturation," in *proceedings of the American Control Conference (ACC), 2018*, 2018.

[19] C. Murguia, I. Shames, F. Farokhi, and D. Nešić, "On privacy of quantized sensor measurements through additive noise," in *proceedings of the 57th IEEE Conference on Decision and Control (CDC)*, 2018.

[20] T. Yang, C. Murguia, M. Kuijper, and D. Nešić, "A robust circle-criterion observer-based estimator for discrete-time nonlinear systems in the presence of sensor attacks," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018.

[21] J. L. Ny and G. J. Pappas, "Differentially private filtering," *IEEE Transactions on Automatic Control*, vol. 59, pp. 341–354, 2014.

[22] C. Murguia, S. I., F. F., N. D., and P. V., "On privacy of dynamical systems: An optimal probabilistic mapping approach," *arxiv:1910.13559*, 2019.

[23] F. Farokhi and H. Sandberg, "Ensuring privacy with constrained additive noise by minimizing fisher information," *Automatica*, vol. 99, pp. 275 – 288, 2019.

[24] Y. Wang, Z. Huang, S. Mitra, and G. E. Dullerud, "Differential privacy in linear distributed control systems: Entropy minimizing mechanisms and performance tradeoffs," *IEEE Transactions on Control of Network Systems*, vol. 4, pp. 118–130, 2017.

[25] T. Tanaka, M. Skoglund, H. Sandberg, and K. H. Johansson, "Directed information and privacy loss in cloud-based control," in *2017 American Control Conference (ACC)*, 2017, pp. 1666–1672.

[26] A. Jones, K. Leahy, and M. Hale, "Towards differential privacy for symbolic systems," in *2019 American Control Conference (ACC)*, 2019, pp. 372–377.

[27] C. Murguia, I. Shames, F. Farokhi, and D. Nešić, *Information-Theoretic Privacy Through Chaos Synchronization and Optimal Additive Noise*. Singapore: Springer Singapore, 2020, pp. 103–129.

[28] A. Sultangazin and P. Tabuada, "Symmetries and isomorphisms for privacy in control over the cloud," *arXiv:1906.07460*, 2019.

[29] C. Dwork, "Differential privacy: A survey of results," in *Theory and Applications of Models of Computation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1–19.

[30] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY, USA: Wiley-Interscience, 1991.

[31] A. Francis, "The linear multivariable regulator problem," *SIAM Journal on Control and Optimization*, vol. 15, pp. 486–505, 1977.

[32] A. Francis and W. Wonham, "The internal model principle of control theory," *Automatica*, vol. 12, pp. 457 – 465, 1976.