Towards the use of symmetries to ensure privacy in control over the cloud*

Alimzhan Sultangazin[1] and Paulo Tabuada[1*†]

April 27, 2018

Abstract

With the advent of cloud computing and the increasing connectivity of devices at the edge of the internet, closing feedback control loops over the cloud is becoming a reality. This is especially the case when computationally expensive algorithms, such as model-predictive control for nonlinear plants, are used to optimize performance. A major roadblock to closing feedback control loops over the cloud, however, is ensuring privacy of the exchanged data. Further exacerbating this difficulty, is the need to minimize the computational overhead of enforcing privacy so as not to degrade control performance. In this paper, we propose several methods for enforcing data privacy using symmetry transformations. We address three different scenarios: a) the cloud has no knowledge about the system being controlled; b) the cloud knows what sensors and actuators the system employs but not the system dynamics; c) the cloud knows the system dynamics, its sensors, and actuators. The proposed methods allow us, in all of these three scenarios, to successfully execute control over the cloud without revealing private information (which information is considered private depends on the considered scenario). The advantage of these methods lies in their generality, which makes them applicable to wide class of systems and with low computational overhead.

1 Introduction

1.1 Motivation

In view of the recent proliferation of cloud computing, the idea of leveraging the power of cloud computing for control, that we will refer as control over the cloud, has garnered wide attention in both industry and academia. As opposed to traditional control wherein a controller is designed to be executed locally on the plant, in control over the cloud, expensive control and optimization jobs are offloaded to the cloud.

Control over the cloud has several advantages, which include the possibility of outsourcing expensive computational tasks to the cloud and the availability of global information from all of the cloud's clients when making control decisions. An illustrative example of the benefits of outsourcing computing is Model Predictive Control (MPC) of complex plants. MPC usually involves solving complex optimization problems in real-time. An experimental study corroborating feasibility of MPC over the cloud can be found in [1], where it was

[*] *The work of the authors was partially supported by the NSF grants 1740047, 1705135 and by the Army Research Laboratory under Cooperative Agreement W911NF-17-2-0196. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

[†1] Alimzhan Sultangazin and Paulo Tabuada are with Department of Electrical and Computer Engineering, University of California - Los Angeles, USA {asultangazin, tabuada}@ucla.edu

used in robot control. A study considering cloud-based MPC for a large-scale solar plant and the benefits of outsourcing computational tasks can be found in [2]. The availability of global information in control over the cloud can produce strong benefits in the scenario of traffic monitoring and management, as shown in [3]. In this work, individual vehicles are considered as individual plants with the control task of efficiently arriving at their respective destinations. In view of the fact that all vehicles share a common infrastructure, the anonymously collected data from each vehicle is used by a centralized controller to improve the overall performance of the system. These, and many other, examples illustrate an some of the advantages that control over the cloud is able to provide.

Despite all the benefits of control over the cloud, a growing number of current studies show that exposing existing systems to connectivity may lead to security vulnerabilities in a vast variety of applications [4][5][6][7][8][9]. Recently, a set of recommendations from the National Institute of Standards and Technology (NIST) [10] has drawn attention to the rising number of cases of IoT attacks and called for the need of standardization of CPS/IoT security. A vivid example of the vulnerability of control systems to attacks is an incident from 2010, when a computer worm Stuxnet attacked numerous industrial control systems with a key strategic value to certain nation states [11]. The examples above illustrate an increased concern by the CPS community with the security and privacy of control systems and a growing need to develop effective solutions to address these issues. Nonetheless, the body of work concerned with CPS/IoT security and/or privacy is still in its infancy.

The control-over-the-cloud architecture inherently allows the cloud to receive sensitive data about its clients, who are forced to disclose it in exchange for a control action. This data can be reasonably viewed as a valuable commodity and there exists the possibility that the cloud may use this data to infer private information about the client [12]. Therefore, in this paper, the cloud is taken to be a curious but honest agent, meaning that it tries to utilize the data it receives to its own benefit, while faithfully executing control over the plant.

A naive approach for ensuring privacy in control systems is to use traditional IT security, such as cryptographic primitives and security protocols. However, as argued in [11], this approach is only a partial solution. First, it would still leave systems vulnerable to adversaries, who somehow gained unauthorized access. While, in the realm of pure information security, attacks have no physical consequences, this is no longer the case in CPS/IoT, where a loss of privacy can have severe physical consequences in real world (e.g. failure on the nuclear plant). Second, encryption solutions will most probably introduce significant delays in the feedback loop, which would deteriorate the control system performance. This calls for privacy solutions that take into account the specific nature and needs of control systems.

1.2 Related work

The existing literature on protecting privacy in control over the cloud can be classified into two types of approaches: data encryption and data perturbation. Examples of data encryption technologies used in the context of control systems include full or partial homomorphic encryption [13, 14, 15], data obfuscation [16], and multi-party computation schemes [17]. Usually, the main roadblock in implementing data encryption in real systems usually comes from the large computational overhead of these methods, vulnerability of some schemes during distributing public keys, or a need to disclose some or all the information about the system.

Data perturbation, an idea widely studied in the context of statistical databases, has only very recently been applied in control theory (see [18, 19]). The idea is to provide the cloud with perturbed aggregate information about a group of systems, while ensuring that nobody can infer information about an individual system from it. As opposed to a traditional binary notion of privacy (i.e., either private or not private), in data perturbation, privacy changes continuously from total privacy to absence of privacy.

1.3 Contributions of this paper

While, as previously noted, ensuring privacy in control systems poses several difficulties due to the inherent nature of the problem (e.g. timing constraints, physical impact on the real world), some specific features of control systems can, in fact, be leveraged to solve the problem of privacy. In this paper, an alternative approach to ensuring privacy in control systems is proposed that utilizes the notion of isomorphisms and symmetries of control systems. The benefits of this approach include a significantly reduced computational overhead, possibility of computation on encrypted data (like in homomorphic computation) and simplicity of design.

When performing control-over-the-cloud, the client needs to provide the cloud with the plant model, the control objective (e.g., a cost to be optimized and constraints), as well as sensor measurements. Our objective is to keep this information private, and we consider three separate scenarios depending on the cloud knowledge:

1. the cloud has no knowledge about the plant, sensors, or actuators;
2. the cloud has no knowledge about the plant but knows its sensors and actuators (e.g., the cloud knows that a house is equipped with a smart meter but does not know which appliances it contains);
3. the cloud knows the plant, its sensors, and actuators (e.g., traffic routing applications know the dynamics of cars, their sensors and actuators).

Our results provide different privacy guarantees for each one of these scenarios in terms of an equivalence relation induced by isomorphisms and symmetries of control systems. We leave it as a future work to produce more detailed descriptions of these equivalence classes as this requires a careful analysis of symmetry groups for control systems.

2 Problem Definition

2.1 Plant dynamics and control objective

To simplify the presentation all the results are described in the context of linear systems. Readers familiar with nonlinear control will see that all the results extend to nonlinear systems, *mutatis mutandis.* We consider discrete-time linear plants, denoted by Σ, and described by:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + c \\ y_k &= Cx_k + d, \end{aligned} \tag{2.1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{p\times n}$, $c \in \mathbb{R}^n$, and $d \in \mathbb{R}^p$ describe the dynamics of the system, and $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$ and $y \in \mathbb{R}^p$ denote the state, input and output of the system, respectively.

To simplify notation, we lift every affine map into a linear map through the following construction:

$$\begin{aligned} \tilde{x}_{k+1} \equiv \begin{bmatrix} x_{k+1} \\ 1 \end{bmatrix} &= \begin{bmatrix} A & c \\ 0_{1\times n} & 1 \end{bmatrix} \begin{bmatrix} x_k \\ 1 \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u \equiv \tilde{A}\tilde{x}_k + \tilde{B}u_k \\ y_k &= \begin{bmatrix} C & d \end{bmatrix} \begin{bmatrix} x_k \\ 1 \end{bmatrix} = \tilde{C}\tilde{x}_k. \end{aligned} \tag{2.2}$$

In the remainder of the paper we will suppress the tilde for simplicity, however, the reader should keep in mind that we are dealing with affine maps. We will refer to system (2.2) as the triple $\Sigma = (A, B, C)$.

A triple $\{x_k, u_k, y_k\}_{k\in\mathbb{N}}$ is said to be a trajectory of Σ if it satisfies (2.1) for all $k \in \mathbb{N}$.

In addition, for each plant, we define a cost function $J : \mathbb{R}^n \times (\mathbb{R}^m)^{N+1} \to \mathbb{R}$ for $N \in \mathbb{N}\cup\{+\infty\}$ that allows to compare trajectories, thereby formulating different control objectives. In keeping with the linear framework, we consider quadratic cost functions given by:

$$J(x, u) = \sum_{k=0}^{N} \Delta\eta_k^T M \Delta\eta_k, \tag{2.3}$$

where $\Delta\eta_k = \begin{bmatrix} x_k - x^* \\ u_k - u^* \end{bmatrix}$. The state x^* and input u^* denote the desired setpoint induced by the cost function.

In addition to a cost, we also consider control objectives that require constraints to be satisfied at all times. These constraints are defined by:

$$D\eta_k \leq 0, \tag{2.4}$$

where $\eta_k = \begin{bmatrix} x_k & u_k \end{bmatrix}^T$. Note that, despite appearing to be linear constraints, the constraints above are in fact affine, in view of the construction in (2.2).

2.2 Attack model and privacy objectives

We treat the plant as a curious but honest adversary. This means that the plant will follow the protocol agreed upon, although it may seek to extract and leak confidential information.

The interaction between the plant and the cloud is performed in two steps. The first step is a handshake during which the plant provides the cloud with a suitably modified version of the plant model, cost, and constraints. In return, the cloud agrees to compute the input minimizing the provided cost, subject to the constraints. This is done prior to the plant operation. The second step corresponds to the plant operation during which the plant sends a suitably modified version of its measurements to the cloud. The cloud computes a new input based on the received measurements and sends it to the plant, where it is suitably modified before being applied to the plant.

In the previous paragraph we purposely used the vague expression "suitably modified". Making this expression more concrete requires that we first define the knowledge available to the plant. We consider the following three scenarios.

Problem 2.1 (Scenario 1)**.** *Assuming the cloud has no knowledge about the plant:*

1. *how to modify the plant (A, B, C), cost J, and constraint matrix D in the handshaking phase;*

2. *how to modify the measurements sent to the plant, and*

3. *how to modify the inputs received from the plant,*

so that the plant's trajectory minimizes cost J in (2.3)*, while preventing the cloud from learning the plant (A, B, C), the cost J, the constraint matrix D, and the plant's trajectory $\{x_k, u_k, y_k\}_{k\in\mathbb{N}}$?*

Problem 2.2 (Scenario 2)**.** *Assuming the cloud has no knowledge about the plant except for knowing what are the sensors and actuators:*

1. *how to modify the plant* (A, B, C)*, cost* J*, and constraint matrix* D *in the handshaking phase;*
2. *how to modify the measurements sent to the plant, and*
3. *how to modify the inputs received from the plant,*

so that the plant's trajectory minimizes cost J *in* (2.3)*, while preventing the cloud from learning the plant* (A, B, C)*, the cost* J*, the constraint matrix* D*, or the plant's trajectory* $\{x_k, u_k, y_k\}_{k\in\mathbb{N}}$*?*

Problem 2.3 (Scenario 3)**.** *Assuming the cloud has complete knowledge about the plant dynamics, including its sensors and actuators:*

1. *how to modify cost* J*, and constraint matrix* D *in the handshaking phase;*
2. *how to modify the measurements sent to the plant, and*
3. *how to modify the inputs received from the plant,*

so that the plant's trajectory minimizes cost J *in* (2.3)*, while preventing the cloud from learning the cost* J*, the constraint matrix* D*, or the plant's trajectory* $\{x_k, u_k, y_k\}_{k\in\mathbb{N}}$*?*

These problems are solved in Section 4 by utilizing isomorphisms and symmetries of control systems, defined in Section 3.

3 Isomorphisms and symmetries of control systems

In this section, we introduce the notions of isomorphism and symmetry of control systems along with several technical results used in Section 4 to provide a solution to the problems described in Section 2.

An *isomorphism* of a control system consists of a change of coordinates, an invertible feedback, and a change of coordinates in the space of outputs. In the context of linear systems, these transformations are linear invertible maps.

Definition 3.1. Let $\Sigma = (A, B, C)$ and $\hat{\Sigma} = (\hat{A}, \hat{B}, \hat{C})$ be linear control systems. The quadruple $\psi = (P, F, G, S)$ is an isomorphism from Σ to $\hat{\Sigma}$, denoted by $\psi_*\Sigma = \hat{\Sigma}$, if $P : \mathbb{R}^n \to \mathbb{R}^n$, $G : \mathbb{R}^m \to \mathbb{R}^m$, and $S : \mathbb{R}^p \to \mathbb{R}^p$ are invertible linear maps, $F : \mathbb{R}^n \to \mathbb{R}^m$ is a linear map and:

$$\begin{aligned}\hat{\Sigma} = \psi_*\Sigma &= (P(A - BG^{-1}F)P^{-1}, PBG^{-1}, SCP^{-1}) \\ &\equiv (\hat{A}, \hat{B}, \hat{C}). \qquad (3.1)\end{aligned}$$

The isomorphism takes the state x_k, input u_k and output y_k of system Σ to the state z_k, input v_k, and output w_k of system $\hat{\Sigma}$ as follows:

$$z_k = Px_k \qquad (3.2)$$

$$v_k = Fx_k + Gu_k \qquad (3.3)$$

$$w_k = Sy_k. \qquad (3.4)$$

We now study the effect of isomorphisms on the cost and constraints. This will be used in Section 4 to prevent the cloud from learning the cost and constraints.

The effect of transformations (3.2)-(3.4) on η can be represented by:

$$\Delta\hat{\eta}_k = \begin{bmatrix}\Delta z_k \\ \Delta v_k\end{bmatrix} = \begin{bmatrix}P & 0 \\ F & G\end{bmatrix}\begin{bmatrix}\Delta x_k \\ \Delta u_k\end{bmatrix} \equiv L\Delta\eta_k. \tag{3.5}$$

Therefore, the cost function J can be expressed as a function of new states z and inputs v as follows:

$$\hat{J}(z,v) = \psi_* J\,(x,u) = \sum_{k=0}^{N} \Delta\hat{\eta}_k^T \hat{M} \Delta\hat{\eta}_k, \tag{3.6}$$

where $\hat{M} = L^{-T} M L^{-1}$.

The same transformations can be applied to the constraints in (2.4):

$$\hat{D}\hat{\eta}_k \leq 0, \tag{3.7}$$

where $\hat{D} = \psi_* D = DL^{-1}$.

Observe that the set of isomorphisms of a given system Σ with function composition as a group operation forms a group[1]. This allows us to define an equivalence relation between quadruples.

Definition 3.2. Let $\mathcal{G}$ be a subgroup of the group of all isomorphisms of Σ. Two quadruples $(\Sigma, J, D, \{x_k, u_k, y_k\}_{k\in\mathbb{N}})$ and $(\hat{\Sigma}, \hat{J}, \hat{D}, \{z_k, v_k, w_k\}_{k\in\mathbb{N}})$ are called $\sim_{\mathcal{G}}$-equivalent if there exists an isomorphism $\psi \in \mathcal{G}$ such that $\psi_*\Sigma_1 = \Sigma_2$, $\hat{J} = \psi_* J$, $\hat{D} = \psi_* D$ and (3.2)-(3.4) hold for every $k \in \mathbb{N}$.

Among the isomorphisms of a control system Σ, there is a special subgroup that we call the symmetries of Σ. A general definition of symmetry can be given as follows:

Definition 3.3. Let Σ be a linear control system. An isomorphism ψ of Σ is said to be a symmetry of Σ if $\psi_*\Sigma = \Sigma$.

The notion of isomorphism was crafted to preserve properties of control systems. Among these, trajectories are of special significance.

Lemma 3.4. *If* $\hat{\Sigma} = \psi_*\Sigma$ *and* $\{x_k, u_k, y_k\}_{k\in\mathbb{N}}$ *is a trajectory of* Σ*, then* $\{Px_k, Fx_k + Gu_k, Sy_k\}_{k\in\mathbb{N}}$ *is a trajectory of* $\hat{\Sigma}$.

Proof. Let us take $z_k = Px_k$ and $v_k = Fx_k + Gu_k$ and show that they are valid state and input trajectories of $\hat{\Sigma}$, respectively. This can be shown by induction. For $k = 1$,:

$$\begin{aligned} z_1 &= \hat{A}z_0 + \hat{B}v_0 = \\ &= P\left(A - BG^{-1}F\right)P^{-1}Px_0 + PBG^{-1}(Fx_0 + Gu_0) = \\ &= P[(A - BG^{-1}F)x_0 + BG^{-1}Fx_0 + Bu_0)] = \\ &= P(Ax_0 + Bu_0) = \\ &= Px_1, \end{aligned}$$

[1] A composition of two isomorphisms is given by $\psi_2 \circ \psi_1 = (P_2P_1, G_2F_1 + F_2P_1, G_2G_1, S_2S_1)$ and the inverse is given by $\psi^{-1} = (P^{-1}, -G^{-1}FP, G^{-1}, S^{-1})$.

where we used transformations (3.2), (3.3) and (3.4).

Suppose that for $k = i$ it is true that $z_i = Px_i$. Let us show that this is true for all $k \in \mathbb{N}$. Consider z_{i+1}. It can be again shown using transformations (3.2), (3.3) and (3.4):

$$\begin{aligned}
z_{i+1} &= \hat{A}z_i + \hat{B}v_i = \\
&= P\left(A - BG^{-1}F\right)P^{-1}Px_i + PBG^{-1}(Fx_i + Gu_i) = \\
&= P[(A - BG^{-1}F)x_i + BG^{-1}Fx_i + Bu_i)] = \\
&= P(Ax_i + Bu_i) = \\
&= Px_{i+1}.
\end{aligned}$$

Hence, it is true for all $k \in \mathbb{N}$ that if x_k and u_k are state and input trajectories of Σ, then $z_k = Px_k$ and $v_k = Fx_k + Gu_k$ are valid trajectories of $\hat{\Sigma}$.

Finally, we observe that:

$$w_k = \hat{C}z = SCP^{-1}Px_k = SCx_k = Sy_k.$$

□

The next result shows that when the cloud optimizes $\hat{J}$ and the plant replaces v_k with u_k, the resulting sequence of inputs u_k optimizes J.

Lemma 3.5. *Suppose the cloud solves the optimization problem:*

$$\begin{array}{ll}
\underset{v}{\mathit{min}} & \hat{J}(Px, v) \\
\textit{subject to} & \quad \hat{D}\hat{\eta}_k \leq 0,
\end{array}$$

for the plant $\hat{\Sigma} = \psi_* \Sigma$ and this optimization problem has the unique solution v^o. Then, the unique solution of the optimization problem:

$$\begin{array}{ll}
\underset{u}{\mathit{min}} & J(x, u) \\
\textit{subject to} & \quad D\eta_k \leq 0,
\end{array}$$

for the plant Σ is given by $u^o = G^{-1}(v^o - Fx)$.

Proof. The proof of this statement is self-evident since both J and $\hat{J}$ represent the same function expressed in terms of different variables.

To illustrate this, assume there exists some optimal $u^1 \neq u^o$. That is, $J(x, u^1) \leq J(x, u^0)$. Define $v^1 \equiv Fx + Gu^1$. This implies that:

$$\hat{J}(v^1) = J(u^1) \leq J(u^0) = \hat{J}(v^o).$$

which contradicts the fact that v^o is the unique optimal solution of $\underset{v}{\min}\hat{J}$.

Additionally, feasibility set of the given problems is the same. Since $\hat{D}\hat{\eta}_k = DL^{-1}L\eta_k = D\eta_k$:

$$\hat{D}\hat{\eta}_k \leq 0 \Longleftrightarrow D\eta_k \leq 0.$$

□

The main reason for using isomorphisms is that the cloud is unable to distinguish between isomorphic systems.

Theorem 3.6. *Any two quadruples* $(\Sigma, J, D, \{x_k, u_k, y_k\}_{k\in\mathbb{N}})$ *and* $(\hat{\Sigma}, \hat{J}, \hat{D}, \{z_k, v_k, w_k\}_{k\in\mathbb{N}})$ *of plants, costs, constraints and trajectories, which are* $\sim_{\mathcal{G}}$*-equivalent, are indistinguishable by the cloud.*

Proof. It suffices to show that for any two isomorphic quadruples $(\Sigma, J, D, \{x_k, u_k, y_k\}_{k\in\mathbb{N}})$ and $(\hat{\Sigma}, \hat{J}, \hat{D}, \{z_k, v_k, w_k\}_{k\in\mathbb{N}})$ the cloud receives and sends the same data if the plant, cost, constraints, and trajectory is $(\Sigma, J, D, \{x_k, u_k, y_k\}_{k\in\mathbb{N}})$ or $(\hat{\Sigma}, \hat{J}, \hat{D}, \{z_k, v_k, w_k\}_{k\in\mathbb{N}})$.

Let ψ be an isomorphism such that $\psi_*\Sigma = \hat{\Sigma}$, $\psi_*J = \hat{J}$, and $\psi_*D = \hat{D}$. If the plant is Σ, we send $\psi_*\Sigma$, ψ_*J, and matrix ψ_*D. If the plant is $\hat{\Sigma}$, we send $\hat{\Sigma}$, $\hat{J}$, and matrix $\hat{D}$. By Lemma 3.4, ψ takes trajectories $\{x_k, u_k, y_k\}_{k\in\mathbb{N}}$ of Σ to trajectories $\{z_k, v_k, w_k\}_{k\in\mathbb{N}}$ of $\psi_*\Sigma$, therefore the cloud receives the same measurements. In response, by Lemma 3.5, the cloud produces the same control inputs when the plant is Σ or $\hat{\Sigma}$.

□

The previous result shows the cloud will not be able to distinguish between any two plants, costs, constraints, or trajectories belonging to the same equivalence class of the $\sim_{\mathcal{G}}$-equivalence relation. In the next section we detail how this result is applied to the three different scenarios.

4 Solving the control-over-the-cloud privacy problem

The results from Section 3 allow us to construct a communication protocol between the plant and the cloud that solves Problems 2.1-2.3. We start by detailing this protocol.

Algorithm 4.1. *(Plant $\Longleftrightarrow$ Cloud)*

1. *Phase 1: Handshaking*

   The plant encodes its dynamics, cost function and constraint matrix into $\hat{\Sigma} = \psi_*\Sigma$, $\hat{J}(z, v) = \psi_*J(x, u)$ *and* $\hat{D} = \psi_*D$ *and sends them to the cloud.*

2. *Phase 2: Execution*

   Encoding: The plant periodically measures y_k*, encodes it into* $w_k = Sy_k$ *and sends it to the cloud.*

   Optimization: The cloud uses the received encoded measurement w_k*, estimates the plant state* z_k*, computes the input* v_k *minimizing* $\hat{J}$ *subject to the constraint* $\hat{D}\eta_k \leq 0$ *and the dynamics* $\hat{\Sigma}$*, and sends* v_k *to the plant.*

   Decoding: The plant decodes v_k *to produce* u_k*, using* (3.3)*, and sends* u_k *to the actuators.*

This protocol will be used in all three scenarios, however, the isomorphism ψ will be chosen from a different group in each case.

To illustrate the protocol on a specific example, we consider its performance for each of the scenarios on the following discrete-time linear plant:

$$\begin{aligned} x_{1,k+1} &= x_{2,k} \\ x_{2,k+1} &= u_k \\ y_k &= x_{1,k}, \end{aligned} \tag{4.1}$$

where $x_{i,k}$ denotes i^{th} state variable at time $k \in \mathbb{N}$. Observe that this plant has a relative degree of 2 and it is controllable and observable.

4.1 Keeping dynamics, state, and I/O private

Consider the first scenario in which the cloud does not know anything about the system. In this scenario, to enforce privacy of the plant dynamics, cost, and constraints, the plant encodes this data using an isomorphism $\psi = (P, F, G, S)$ that can be regarded as a private key used to encode and decode the information exchanged with the cloud. In the following result we denote by $\mathcal{G}$ the group of all isomorphisms of a control system Σ.

Corollary 4.2. *Using the protocol described in Algorithm 4.1 and any isomorphism $\psi \in \mathcal{G}$, the following holds:*

1. *the trajectory of the plant in closed loop with the cloud optimizes the cost J subject to the constraints $D\eta_k \leq 0$;*
2. *the cloud is not able to distinguish between $(\Sigma, J, D, \{x_k, u_k, y_k\}_{k\in\mathbb{N}})$ and any other quadruple $(\hat{\Sigma}, \hat{J}, \hat{D}, \{z_k, v_k, w_k\}_{k\in\mathbb{N}})$ in the same equivalence class of the $\sim_{\mathcal{G}}$-equivalence relation.*

This result is a direct solution to Problem 2.1 stated earlier. The privacy guarantee in Corollary 4.2 is expressed in terms of the equivalence classes of the $\sim_{\mathcal{G}}$-equivalence relation. We leave for future work a more detailed description of these equivalence classes and illustrate the result with system (4.1).

Example 4.3. Let us apply an arbitrary isomorphism $\psi = (P, F, G, S)$ to the plant (4.1), where $P \in \mathbb{R}^{2\times 2}$, $F \in \mathbb{R}^{1\times 2}$, $G \in \mathbb{R}$ and $S \in \mathbb{R}$. As a result, the cloud receives a quadruple $(\hat{\Sigma}, \hat{J}, \hat{D}, \{z_k, v_k, w_k\}_{k\in\mathbb{N}})$, where $\hat{\Sigma} = (\hat{A}, \hat{B}, \hat{C})$ is given by (3.1), $\hat{J}$ is given by (3.6), and $\hat{D}$ is given by (3.7). To determine how much knowledge the cloud can extract about the plant, we need to determine which properties of the plant remain invariant under isomorphisms. Controllability, observability, and the relative degree remain invariant. Moreover, for this example (because the relative degree is equal to the number of states), it is known from [20] that this is a complete set of invariants. In other words, the cloud will not learn anything else beyond knowing the plant is controllable, observable, and has relative degree 2.

The cost received by the plant is described by the matrix $\hat{M} = L^{-T}ML^{-1}$. Since the cloud does not know the matrix L (defined in (3.6)) it will not be able to compute M and the same happens with the constraint matrix D.

Regarding the state x_k, the cloud will know that $x_k = P^{-1}z_k$ for an invertible transformation P, however, the cloud will not be able to compute P and thus will not be able to recover x_k. Similarly for the input and output.

4.2 Keeping dynamics, state and I/O private when the cloud knows the plant's sensors and actuators

We now consider the scenario where the cloud does not know the dynamics but knows which sensors and actuators will be used. Because of this knowledge, we can no longer use an arbitrary isomorphism since it could lead to inputs and outputs that are not consistent with existing sensors and actuators. For example, a linear combination of sensor measurements (dictated by an arbitrary invertible transformation S) may result in values that cannot be produced by the existing sensors. Therefore, we need to restrict the group of isomorphisms that will be used. We construct the relevant isomorphisms in two steps. In the first step, we take any isomorphism ψ of the form $(P, 0, I, I)$, where 0 and I denote the zero and identity matrices, respectively, so as to leave the inputs and outputs unaltered while encoding the state and plant realization via P. In the second step we encode the inputs and outputs using a symmetry ψ_2. The resulting isomorphism is then $\psi_2 \circ \psi_1$.

It can be shown that this set of isomorphisms stills forms a group that we denote by $\mathcal{G}_m$.

Corollary 4.4. *Using the protocol described in Algorithm 4.1 and any isomorphism $\psi \in \mathcal{G}_m$, the following holds:*

1. *the trajectory of the plant in closed loop with the cloud optimizes the cost J subject to the constraints $D\eta_k \leq 0$;*
2. *the cloud is not able to distinguish between $(\Sigma, J, D, \{x_k, u_k, y_k\}_{k\in\mathbb{N}})$ and any other quadruple $(\hat{\Sigma}, \hat{J}, \hat{D}, \{z_k, v_k, w_k\}_{k\in\mathbb{N}})$ in the same equivalence class of the $\sim_{G_m}$-equivalence relation.*

This result is a direct solution to Problem 2.2. Again, the privacy guarantee in Corollary 4.4 is expressed in terms of the equivalence classes of the $\sim_{\mathcal{G}_m}$-equivalence relation, but this time on a smaller group $\mathcal{G}_m$.

Example 4.5. Let us apply any isomorphism $\psi = \psi_2 \circ \psi_1$ to the plant (4.1), where $\psi_1 = (P, 0, I, I)$, $P \in \mathbb{R}^{2\times 2}$, $\psi_2 = (\alpha I, 0, \alpha, \alpha)$ and $\alpha \in \mathbb{R}$. Observe that ψ_2 is a symmetry of system in (4.1). As a result, the cloud receives $\hat{\Sigma}$, $\hat{J}$, and $\hat{D}$ where $\hat{\Sigma} = (\hat{A}, \hat{B}, \hat{C})$ is given by (3.1), $\hat{J}$ is given by (3.6), and $\hat{D}$ is given by (3.7). To determine how much knowledge the cloud can extract about the plant, we need to determine which properties of the plant remain invariant under isomorphisms this set of isomorphisms. Since we are no longer changing the input via (3.3), the cloud will learn the transfer function although not the particular realization of the plant. This follows from the fact that ψ_1 does not change the transfer function, just the realization, and ψ_2 will also not change the transfer function (the α factor cancels out when computing the transfer function).

There is no change with respect to the first scenario in what regards the cloud's knowledge about the state and the output, however, the cloud will now learn more about the input since the $v_k = \alpha u_k$, i.e., the cloud will learn the input up to a scalar multiple.

4.3 Keeping I/O values private when a cloud knows both system dynamics and its sensors and actuators

We now consider the scenario where the cloud knows both the dynamics and which sensors and actuators will be used. Because of this knowledge, we can no longer use an arbitrary isomorphism since it could produce a system, which is different from the one expected by the cloud, and could lead to inputs and outputs that are not consistent with existing sensors.

Therefore, we need to further restrict the group of isomorphisms to the group of symmetries of control systems that we denote by $\mathcal{G}_s$.

Corollary 4.6. *Using the protocol described in Algorithm 4.1 and $\psi \in \mathcal{G}_s$, the following holds:*

1. *the trajectory of the plant in closed loop with the cloud optimizes the cost J subject to the constraints $D\eta_k \leq 0$*
2. *the cloud is not able to distinguish between $(\Sigma, J, D, \{x_k, u_k, y_k\}_{k\in\mathbb{N}})$ and any other quadruple $(\Sigma, \hat{J}, \hat{D}, \{z_k, v_k, w_k\}_{k\in\mathbb{N}})$ in the same equivalence class.*

Problem 2.3 corresponds to the scenario where the cloud has the most knowledge and less information can be kept private. We now illustrate the differences with respect to the previous cases.

Example 4.7. Let us apply the symmetry $\psi = (\alpha I, 0, \alpha, \alpha)$ and $\alpha \in \mathbb{R}$ to the plant (4.1). Recalling that the cloud already knows Σ, the cloud receives $\hat{J}$ and $\hat{D}$ where $\hat{J}$ is given by (3.6) and $\hat{D}$ is given by (3.7). We also note, due to the form of the symmetry ψ that $\hat{M} = \frac{1}{\alpha^2}M$ and $\hat{D} = \frac{1}{\alpha}D$, i.e., the cost and constraints will be known up to a scalar.

Similarly to the second scenario, the inputs and outputs will be known up to a scalar multiple, but differently from the previous scenario, the state will be known up to a scalar multiple, in virtue of the matrix P being a multiple of the identity.

5 Conclusion

In this paper, the first steps towards using isomorphisms and symmetries of control systems to protect privacy were presented. In a future paper we will provide a more concrete description of the privacy guarantees for each subgroup of isomorphisms. This will require a careful analysis of the equivalence classes and may even require extending the notion of isomorphism to include the Morse group [20].

References

[1] A. Vick, J. Guhl, and J. Krger, "Model predictive control as a service - concept and architecture for use in cloud-based robot control," in *2016 21st International Conference on Methods and Models in Automation and Robotics (MMAR)*, Aug 2016, pp. 607–612.

[2] T. Hegazy and M. Hefeeda, "Industrial automation as a cloud service," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 10, pp. 2750–2763, Oct 2015.

[3] B. Hoh, T. Iwuchukwu, Q. Jacobson, D. Work, A. M. Bayen, R. Herring, J. C. Herrera, M. Gruteser, M. Annavaram, and J. Ban, "Enhancing privacy and accuracy in probe vehicle-based traffic monitoring via virtual trip lines," *IEEE Transactions on Mobile Computing*, vol. 11, no. 5, pp. 849–864, May 2012.

[4] A. Burg, A. Chattopadhyay, and K. Y. Lam, "Wireless communication and security issues for cyber-physical systems and the internet-of-things," *Proceedings of the IEEE*, vol. 106, no. 1, pp. 38–60, Jan 2018.

[5] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *Proceedings of the 20th USENIX Conference on Security*, ser. SEC'11. Berkeley, CA, USA: USENIX Association, 2011, pp. 6–6. [Online]. Available: http://dl.acm.org/citation.cfm?id=2028067.2028073

[6] D. Gollmann, P. Gurikov, A. Isakov, M. Krotofil, J. Larsen, and A. Winnicki, "Cyber-physical systems security: Experimental analysis of a vinyl acetate monomer plant," in *Proceedings of the 1st ACM Workshop on Cyber-Physical System Security*, ser. CPSS '15. New York, NY, USA: ACM, 2015, pp. 1–12. [Online]. Available: http://doi.acm.org/10.1145/2732198.2732208

[7] F. M. Cleveland, "Cyber security issues for advanced metering infrasttructure (ami)," in *2008 IEEE Power and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century*, July 2008, pp. 1–5.

[8] B. Ghena, W. Beyer, A. Hillaker, J. Pevarnek, and J. A. Halderman, "Green lights forever: Analyzing the security of traffic infrastructure," in *Proceedings of the 8th USENIX Conference on Offensive Technologies*, ser. WOOT'14. Berkeley, CA, USA: USENIX Association, 2014, pp. 7–7. [Online]. Available: http://dl.acm.org/citation.cfm?id=2671293.2671300

[9] J. Sametinger, J. Rozenblit, R. Lysecky, and P. Ott, "Security challenges for medical devices," *Commun. ACM*, vol. 58, no. 4, pp. 74–82, Mar. 2015. [Online]. Available: http://doi.acm.org/10.1145/2667218

[10] M. R. Ross and J. C. Oren, "Systems security engineering: Considerations for a multidisciplinary approach in the engineering of trustworthy secure systems," NIST, Tech. Rep., 11 2016.

[11] H. Sandberg, S. Amin, and K. H. Johansson, "Cyberphysical security in networked control systems: An introduction to the issue," *IEEE Control Systems*, vol. 35, no. 1, pp. 20–23, Feb 2015.

[12] T. Tanaka, M. Skoglund, H. Sandberg, and K. H. Johansson, "Directed Information as Privacy Measure in Cloud-based Control," *ArXiv e-prints*, May 2017.

[13] K. Kogiso and T. Fujita, “Cyber-security enhancement of networked control systems using homomorphic encryption,” in *2015 54th IEEE Conference on Decision and Control (CDC)*, Dec 2015, pp. 6836–6843.

[14] F. Farokhi, I. Shames, and N. Batterham, “Secure and private cloud-based control using semi-homomorphic encryption,” *IFAC-PapersOnLine*, vol. 49, no. 22, pp. 163 – 168, 2016, 6th IFAC Workshop on Distributed Estimation and Control in Networked Systems NECSYS 2016.

[15] Y. Shoukry, K. Gatsis, A. Alanwar, G. J. Pappas, S. A. Seshia, M. Srivastava, and P. Tabuada, “Privacy-aware quadratic optimization using partially homomorphic encryption,” in *2016 IEEE 55th Conference on Decision and Control (CDC)*, Dec 2016, pp. 5053–5058.

[16] C. Wang, K. Ren, and J. Wang, “Secure and practical outsourcing of linear programming in cloud computing,” in *2011 Proceedings IEEE INFOCOM*, April 2011, pp. 820–828.

[17] W. Du and M. J. Atallah, “Secure multi-party computation problems and their applications: A review and open problems,” in *Proceedings of the 2001 Workshop on New Security Paradigms*, ser. NSPW ’01. New York, NY, USA: ACM, 2001, pp. 13–22. [Online]. Available: http://doi.acm.org/10.1145/508171.508174

[18] J. Corts, G. E. Dullerud, S. Han, J. L. Ny, S. Mitra, and G. J. Pappas, “Differential privacy in control and network systems,” in *2016 IEEE 55th Conference on Decision and Control (CDC)*, Dec 2016, pp. 4252–4272.

[19] F. Koufogiannis and G. J. Pappas, “Differential privacy for dynamical sensitive data,” in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, Dec 2017, pp. 1118–1125.

[20] A. S. Morse, “Structural invariants of linear multivariable systems,” *SIAM Journal on Control*, vol. 11, no. 3, pp. 446–465, 1973. [Online]. Available: https://doi.org/10.1137/0311037