2018 IEEE Conference on Decision and Control (CDC)
Miami Beach, FL, USA, Dec. 17-19, 2018

Cloud-based MPC with Encrypted Data

Andreea B. Alexandru Manfred Morari George J. Pappas

Abstract— **This paper explores the privacy of cloud outsourced Model Predictive Control (MPC) for a linear system with input constraints. A client sends her private states to the cloud who performs the MPC computation and returns the control inputs. In order to guarantee that the cloud can perform this computation without obtaining *anything* about the client's private data, we employ a partially homomorphic cryptosystem. We propose protocols for two cloud-MPC architectures: a client-server architecture and a two-server architecture. In the first case, a control input for the system is privately computed by the cloud server with the assistance of the client. In the second case, the control input is privately computed by two independent, non-colluding servers, with no additional requirements from the client. We prove that the proposed protocols preserve the privacy of the client's data and of the resulting control input. Furthermore, we compute bounds on the errors introduced by encryption. We discuss the trade-off between communication, MPC performance and privacy.**

I. Introduction

The increase in the number of connected devices, as well as their reduction in size and resources, determined a growing demand for cloud-based services, in which a centralized powerful server offers storage and processing capabilities to users. However, issues regarding the privacy of the shared data arise, as the users have no control over the actions of the cloud, which can leak or abuse the data it receives.

Model Predictive Control (MPC) is a powerful scheme that is successfully deployed in practice [1] for systems of varying dimension and architecture, including cloud platforms. In competitive scenarios, such as energy generation in the power grid, domestic scenarios, such as heating control in smart houses, or time-sensitive scenarios, such as traffic control, the control scheme should come with privacy guarantees to protect the data of the users from eavesdroppers or from an untrustworthy cloud.

Much effort has been dedicated to secure cloud computing applications. For a single user, fully homomorphic encryption (FHE) [2] guarantees privacy, but at high complexity requirements. For multiple users, functional privacy is required, which can be attained by functional encryption [3], developed only for limited functionalities. More tractable solutions that involve interactions between the participating parties to ensure the confidentiality of the users' data have been proposed: in client-server computation, we mention partially homomorphic encryption (PHE) [4] and differential privacy (DP) [5]; in two-server computation, solutions using garbled circuits [6], secret sharing [7] and PHE [8] are available.

This work was supported in part by ONR N00014-17-1-2012, and by NSF CNS-1505799 grant and the Intel-NSF Partnership for Cyber-Physical Systems Security and Privacy.

The authors are with the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104. {aandreea, morari, pappasg}@seas.upenn.edu.

A. Contributions

In this paper, we discuss the implicit MPC computation for a linear system with input constraints. We compute the control input, while maintaining the privacy of the state, using a cryptosystem that is partially homomorphic, i.e., supports additions of encrypted data. We first consider a case where the control input is privately computed by a server, with the help of the client. In the second case, the computation is performed by two non-colluding servers, with no requirements from the client. We use a privacy model that stipulates that *no computationally efficient algorithm run by the cloud can infer anything about the private data*, or, in other words, an adversary doesn't know more about the private data than a random guess. Although this model is very strict, it thoroughly characterizes the loss of information.

This work explores fundamental issues of privacy in control: the trade-off between efficiency, performance and privacy. We present two main contributions: proposing two privacy-preserving protocols for MPC and evaluating the errors induced by the encryption. A more detailed version of this paper, that includes numerical results, is available at [9].

B. Related work

Differentially private distributed MPC was addressed in [10]. Encrypted controllers were introduced in [11] and [12] using PHE, and in [13] with FHE. Optimization problems with DP were addressed in [14], [15] and PHE in [16], [17].

Recent work in [18] has tackled the problem of privately computing the input for a constrained linear system using explicit MPC, in a client-server setup. There, the client performs the computationally intensive trajectory localization and sends the result to the server, which then evaluates the corresponding affine control law on the encrypted state using PHE. In our work, we focus on implicit MPC.

The performance degradation of a linear controller due to encryption is analyzed in [19]. In our work, we investigate performance degradation for the nonlinear control obtained from MPC.

II. Problem setup

We consider a discrete-time linear time-invariant system:

$$x(t+1) = Ax(t) + Bu(t), \tag{1}$$

with the state $x \in \mathcal{X} \subseteq \mathbb{R}^n$ and the control input $u \in \mathcal{U} \subseteq \mathbb{R}^m$. The optimal control receding horizon problem with constraints on the states and inputs can be written as:

978-1-5386-1395-5/18/$31.00 ©2018 IEEE

$$J_N^*(x(t)) = \min_{u_{0,\dots,N-1}} \frac{1}{2}\left(x_N^\intercal P x_N + \sum_{k=0}^{N-1} x_k^\intercal Q x_k + u_k^\intercal R u_k\right)$$
$$s.t.\ x_{k+1} = Ax_k + Bu_k,\ k = 0,\dots,N-1 \quad (2)$$
$$x_k \in \mathcal{X},\ u_k \in \mathcal{U},\ k = 0,\dots,N-1$$
$$x_N \in \mathcal{X}_f,\quad x_0 = x(t),$$

where N is the horizon length and $P, Q, R \succ 0$ are cost matrices. For reasons related to error bounding, we consider input constrained systems: $0 \in \mathcal{U} = \{-l_u \preceq u \preceq h_u\}$, $\mathcal{X} = \mathbb{R}^n, \mathcal{X}_f = \mathbb{R}^n$, and impose stability with appropriately chosen costs and horizon such that the closed-loop system has robust performance to bounded errors due to encryption, described in Section VI. A survey on the conditions for stability of MPC is given in [20].

Through straightforward manipulations, (2) can be written as a quadratic program (see details in [21, Ch. 8,11]) in the variable $U := [u_0\ u_1\ \dots\ u_{N-1}]^\intercal$.

$$U^*(x(t)) = \underset{U\in\mathcal{U}}{\operatorname{argmin}}\ 1/2\ U^\intercal H U + U^\intercal F^\intercal x(t) \quad (3)$$

For the sake of simplicity, we keep the same notation for the augmented constraint set $\mathcal{U}$. After obtaining the optimal solution, the first m components of $U^*(x(t))$ are applied as input to the system (1): $u^*(x(t)) = (U^*(x(t)))_{1:m}$.

A. Solution without privacy requirements

The constraint set $\mathcal{U}$ is a hyperbox, so the projection step required for solving (3) has a simple closed form solution and the optimization problem can be efficiently solved with the projected Fast Gradient Method (FGM) [22], given in Algorithm 1. The objective function is strongly convex, since $H \succ 0$, therefore we can use the constant step sizes $L = \lambda_{max}(H)$ and $\eta = (\sqrt{\kappa(H)} - 1)/(\sqrt{\kappa(H)} + 1)$, where $\kappa(H)$ is the condition number of H. Warm starting can be used at subsequent time steps of the receding horizon problem by using part of the previous solution U_K to construct a new feasible initial iterate.

ALGORITHM 1: Projected Fast Gradient Descent

Input: $H, F, x(t), \mathcal{U}, L, \kappa(H), \eta, U_0 \in \mathcal{U}, z_0 = U_0, K$
Output: $U_K(x(t))$
1: **for** k=0…,K-1 **do**
2: $t_k = (\mathbf{I}_{Nm} - \frac{1}{L}H)z_k - \frac{1}{L}F^\intercal x(t)$
3: $U_{k+1}^i = \begin{cases} -l_u^i, & \text{if } t_k^i < -l_u^i \\ t_k^i, & \text{if } t_k^i \in [-l_u^i,\ h_u^i] \\ h_u^i, & \text{if } t_k^i > h_u^i \end{cases},\ i = 1,\dots,Nm$
4: $z_{k+1} = (1+\eta)U_{k+1} - \eta U_k$
5: **end for**

B. Privacy objectives

The unsecure cloud-MPC problem is depicted in Figure 1. The system's constant parameters A, B, P, Q, R, N are public, motivated by the fact the parameters are intrinsic to the system and hardware, and could be guessed or identified; however, the measurements, control inputs and constraints are not known and should remain private. We want to devise private cloud-outsourced versions of Algorithm 1 such that the client obtains the control input $u^*(t)$ with only a minimum amount of work. The cloud (consisting of either one or two servers) should not infer anything else than what was known prior to the computation about the measurements $x(t)$, the control inputs $u^*(t)$ and the constraints $\mathcal{U}$. We tolerate **semi-honest** servers, meaning that they correctly follow the steps of the protocol but may store the transcript of the messages exchanged and process the data received to try to learn more information than allowed.

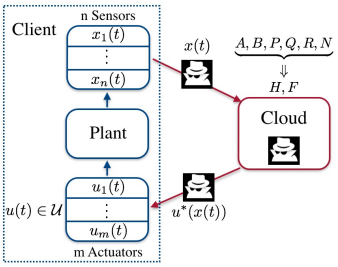

Fig. 1. Unsecure MPC: the system model, horizon and costs are public. The states, control inputs and input constraints are privacy-sensitive.

To formalize the privacy objectives, we introduce the privacy definitions that we want our protocols to satisfy, described in [23, Ch. 7]. In what follows, $\{0,1\}^*$ defines a sequence of bits of unspecified length. Given a countable index set I, an ensemble $X = \{X_i\}_{i\in I}$, indexed by I, is a sequence of random variables X_i, for all $i \in I$.

Two ensembles are called computationally indistinguishable if no *efficient* algorithm can distinguish between them.

Definition 1: The ensembles $X = \{X_n\}_{n\in\mathbb{N}}$ and $Y = \{Y_n\}_{n\in\mathbb{N}}$ are **computationally indistinguishable**, denoted $\overset{c}{\equiv}$, if for every polynomial-time algorithm D, every positive polynomial p and all sufficiently large n, the following holds:

$$|\Pr[D(X_n) = 1] - \Pr[D(Y_n) = 1]| < 1/p(n).$$

The definition of **two-party privacy** says that a protocol privately computes the functionality it runs if all information obtained by a party after the execution of the protocol, while also keeping a record of the intermediate computations, can be obtained only from the inputs and outputs of that party.

Definition 2: Let $f\colon (\{0,1\}^*)^2 \to (\{0,1\}^*)^2$ be a functionality, and $f_i(x_1, x_2)$ be the ith component of $f(x_1, x_2)$, $i = 1, 2$. Let Π be a two-party protocol for computing f. The **view** of the ith party during an execution of Π on the inputs (x_1, x_2), denoted by $V_i^\Pi(x_1, x_2)$, is $(x_i, coins, m_1, \dots, m_t)$, where $coins$ represents the outcome of the ith party's internal coin tosses, and m_j represents the jth message it has received. For a deterministic functionality f, we say that Π **privately computes** f if there exist probabilistic polynomial-time algorithms, called **simulators**, denoted by S_i, such that:

$$\{S_i(x_i, f_i(x_1, x_2))\}_{x_{1,2}\in\{0,1\}^*} \overset{c}{\equiv} \{V_i^\Pi(x_1, x_2)\}_{x_{1,2}\in\{0,1\}^*}.$$

The purpose of the paper is to design protocols with the functionality of Algorithm 1 that satisfy Definition 2. To this end, we use the encryption scheme defined in Section III. In Sections IV and V, we address two private cloud-MPC solutions that present a trade-off between the computational effort at the client and the total time required to compute the solution $u^*(t)$. We discuss in Section VI how to connect the domain of the inputs in Definiton 2 with the domain of real numbers needed for the MPC problem.

III. Partially homomorphic cryptosystem

In this paper, we use the Paillier cryptosystem [4], which is an asymmetric additively homomorphic encryption scheme. The message space for the Paillier scheme is $\mathbb{Z}_{N_\sigma}$, where N_σ is a large integer that is the product of two prime numbers p, q. The pair of keys is (pk, sk), where the public key is $pk = (N_\sigma, g)$, with $g \in \mathbb{Z}_{N_\sigma^2}$ having order N_σ and the secret key is $sk = (\gamma, \delta)$: $\gamma = \text{lcm}(p-1, q-1)$, $\delta = ((g^\gamma \bmod N_\sigma^2 - 1)/N_\sigma)^{-1} \bmod N_\sigma$.

For a message $a \in \mathbb{Z}_{N_\sigma}$, called plaintext, the Pailler encryption primitive is defined as: $[[a]] := g^a r^{N_\sigma} \bmod N_\sigma^2$, with r random value in $\mathbb{Z}_{N_\sigma}$. Intuitively, the additively homomorphic property means that there exists an operator $\oplus$ defined on the space of encrypted messages, called ciphertexts, such that: $[[a]] \oplus [[b]] = [[a+b]], \ \ \forall a, b \in \mathbb{Z}_{N_\sigma}$. Formally, the decryption primitive is a homomorphism between the group of ciphertexts with the operator $\oplus$ and the group of plaintexts with addition $+$. The scheme also supports multiplication between a plaintext and an encrypted message, obtained by adding the encrypted message for the corresponding integer number of times: $b \otimes [[a]] = [[ba]]$. We will use the same notation for operations on vectors and matrices.

Proving the privacy of a protocol that makes use of cryptosystems involves the concept of **semantic security** [23, Ch. 5]. Under the assumption of decisional composite residuosity [4], the Paillier cryptosystem is semantically secure and has indistinguishable encryptions, which, in essence, means that an adversary cannot distinguish between the ciphertext $[[a]]$ and a ciphertext $[[b]]$ based on the messages a and b.

To summarize, PHE allows a party that does not have the private key to perform linear operations on encrypted integer data. For instance, a cloud-based Linear Quadratic Controller can be computed entirely by one server, because the control action is linear in the state. Nonlinear operations are not supported within this cryptosystem, but can be achieved with communication between the party that has the encrypted data and the party that has the private key.

IV. Client-Server architecture

To be able to use the Paillier encryption, we need to represent the messages on a finite set of integers, parametrized by N_σ, i.e., each message is an element in $\mathbb{Z}_{N_\sigma}$.

Notation: Given a real quantity $x \in \mathbb{R}$, we use the notation $\bar{x}$ for the corresponding quantity in fixed-point representation on one sign bit, l_i integer and l_f fractional bits.

In this section and Section V, we consider a fixed-point representation of the values and perform implicit multiplication steps to obtain integers and division steps to retrieve the true values. We analyze the implications of the fixed-point representation over the MPC solution in Section VI. We drop the $\bar{(\cdot)}$ from the variables in order to not burden the notation.

We introduce a client-server (CS) model in Figure 2 and present an interactive protocol that privately computes the control input, while maintaining the privacy of the state of the client, in Protocol 2. The Paillier encryption is not order preserving, due to the random value required in the encryption primitive, so the projection operation cannot be performed locally by the server. Hence, the server sends the encrypted iterate $[[t_k]]$ to the client to project it. Then, the latter encrypts the feasible iterate and sends it back to the server.

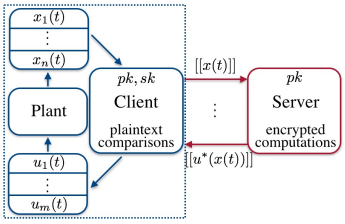

Fig. 2. Private client-server (CS) setup for MPC.

Protocol 2: Encrypted MPC in a CS architecture

Input: C: $x(t), K, l_i, l_f, \bar{\mathcal{U}}, pk, sk$; S: $\bar{H}_f, \bar{F}, \bar{\eta}, K, l_i, l_f, pk, [[U_0]]$
Output: C: $u = (U_K(x(t)))_{1:m}$

C: Encrypt and send $[[x(t)]]$ to S
S: $[[z_0]] = [[U_0]]$
for k=0…,K-1 **do**
  S: $[[t_k]] = (\mathbf{I}_{Nm} - \bar{H}_f) \otimes [[z_k]] \oplus (-\bar{F}_f^\intercal) \otimes [[x(t)]]$, send it to C
  C: Decrypt t_k and truncate to l_f fractional bits
  C: $U_{k+1} = \Pi_{\bar{\mathcal{U}}}(t_k)$ ▷ Projection on $\bar{\mathcal{U}}$
  C: Encrypt and send $[[U_{k+1}]]$ to S
  S: $[[z_{k+1}]] = (1 + \bar{\eta}) \otimes [[U_{k+1}]] \oplus (-\bar{\eta}) \otimes [[U_k]]$
end for
C: Decrypt and output $u = (U_K)_{1:m}$

We assume that $[[U_0]]$ already captures the information of whether the iteration is cold or warm start.

Theorem 1: Protocol 2 achieves privacy as in Definition 2 with respect to a semi-honest server.

Proof: The initial value of the iterate gives no information to the server about the result, as the final result is encrypted and the number of iterations is a priori fixed. The view of the server, as in Definition 2, is composed of its inputs, the messages received $\{[[U_k]]\}_{k=0,\dots,K}$, which are all encrypted, and no output. We construct a simulator that replaces the messages with random encryptions of corresponding length. Due to the semantic security of the Paillier cryptosystem, proved in [4], the view of the simulator is computationally indistinguishable from the view of the server. ■

V. Two-server architecture

Although in Protocol 2, the client needs to store and process substantially less data than the server, the computational requirements might be too stringent for large values of K and N_σ. In such a case, we outsource the problem to two servers (SS), and only require the client to encrypt $x(t)$, send it to one server and decrypt the received result $[[u^*]]$. In this setup, depicted in Figure 3, the existence of two non-colluding servers is assumed.

In Figure 3 and Protocol 3, we denote by $[[\cdot]]$ a message encrypted by pk_1 and by $[\{\cdot\}]$ a message encrypted by pk_2. We use two pairs of keys so that the client and second server do not have the same private key and do not need to interact.

As before, we need an interactive protocol to achieve the projection. We use the DGK comparison protocol, proposed in [24], such that, given two encrypted values of l bits $[[a]], [[b]]$ to S_1, after the protocol, S_2 obtains a bit $(\beta = 1) \equiv (a \leq b)$, without finding anything about the inputs. Moreover, S_1 finds nothing about β. We augment this protocol by introducing a step before the comparison in which S_1

randomizes the order of the two values to be compared, such that S_2 does not know the significance of β with respect to the inputs. Furthermore, by performing a blinded exchange, S_1 obtains the minimum (respectively, maximum) value of the two inputs, without any of the two servers knowing what the result is. The above procedure is performed in lines 6–11 in Protocol 3. More details can be found in [17].

In order to guarantee that S_2 does not find out the private values after decryption, S_1 adds sufficiently large random noises to the messages. The random numbers in lines 8 and 14 are chosen from $(0, 2^{l+\lambda_\sigma}) \cap \mathbb{Z}_{N_\sigma}$, which ensures the indistinguishability between the sum of the random number and the private value and a random number of equivalent length [25], where λ_σ is the statistical security parameter.

Since the variables we compare are results of additions and multiplications, we need to ensure that they are represented on l bits before performing the comparison protocol. This introduces a truncation step in line 5: S_1 adds noise to t_k and sends it to S_2 which decrypts it, truncates the result to l bits and sends it back. S_1 then subtracts the truncated noise.

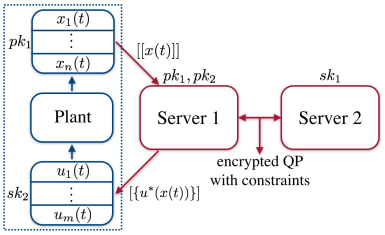

Fig. 3. Private two-server (SS) setup for MPC.

PROTOCOL 3: Encrypted MPC in a SS architecture

Input: C: $x(t), pk_1, pk_2, sk_2$; S_1: $\bar{H}_f, \bar{F}, \bar{\eta}, K, l_i, l_f, pk_1, pk_2, [[U_0]]$; S_2: $K, l_i, l_f, pk_1, pk_2, sk_1$

Output: C: $u = (U_K(x(t)))_{1:m}$

1: C: Encrypt and send $[[x(t)]], [[h_u]], [[-l_u]]$ to S_1
2: S_1: $[[z_0]] \leftarrow [[U_0]]$
3: **for** k=0…,K-1 **do**
4: S_1: $[[t_k]] \leftarrow (\mathbf{I}_{Nm} - \bar{H}_f) \otimes [[z_k]] \oplus (-\bar{F}_f^\intercal) \otimes [[x(t)]]$
5: S_1, S_2: $[[t_k]] \leftarrow$ truncate $[[t_k]]$
6: S_1: $a_k, b_k \leftarrow$ randomize $[[t_k]], [[h_u]]$
7: S_1, S_2: DGK s.t. S_2 obtains $(\beta_k = 1) \equiv (a_k \leq b_k)$
8: S_1: Pick r_k, s_k and send $[[a_k]] \oplus [[r_k]], [[b_k]] \oplus [[s_k]]$ to S_2
9: S_2: Send back $[[\beta_k]]$ and $[[v_k]] \leftarrow [[a_k + r_k]] \oplus [[0]]$ if $\beta_k = 1$ or $[[v_k]] \leftarrow [[b_k + s_k]] \oplus [[0]]$ if $\beta_k = 0$
10: S_1: $[[U_{k+1}]] \leftarrow [[v_k]] \oplus s_k \otimes ([[\beta_k]] \oplus [[-1]]) \oplus r_k \otimes [[\beta_k]]$ ▷ $U_{k+1} \leftarrow \min(t_k, h_u)$
11: S_1, S_2: Redo 6–10 to get $[[U_{k+1}]] \leftarrow \max(U_{k+1}, -l_u)$
12: S_1: $[[z_{k+1}]] \leftarrow (1 + \bar{\eta}) \otimes [[U_{k+1}]] \oplus (-\bar{\eta}) \otimes [[U_k]]$
13: **end for**
14: S_1: Pick ρ and send $[[(U_K)_{1:m}]] \oplus [[\rho]]$ to S_2
15: S_2: Decrypt and re-encrypt with pk_2 and send to S_1: $[\{u + \rho\}]$
16: S_1: $[\{u\}] \leftarrow [\{u + \rho\}] \oplus [\{-\rho\}]$ and send it to C
17: C: Decrypt and output u

Theorem 2: Protocol 3 achieves privacy as in Definition 2, as long as the two semi-honest servers do not collude.

Proof: The view of S_1 is composed by its inputs and received messages, and no output. All the messages are encrypted (the same holds for the comparison subprotocol). Furthermore, in line 9, an encryption of zero is added to the quantity S_1 receives such that the encryption is re-randomized and S_1 cannot recognize it. Due to the semantic security of the cryptosystems, the view of S_1 is computationally indistinguishable from the view of a simulator which follows the same steps as S_1, but replaces the incoming messages by random encryptions of corresponding length.

The view of S_2 is composed by its inputs and received messages, and no output. Apart from the comparison bits, the messages are always blinded by noise that has at least λ_σ bits more than the private data being sent, with λ_σ chosen appropriately large (e.g. 100 bits [25]).

Crucially, the noise selected by S_1 is different at each iteration. Hence, S_2 cannot extract any information by combining messages from multiple iterations, as they are always blinded by a different large enough noise. Moreover, the randomization step in line 6 ensures that S_2 cannot infer anything from the values of β_k, as the order of the inputs is unknown. Thus, we construct a simulator that follows the same steps as S_2, but instead of the received messages, it randomly generates values of appropriate length, corresponding to the blinded private values, and random bits corresponding to the comparison bits. The view of such a simulator will be computationally indistinguishable from the view of S_2. ■

Remark 1: One can expand Protocols 2 and 3 over multiple time steps, such that U_0 is obtained from the previous iteration and not given as input, and formally prove their privacy. A more detailed proof that explicitly constructs the simulators can be found in [17].

VI. FIXED-POINT PRECISION MPC

We consider fixed-point representations with one bit sign, l_i integer bits and l_f fractional bits and multiply them by 2^{l_f} to obtain integers. Working with fixed-point representations can lead to overflow, quantization and arithmetic round-off errors. Thus, we want to compute the deviation between the fixed-point solution and optimal solution of Algorithm 1.

In order to preserve the feasibility of the fixed-point precision solution, ensure that the strong convexity of the fixed-point objective function still holds and the fixed-point step size is such that FGM converges, we consider the following:

Assumption 1: The number of fractional bits l_f and constant $c \geq 1$ are chosen large enough such that:

(i) $\bar{\mathcal{U}} \subseteq \mathcal{U}$: the fixed-point precision solution is still feasible.

(ii) The eigenvalues of the fixed-point representation $\bar{H}_f$ are contained in the set $(0,\ 1]$, where $H_f := \bar{H}/(c\bar{L})$ and $\bar{L} := \lambda_{max}(\bar{H})$. The constant c is required in order to overcome the possibility that $\overline{(1/\bar{L})\bar{H}}$ has the maximum eigenvalue larger than 1 due to fixed-point errors.

(iii) The fixed-point representation of the step size satisfies:

$$0 \leq \left(\sqrt{\kappa(\bar{H})} - 1\right) \Big/ \left(\sqrt{\kappa(\bar{H})} + 1\right) \leq \bar{\eta} < 1.$$

Overflow errors: Bounds on the infinity-norm on the fixed-point dynamic quantities of interest in Algorithm 1 were derived in [26] for each iteration k, and depend on a bounded set $\mathcal{X}_0$ such that $x(t) \in \mathcal{X}_0$ and $\bar{x}(t) \in \bar{\mathcal{X}}_0$:

$$\begin{aligned} ||\bar{U}_{k+1}||_\infty &\leq \max\{||\bar{l}_u||_\infty, ||\bar{h}_u||_\infty\} = R_{\bar{\mathcal{U}}} \\ ||\bar{z}_{k+1}||_\infty &\leq (1 + 2\bar{\eta})\mathcal{R}_{\bar{\mathcal{U}}} := \zeta, \\ ||\bar{t}_k||_\infty &\leq ||\mathbf{I}_{Nm} - \bar{H}_f||_\infty \zeta + ||\bar{F}_f||_\infty \mathcal{R}_{\bar{\mathcal{X}}_0}, \end{aligned}$$

where $F_f = \bar{F}/(c\bar{L})$ and $\mathcal{R}_{\mathcal{S}}$ represents the radius of a set $\mathcal{S}$ w.r.t. the infinity norm. We select from these bounds the number of integer bits l_i such that there is no overflow.

A. Difference between real and fixed-point solution

Denote by U_K the solution in exact arithmetic of the MPC problem (3) obtained after K iterations of Algorithm 1. Furthermore, denote by $\tilde{U}_K$ the solution obtained after K iterations but with $H, F, x(t), \mathcal{U}, L, \eta$ replaced by their fixed-point representations. Finally, denote by $\bar{U}_K$ the solution of Protocols 2 and 3 after K iterations, where the iterates $[[t_k]], [[U_k]]$ have fixed-point representation. We obtain the following upper bound on the difference between the solution obtained on the encrypted data and the nominal solution of the MPC problem (3) after K iterations:

$$||\bar{U}_K - U_K||_2 \leq ||\tilde{U}_K - U_K||_2 + ||\bar{U}_K - \tilde{U}_K||_2.$$

1) Quantization errors: We use the following remark to investigate the quantization error bounds. Define $\epsilon_a = \bar{a} - a$ and $\epsilon_b = \bar{b} - b$. Then, $\bar{a}\bar{b} - ab = \bar{a}\bar{b} - \bar{a}b + \bar{a}b - ab = \epsilon_a b + \bar{a}\epsilon_b$.

Consider iteration k of the projected FGM where the coefficients are replaced by the fixed-point representations of the matrices $H/L, F/L$, the vector $x(t)$ and the set $\mathcal{U}$. The errors induced by quantization of the coefficients between the original iterates and the approximation iterates are:

$$\begin{aligned} \tilde{t}_k - t_k =& -\epsilon_{H_f} z_k + (\mathbf{I}_{Nm} - \bar{H}_f)\epsilon_{z,k} - \epsilon_{Fx} \\ \xi^q_{k+1} :=& \tilde{U}_{k+1} - U_{k+1} = D^q_k(\tilde{t}_k - t_k) \\ \tilde{z}_{k+1} - z_{k+1} =& \epsilon_\eta \Delta U_k + (1+\bar{\eta})\xi^q_{k+1} - \bar{\eta}\xi^q_k, \end{aligned} \tag{4}$$

where we used the notation: $\Delta U_k = U_{k+1} - U_k$; $\epsilon_\eta = \bar{\eta} - \eta$; $\epsilon_{H_f} = \bar{H}_f - H/(cL)$; $\epsilon_{Fx} = \bar{F}_f^\intercal \bar{x}(t) - F^\intercal x(t)/(cL) = \epsilon_{F_f}^\intercal x(t) + \bar{F}_f \epsilon_x$; $\epsilon_x = \bar{x}(t) - x(t)$; $\epsilon_{F_f} = \bar{F}_f - F/(cL)$.

The error ξ^q_{k+1} is reduced from $\tilde{t}_k - t_k$ due to the projection on the hyperbox. We represent this in (4) via a diagonal matrix D^q_k with positive elements at most one.

We set $\xi^q_{-1} = \xi^q_0$. From (4), we derive a recursive iteration that characterizes the error of the primal iterate, for $k = 0, \ldots, K$, which we can write as a linear system:

$$\begin{aligned} \tilde{A}(D^q_k) :=& \begin{bmatrix} (1+\bar{\eta})D^q_k(\mathbf{I}_{Nm} - \bar{H}_f) & -\bar{\eta}D^q_k(\mathbf{I}_{Nm} - \bar{H}_f) \\ \mathbf{I}_{Nm} & \mathbf{0}_{Nm} \end{bmatrix} \\ \tilde{B}(D^q_k) :=& \begin{bmatrix} -\epsilon_{H_f} D^q_k & \epsilon_\eta D^q_k(\mathbf{I}_{Nm} - \bar{H}_f) \\ \mathbf{0}_{Nm} & \mathbf{0}_{Nm} \end{bmatrix} \\ \begin{bmatrix} \xi^q_{k+1} \\ \xi^q_k \end{bmatrix} =& \tilde{A}(D^q_k) \begin{bmatrix} \xi^q_k \\ \xi^q_{k-1} \end{bmatrix} + \tilde{B}(D^q_k) \begin{bmatrix} z_k \\ \Delta U_{k-1} \end{bmatrix} - \epsilon_{Fx}. \end{aligned} \tag{5}$$

We choose this representation in order to have a relevant error bound in Theorem 3, that shrinks to zero as the number of fractional bits grows. In the following, we find an upper bound of the error using $\tilde{A} := \tilde{A}(\mathbf{I}_{Nm})$ and $\tilde{B} := \tilde{B}(\mathbf{I}_{Nm})$.

Theorem 3: Under Assumption 1, the system defined by (5) is bounded. Furthermore, the norm of the error between the primal iterates of the original problem and of the problem with quantized coefficients is bounded by:

$$\begin{aligned} &||\xi^q_{k+1}||_2 \leq \left|\left|E\tilde{A}^k\right|\right|_2 \left|\left|\begin{bmatrix} \xi^q_0 \\ \xi^q_{-1} \end{bmatrix}\right|\right|_2 + \gamma \sum_{l=0}^{k-1} \left|\left|E\tilde{A}^{k-1-l}\tilde{B}\right|\right|_2 + \\ &+ \zeta \sum_{l=0}^{k-1} \left|\left|E\tilde{A}^{k-1-l}\right|\right|_2 =: \epsilon_1; \quad \gamma = (3+2\bar{\eta})\sqrt{Nm}\mathcal{R}_{\bar{\mathcal{U}}}; \\ &\zeta = ||\epsilon_{F_f}||_2 \mathcal{R}^2_{\mathcal{X}_0} + 2^{-l_f}\sqrt{n}||\bar{F}_f||_2, \end{aligned}$$

where $E = [\mathbf{I}_{Nm} \;\; \mathbf{0}_{Nm}]$, $\mathcal{R}^2_{\mathcal{X}_0}$ is the radius of the compact set $\mathcal{X}_0$ w.r.t. the 2-norm and $\mathcal{R}_{\bar{\mathcal{U}}} = \max\{||l_u||_\infty, ||h_u||_\infty\}$.

Proof: The inner stability of the system is given by the fact that $\tilde{A}$ has spectral radius $\rho(\tilde{A}) < 1$, proven in Lemma 1 in [26]. We use that $||\tilde{A}(D^q_k)||_2 \leq ||\tilde{A}||_2$ (resp., $||\tilde{B}(D^q_k)||_2 \leq ||\tilde{B}||_2$) and express the bounds in terms of the latter.

From (5), one can obtain the following expression for the errors at time $k+1$ and k, for $k = 0, \ldots, K-1$:

$$\begin{bmatrix} \xi^q_{k+1} \\ \xi^q_k \end{bmatrix} \leq \tilde{A}^k \begin{bmatrix} \xi^q_0 \\ \xi^q_{-1} \end{bmatrix} + \sum_{l=0}^{k-1} \tilde{A}^{k-1-l} \left(\tilde{B} \begin{bmatrix} z_l \\ \Delta U_{l-1} \end{bmatrix} - \epsilon_{Fx} \right),$$

and the first term goes to zero as $k \to \infty$. We multiply this by $E = [\mathbf{I}_{Nm} \; \mathbf{0}_{Nm}]$ to obtain the expression of $||\xi^q_{k+1}||_2$.

Subsequently, for any $0 \leq k \leq K-1$:

$$\begin{aligned} &\left|\left|\begin{bmatrix} z_k^\intercal & \Delta U_{k-1}^\intercal \end{bmatrix}^\intercal\right|\right|_2 \leq ||U_k + \bar{\eta}\Delta U_{k-1}||_2 + ||\Delta U_{k-1}||_2 \\ &\leq (3 + 2\bar{\eta})\sqrt{Nm(\max_i\{l^i_u, h^i_u\})^2} := \gamma; \\ &||\epsilon_{Fx}||_2 \leq ||\epsilon_{F_f}||_2 ||x(t)||_2 + ||\bar{F}_f||_2 ||\epsilon_x||_2 \\ &\leq ||\epsilon_{F_f}||_2 \mathcal{R}^2_{\mathcal{X}_0} + 2^{-l_f}\sqrt{n}||\bar{F}_f||_2 := \zeta. \end{aligned}$$ ■

Remark 2: In primal-dual algorithms, the maximum values of the dual variables corresponding to the complicating constraints cannot be bounded a priori, i.e., we cannot give overflow or quantization error bounds. This justifies our focus on a problem with only simple input constraints.

2) Arithmetic round-off errors: The encrypted values do not necessarily maintain the same number of bits after operations, so we might have round-off errors where we perform truncations (line 5 in Protocols 2 and 3). In this case, we obtain similar results to [26], where the quantization errors were not analyzed. Consider iteration k of the projected FGM. The errors due to round-off between the primal iterates of the two solutions are:

$$\begin{aligned} \bar{t}_k - \tilde{t}_k =& (\mathbf{I}_{Nm} - \bar{H}_f)(\bar{z}_k - \tilde{z}_k) + \epsilon'_{t,k} \\ \xi^r_{k+1} :=& \bar{U}_{k+1} - \tilde{U}_{k+1} = D^r_k(\bar{t}_k - t_k) \\ \bar{z}_{k+1} - \tilde{z}_{k+1} =& (1+\bar{\eta})\xi^r_{k+1} - \bar{\eta}\xi^r_k. \end{aligned} \tag{6}$$

Again, the projection on the hyperbox reduces the error, so D^r_k has positive diagonal elements less than one. For Protocol 2, the round-off error due to truncation is $(\epsilon'_{t,k})^i \in [-Nm2^{-l_f},\ 0]$, $i = 1, \ldots, Nm$. The encrypted truncation step in Protocol 3 introduces an extra term, making $(\epsilon'_{t,k})^i \in [-(1+Nm)2^{-l_f},\ 2^{-l_f}]$.

We set $\xi^r_{-1} = \xi^r_0$. From (6), we derive a recursive iteration that characterizes the error of the primal iterate, which we can write as a linear system, with $\tilde{A}(\cdot)$ as before:

$$\begin{bmatrix} \xi^r_{k+1} \\ \xi^r_k \end{bmatrix} = \tilde{A}(D^r_k) \begin{bmatrix} \xi^r_k \\ \xi^r_{k-1} \end{bmatrix} + D^r_k \epsilon'_{t,k}. \tag{7}$$

Theorem 4: Under Assumption 1, the system defined by (7) is bounded. Furthermore, the norm of the error of the primal iterate is bounded by:

$$\begin{aligned} &||\xi^r_k||_2 \leq \left|\left|E\tilde{A}^k\right|\right|_2 \left|\left|\begin{bmatrix} \xi^r_0 \\ \xi^r_{-1} \end{bmatrix}\right|\right|_2 + \gamma' \sum_{l=0}^{k-1} \left|\left|E\tilde{A}^{k-1-l}\right|\right|_2 =: \epsilon_2, \\ &\gamma'_{CS} = 2^{-l_f}(Nm)^{\frac{3}{2}}; \gamma'_{SS} = 2^{-l_f}\sqrt{Nm}(1+Nm). \end{aligned}$$

The proof is straightforward.

One can eliminate the initial errors ξ^q_0 and ξ^r_0 and their

effects by choosing the same initial iterates represented on l_f fractional bits for both problems.

Remark 3: As $l_f \to \infty$, $\epsilon_1 \to 0$ and $\epsilon_2 \to 0$. The persistent noise in (5) and (7), which is composed by quantization errors and round-off errors, becomes zero when the number of fractional bits mimics a real value.

B. Trade-off between performance and privacy

We can incorporate the error $\epsilon := \epsilon_1 + \epsilon_2$ either as a bounded disturbance in system (1) and design the terminal cost as described in [27], so that the controller achieves inherent robust stability, or, alternatively, as a suboptimality in the cost $\bar{J}_N(x(t), \bar{U}_K)$, and seek asymptotic stability as in [28].

For every instance of problem (2), the error bounds can be computed as a function of the number of integer and fractional bits. Therefore, in the offline phase, the fixed-point precision of the variables is chosen such that there is no overflow and one of the conditions on ϵ is satisfied. Note that these conditions can be overly-conservative.

The protocol for the two-server architecture is slower than the protocol for client-server architecture and the reason for that is communication. Performing the projection with PHE requires l communication rounds, where l is the size of the messages compared. Privately updating the iterates requires another communication round. Hence, privacy comes at the price of complexity: hiding the private data requires working with large encrypted numbers, and the nonlinear computations on private data require communication. Furthermore, the more parties that need to be oblivious to the private data (two servers in the second setup compared to one server in the first), the more complex the private protocols become.

VII. Conclusion

We presented two methods to achieve the private computation of the solution to cloud-outsourced MPC via the fast gradient method, using additively homomorphic encryption. The client desires to keep the state, the control inputs and the constraints private. To this end, it encrypts the current state. The cloud can be composed by one or two servers and has to relieve the computation from the client in a private manner. First, we proposed an architecture where the computation is split between the client and one server: the client performs the projection on the constraints, while the server performs the rest of the computations needed to solve the optimization problem. Second, we proposed a two-server architecture, in which the client is exempt from any computations and the two servers use blinded communication to perform the nonlinear operations on encrypted data. We proved that both protocols achieve privacy for semi-honest servers. Furthermore, we analyzed the quantization and round-off errors introduced by encryptions and gave upper bounds which can be used to choose an appropriate precision that corresponds to performance requirements on the MPC.

References

[1] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967–2986, 2014.

[2] C. Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertation, Department of Computer Science, Stanford University, 2009.

[3] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," in *Theory of Cryptography Conference*. Springer, 2011, pp. 253–273.

[4] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 1999, pp. 223–238.

[5] C. Dwork, "Differential privacy: A survey of results," in *International Conference on Theory and Applications of Models of Computation*. Springer, 2008, pp. 1–19.

[6] A. C. Yao, "Protocols for secure computations," in *23rd Symposium on Foundations of Computer Science*. IEEE, 1982, pp. 160–164.

[7] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[8] Z. Erkin, T. Veugen, T. Toft, and R. L. Lagendijk, "Generating private recommendations efficiently using homomorphic encryption and data packing," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 3, pp. 1053–1066, 2012.

[9] A. B. Alexandru, M. Morari, and G. J. Pappas, "Cloud-based MPC with encrypted data," *arXiv preprint arXiv:1803.09891*, 2018.

[10] M. Zellner, T. T. De Rubira, G. Hug, and M. Zeilinger, "Distributed differentially private model predictive control for energy storage," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 12 464–12 470, 2017.

[11] K. Kogiso and T. Fujita, "Cyber-security enhancement of networked control systems using homomorphic encryption," in *54th Annual Conference on Decision and Control*. IEEE, 2015, pp. 6836–6843.

[12] F. Farokhi, I. Shames, and N. Batterham, "Secure and private control using semi-homomorphic encryption," *Control Engineering Practice*, vol. 67, pp. 13–20, 2017.

[13] J. Kim, C. Lee, H. Shim, J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Encrypting controller using fully homomorphic encryption for security of cyber-physical systems," *IFAC-PapersOnLine*, vol. 49, no. 22, pp. 175–180, 2016.

[14] E. Nozari, P. Tallapragada, and J. Cortes, "Differentially private distributed convex optimization via functional perturbation," *IEEE Transactions on Control of Network Systems*, 2016.

[15] S. Han, U. Topcu, and G. J. Pappas, "Differentially private distributed constrained optimization," *IEEE Transactions on Automatic Control*, vol. 62, no. 1, pp. 50–64, 2017.

[16] N. M. Freris and P. Patrinos, "Distributed computing over encrypted data," in *54th Annual Allerton Conference on Communication, Control, and Computing*. IEEE, 2016, pp. 1116–1122.

[17] A. B. Alexandru, K. Gatsis, Y. Shoukry, S. A. Seshia, P. Tabuada, and G. J. Pappas, "Cloud-based quadratic optimization with partially homomorphic encryption," *arXiv preprint arXiv:1809.02267*, 2018.

[18] M. S. Darup, A. Redder, I. Shames, F. Farokhi, and D. Quevedo, "Towards encrypted MPC for linear constrained systems," *IEEE Control Systems Letters*, vol. 2, no. 2, pp. 195–200, 2018.

[19] K. Kogiso, "Upper-bound analysis of performance degradation in encrypted control system," in *2018 Annual American Control Conference*. IEEE, 2018, pp. 1250–1255.

[20] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[21] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[22] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013, vol. 87.

[23] O. Goldreich, *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, 2004.

[24] I. Damgård, M. Geisler, and M. Krøigaard, "Efficient and secure comparison for on-line auctions," in *Australasian Conference on Information Security and Privacy*. Springer, 2007, pp. 416–430.

[25] R. Bost, R. A. Popa, S. Tu, and S. Goldwasser, "Machine learning classification over encrypted data," in *NDSS*, 2015.

[26] J. L. Jerez, P. J. Goulart, S. Richter, G. A. Constantinides, E. C. Kerrigan, and M. Morari, "Embedded online optimization for model predictive control at megahertz rates," *IEEE Transactions on Automatic Control*, vol. 59, no. 12, pp. 3238–3251, 2014.

[27] G. Pannocchia, J. B. Rawlings, and S. J. Wright, "Inherently robust suboptimal nonlinear MPC: theory and application," in *50th Conference on Decision and Control and European Control*. IEEE, 2011, pp. 3398–3403.

[28] L. K. McGovern and E. Feron, "Closed-loop stability of systems driven by real-time, dynamic optimization algorithms," in *38th Conference on Decision and Control*, vol. 4. IEEE, 1999, pp. 3690–3696.