processes

MDPI

Article

Encrypted Model Predictive Control of a Nonlinear Chemical Process Network

Yash A. Kadakia [1], Atharva Suryavanshi [1], Aisha Alnajdi [2], Fahim Abdullah [1] and Panagiotis D. Christofides [1,2,*]

[1] Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA 90095, USA; yash14@g.ucla.edu (Y.A.K.); atharvasurya99@g.ucla.edu (A.S.); fa2@g.ucla.edu (F.A.)
[2] Department of Electrical and Computer Engineering, University of California, Los Angeles, CA 90095, USA; aishaaln2@gmail.com
* Correspondence: pdc@seas.ucla.edu; Tel.: +1-(310)-794-1015

Abstract: This work focuses on developing and applying Encrypted Lyapunov-based Model Predictive Control (LMPC) in a nonlinear chemical process network for Ethylbenzene production. The network, governed by a nonlinear dynamic model, comprises two continuously stirred tank reactors that are connected in series and is simulated using Aspen Plus Dynamics. For enhancing system cybersecurity, the Paillier cryptosystem is employed for encryption–decryption operations in the communication channels between the sensor–controller and controller–actuator, establishing a secure network infrastructure. Cryptosystems generally require integer inputs, necessitating a quantization parameter d, for quantization of real-valued signals. We utilize the quantization parameter to quantize process measurements and control inputs before encryption. Through closed-loop simulations under the encrypted LMPC scheme, where the LMPC uses a first-principles nonlinear dynamical model, we examine the effect of the quantization parameter on the performance of the controller and the overall encryption to control the input calculation time. We illustrate that the impact of quantization can outweigh those of plant/model mismatch, showcasing this phenomenon through the implementation of a first-principles-based LMPC on an Aspen Plus Dynamics process model. Based on the findings, we propose a strategy to mitigate the quantization effect on controller performance while maintaining a manageable computational burden on the control input calculation time.

Keywords: model predictive control; cybersecurity; encrypted control; semi-homomorphic encryption; quantization; process control

Citation: Kadakia, Y.A.; Suryavanshi, A.; Alnajdi, A.; Abdullah, F.; Christofides, P.D. Encrypted Model Predictive Control of a Nonlinear Chemical Process Network. *Processes* **2023**, *11*, 2501. https://doi.org/10.3390/pr11082501

Academic Editors: Jie Zhang and Stefania Tronci

Received: 14 July 2023
Revised: 10 August 2023
Accepted: 18 August 2023
Published: 20 August 2023

1. Introduction

With the rapid advancement of technology and the increasing integration of devices, networked cyber–physical systems, particularly those utilizing SCADA (Supervisory Control and Data Acquisition) technology, have become integral components of critical infrastructure across industries, such as energy, water, transportation, and manufacturing. These systems enable efficient monitoring, control, and automation of complex processes, enhancing productivity and operational efficiency. However, the increased connectivity and integration of SCADA systems with corporate networks and the Internet have exposed them to potential cyber threats. A breach or compromise in these systems can have severe consequences, including the disruption of essential services, physical damage, financial losses, and even threats to public safety. Recent advances in cyberattack techniques and the growing sophistication of threat actors have further highlighted the criticality of implementing robust cybersecurity measures.

Various real-world examples underscore the importance of cybersecurity in networked cyber–physical systems and SCADA environments. For instance, the Stuxnet worm, discovered in 2010, specifically targeted SCADA systems in Iranian nuclear facilities. Stuxnet infiltrated Iranian PLCs (Programmable Logic Controllers), gathering data about industrial systems and causing the fast-spinning centrifuges to burn out [1]. Another notable incident

is the Ukrainian power grid cyberattack in 2015, where hackers successfully compromised SCADA systems, leading to widespread power outages affecting thousands of people. In a recent incident in May 2021, the Colonial Pipeline, a major fuel pipeline operator in the United States, fell victim to a ransomware attack. The attackers infiltrated Colonial Pipeline's network through the DarkSide ransomware. They encrypted the company's systems and demanded a ransom payment in exchange for the decryption keys. As a result, Colonial Pipeline shut down its operations, leading to disruptions in fuel supply and causing a significant economic impact.

Despite significant advancements in addressing cybersecurity challenges within the information technology (IT) domain, the operational technology (OT) domain is still catching up in terms of progress. IT primarily focuses on the software component of systems, encompassing network infrastructure and data management. In contrast, OT ensures the smooth operation of critical infrastructure, including power grids, smart meters, and distribution systems. Notably, cyberattacks targeting OT systems tend to have more severe and far-reaching consequences compared to those in IT. These attacks can lead to outcomes such as shutdowns, outages, leakages, and even explosions. Consequently, standards development organizations like the National Institute of Standards and Technology (NIST) [2] have devised essential cybersecurity research roadmaps. These roadmaps serve as frameworks designed to identify and mitigate the impact of cyberattacks, thereby exerting a notable influence on the security protocols adopted across various industries.

While significant research efforts continue to focus on diverse domains, such as the creation of machine learning-based cyberattack detectors [3–6], the design of backup controllers in a two-tier safety-performance control architecture [7], the recovery of process states following a cyberattack [8], and the development of cyberattack-resilient controllers [9,10], one critical and fundamental research issue remains unresolved: the establishment of universally implementable secure data transmission lines in any cyber–physical networked system, without requiring controller modifications, the installation of backup control systems, the development of system-specific detection mechanisms, or tailor-made solutions for individual platforms. A promising solution to address this issue is utilizing an encrypted control system. This approach offers a versatile and effective solution for enhancing data security and confidentiality. It can be easily implemented across various systems without necessitating system-specific modifications, thereby addressing the fundamental challenge of secure data transmission in networked systems.

Regarding encrypted control, extensive research has been conducted in the field of linear control systems, with control computations performed in a fully encrypted space. The fundamental concept behind such systems is multiplicative homomorphism, which enables multiplication operations to be executed in an encrypted medium using complex cryptosystems, like ElGamal [11]. However, such operations in an encrypted space can be computationally demanding and not applicable to systems governed by complicated nonlinear dynamics where nonlinear controllers may be needed, limiting their widespread adoption. Alternatively, a more viable approach could involve using encryption to secure data transmission lines. The data collected by the sensors can be encrypted, subsequently transferred, and decrypted at the controller, which can be isolated and fortified against potential security breaches. Therefore, within the context of this research, we consider that the edge computer, responsible for executing controller computations within a SCADA architecture, operates within a completely secure cyber–physical setting due to encryption of the sensor-to-controller and controller-to-actuator signals. Specifically, in our formulation, the controller can compute the control action in plaintext, eliminating the need for convoluted calculations in an encrypted space. Subsequently, the control action can undergo encryption before transmission to the actuator, where the encrypted control action is decrypted and executed. This method avoids computationally demanding operations in an encrypted space and is effectively implementable in systems employing advanced process control schemes for nonlinear systems, such as Model Predictive Control (MPC).

Since its inception, the chemical industry has extensively adopted Model Predictive Control (MPC) due to its effectiveness in achieving closed-loop stability and optimizing key performance metrics and its capability to handle multiple inputs and outputs and accommodate constraints on system states and inputs. These benefits arise from employing a mathematical model of the system to predict future behavior and optimize control inputs accordingly. However, implementing MPC necessitates decryption at the controller to obtain the essential information required for prediction and optimization. In an industrial setting, an edge computer, accessible remotely by the sensors and actuators through the network, can perform nonlinear MPC computations. The objective is to utilize encryption techniques to establish secure connections between the sensors–edge computer and edge computer–actuators. The referenced work [12] provides a comprehensive exploration of the design of an encrypted Model Predictive Control framework, as well as the influence of quantization on system performance. Building upon that foundation, in this work, we go a step further by implementing the encrypted Lyapunov-based Model Predictive Control (LMPC) scheme in a large-scale chemical process network used for Ethylbenzene production, using an Aspen Plus Dynamics-based process model in conjunction with a first-principles-based LMPC to showcase that the influence of quantization can surpass the impact of plant/model mismatch. Moreover, this study conducts a comprehensive and innovative investigation to assess how encryption–decryption affects the computation time required for computing the control action. By thoroughly examining the impact of the quantization parameter selected for encryption on the computation time, this research aims to provide new perspectives and deeper insights into the practical implications of data encryption. To our knowledge, prior investigations have not explored the implications of an encrypted MPC scheme in the aforementioned domains.

To apply the encrypted LMPC, we develop two distinct nonlinear dynamical models: one utilizing Aspen Plus Dynamics V12 and the other based on first-principles modeling fundamentals. In Section 4, we conduct closed-loop simulations for the Aspen Plus Dynamics model, employing the first-principles model-based encrypted LMPC for various quantization parameters. Further, we investigate the impact of these parameters on controller performance and put forth a proposal to mitigate quantization errors and their effects on controller performance. Additionally, in Section 5, we explore the influence of encryption–decryption on the total control input calculation time. Expanding on the previous recommendation, we provide clear guidance on implementing the encrypted LMPC approach. This implementation ensures a feasible computation time for control action computation (with encryption) while establishing secure communication pathways between the sensor–controller and controller–actuator components, without compromising the performance of the controller.

2. Preliminaries

2.1. Notation

The Euclidean norm of a vector is denoted by the symbol $\|\cdot\|$. The notation x^T represents the transpose of the vector x. The standard Lie derivative $L_f V(x)$ is defined as the partial derivative of the function $V(x)$ with respect to x multiplied by the vector field $f(x)$, $L_f V(x) := \frac{\partial V(x)}{\partial x} f(x)$. The sets $\mathbb{R}$, $\mathbb{Z}$, and $\mathbb{N}$ refer to the sets of real numbers, integers, and natural numbers, respectively. Additionally, $\mathbb{Z}_M$ and $\mathbb{Z}_M^*$ represent the additive and multiplicative groups of integers modulo M, respectively.

The set subtraction operation is denoted by "\", meaning that $A \backslash B$ represents the set of elements in A that are not in B. A function $f(\cdot)$ is said to be of class $\mathcal{C}^1$ if it is continuously differentiable in its domain. A continuous function $\alpha : [0, a) \to [0, \infty)$ is considered to be in the class $\mathcal{K}$ if it is strictly increasing and only evaluates to zero at zero. The function $\gcd(i, j)$ denotes the greatest common divisor, which returns the largest positive integer that divides both i and j without leaving a remainder. On the other hand, $\text{lcm}(i, j)$ represents the least common multiple of the integers i and j.

2.2. Class of Systems

In this work, we primarily focus on a specific category of systems known as nonlinear continuous-time systems with multiple inputs and multiple outputs (MIMO). These systems represent a set of first-order ordinary differential equations (ODEs) that exhibit nonlinear behavior. The general representation of these systems is

$$\dot{x} = F(x, u) = f(x) + g(x)u \tag{1}$$

The system is described by a state vector $x = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^n$ and a control input vector $u \in \mathbb{R}^m$. The inputs applied to the system are subject to certain bounds, defined by the set $U \subset \mathbb{R}^m$, where $U := u \in U | u_{\min,i} \leq u_i \leq u_{\max,i}, \forall; i = 1, 2, \cdots, m$. The values $u_{\min,i}$ and $u_{\max,i}$ represent the minimum and maximum limits for each manipulated input, respectively. The functions $f(\cdot)$ and $g(\cdot)$ are assumed to be sufficiently smooth vector and matrix functions, respectively. For the sake of simplicity and without sacrificing the general applicability, we make the assumption that $f(0) = 0$, thereby considering the origin as a steady state of the nonlinear system described by Equation (1). For convenience, we set the initial time to zero throughout the paper ($t_0 = 0$). In addition, we introduce some notation: the space of continuous functions that map the interval $[a, b]$ to $\mathbb{R}^n$ is denoted by $C([a, b], \mathbb{R}^n)$. We also define the set $S(\Delta)$ as the collection of piece-wise constant functions with a period of Δ.

2.3. Achieving Stability through Lyapunov-Based Feedback Control

We assume the existence of a feedback controller denoted as $u = \Phi(x) \in U$ to achieve exponential stability at the origin within the system described by Equation (1). This exponential stability is characterized by the presence of a continuously differentiable control Lyapunov function denoted as $V(x)$, satisfying the following inequalities for all x within an open neighborhood D around the origin [13,14]:

$$c_1|x|^2 \leq V(x) \leq c_2|x|^2, \tag{2a}$$

$$\frac{\partial V(x)}{\partial x} F(x, \Phi(x)) \leq -c_3|x|^2, \tag{2b}$$

$$\left| \frac{\partial V(x)}{\partial x} \right| \leq c_4|x| \tag{2c}$$

where c_1, c_2, c_3, and c_4 are positive constants. The method presented in the referenced work [15] offers an approach to construct a stabilizing controller that satisfies the desired criteria. For the nonlinear system of Equation (1), the closed-loop stability region is characterized as a level set of the Lyapunov function V. This stability region Ω_ρ is defined as $\Omega_\rho := \{x \in D | V(x) \leq \rho\}$, where $\rho > 0$.

2.4. Paillier Cryptosystem

In this research article, we utilize the Paillier cryptosystem [16] to apply encryption and decryption to process measurements (represented as x) and control inputs (represented as u). The Paillier cryptosystem is a partially homomorphic encryption scheme that enables performing addition operations within the encrypted message space. However, the primary rationale for utilizing the Paillier cryptosystem in this paper is its computational efficiency compared to other cryptosystems, such as ElGamal or AES, rather than its partial homomorphic property. Like most cryptosystems, the Paillier cryptosystem operates by encrypting plaintext data presented in the form of non-negative integers. The encryption process commences with the generation of public and private keys. The public key is used to encrypt integer messages and produce ciphertexts. Conversely, the private key decrypts the ciphertexts and recovers the original integer messages. The generation of the public and private keys in the Paillier cryptosystem follows a specific set of steps:

1. Select two large random prime integers (p and q) satisfying the condition $\gcd(pq, (p-1)(q-1)) = 1$.
2. Calculate the product of these integers, denoted by $M = pq$.
3. Select a random integer g such that $g \in \mathbb{Z}^*_{M^2}$, where $\mathbb{Z}^*_{M^2}$ is the multiplicative group of integers modulo M^2.
4. Calculate $\lambda = \text{lcm}(q-1, p-1)$.
5. Define $L(x) = (x-1)/M$.
6. Check the existence of the following modular multiplicative inverse: $u = (L(g^\lambda \bmod M^2))^{-1} \bmod M$.
7. If the inverse does not exist, return to step 3 and select an alternative value for g. In the event that the inverse does exist, we obtain the public key (M, g) and the private key (λ, u).

After obtaining the keys, we distribute the public key to the intended recipients that perform the encryption process. Similarly, we share the private key exclusively with the authorized recipients responsible for decrypting the data. The process of encryption–decryption consists of the following steps:

$$E_M(m, r) = c = g^m r^M \bmod M^2 \tag{3}$$

where $r \in \mathbb{Z}_M$ is a random integer and c is the ciphertext obtained after encryption of m. The decryption process of the ciphertext $c \in \mathbb{Z}_{M^2}$, is performed as follows:

$$D_M(c) = m = L(c^\lambda \bmod M^2)u \bmod M \tag{4}$$

2.5. Quantization

In order to utilize the Paillier cryptosystem, it is necessary to represent the input data to be encrypted as natural numbers. However, it is important to note that the signal measurements provided before encryption are typically in the form of floating-point numbers. Consequently, a mapping procedure becomes essential to convert these floating-point numbers into elements within the set $\mathbb{Z}_M$. This procedure involves quantization, where a quantization parameter denoted by d is chosen to perform the quantization operations [17].

To achieve this objective, we adopt signed fixed-point numbers in binary representation. The quantization parameters l_1 and d refer to the total number of bits and the number of fractional bits, respectively. Using these quantization parameters, we construct a set denoted as $\mathbb{Q}_{l_1,d}$. This set encompasses rational numbers ranging from -2^{l_1-d-1} to $2^{l_1-d-1} - 2^{-d}$, with each rational number separated by a resolution of 2^{-d}. A rational number q belonging to the set $\mathbb{Q}_{l_1,d}$ can be expressed as follows: $q \in \mathbb{Q}_{l_1,d}$, where $\exists \beta \in \{0,1\}^{l_1}$ and $q = -2^{l_1-d-1}\beta_{l_1} + \sum_{i=1}^{l_1-1} 2^{i-d-1}\beta_i$. To map a real number data point a to the set $\mathbb{Q}_{l_1,d}$, we employ the function $g_{l_1,d}$ given by the following equation:

$$\begin{aligned} g_{l_1,d} &: \mathbb{R} \to \mathbb{Q}_{l_1,d} \\ g_{l_1,d}(a) &:= \arg\min_{q \in \mathbb{Q}_{l_1,d}} |a - q| \end{aligned} \tag{5}$$

This function allows us to determine the closest quantized rational number to a given real number data point. Following this quantization step, the quantized data are mapped to a set of integers using bijective mapping denoted as $f_{l_2,d}$ [17]. This mapping ensures that the quantized data are transformed into a subset of the message space $\mathbb{Z}_M$. The bijective mapping can be defined as

$$\begin{aligned} f_{l_2,d} &: \mathbb{Q}_{l_1,d} \to \mathbb{Z}_{2^{l_2}} \\ f_{l_2,d}(q) &:= 2^d q \bmod 2^{l_2} \end{aligned} \tag{6}$$

The encryption process involves encrypting integer plaintext messages using the set $Z_{2^{l_2}}$, and the resulting ciphertexts can be decrypted back into the same set $Z_{2^{l_2}}$. Once the controller and actuator receive the encrypted signals, the ciphertexts undergo decryption to

extract integer plaintext messages that represent quantized states and inputs, respectively. Consequently, it becomes essential to remap these decrypted plaintext messages back to the set $\mathbb{Q}_{l_1,d}$. The inverse mapping, denoted as $f^{-1}_{l_2,d}$, is defined as follows:

$$f^{-1}_{l_2,d} : \mathbb{Z}_{2^{l_2}} \to \mathbb{Q}_{l_1,d} \tag{7}$$

$$f^{-1}_{l_2,d}(m) := \frac{1}{2^d} \begin{cases} m - 2^{l_2} & \text{if } m \geq 2^{l_2-1} \\ m & \text{otherwise} \end{cases} \tag{8}$$

To demonstrate encryption and decryption, we can refer to Figure 1. For this example, the chosen quantization parameter, total number of bits, and bijective mapping parameter are $d = 3$, $l_1 = 18$, and $l_2 = 30$. Let us consider the rational number $a = -1.31752$ which is the input data to be encrypted to illustrate the encryption–decryption process and the effect of quantization.

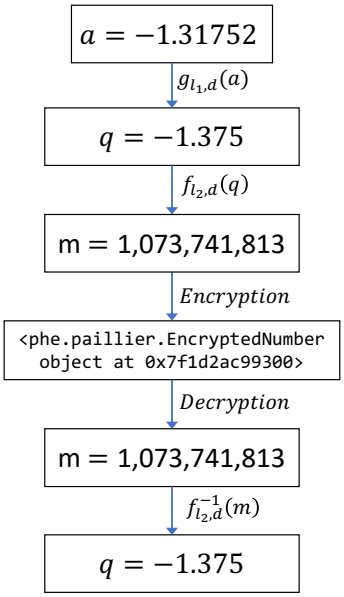

Figure 1. Illustration of encryption–decryption applied to a floating-point real number.

3. Design of the Encrypted MPC

In the envisioned closed-loop architecture of the encrypted MPC, as depicted in Figure 2, the sensor signals $x(t)$ are subjected to encryption before being sent to the model predictive controller (MPC). After obtaining the encrypted data, it undergoes decryption, resulting in quantized states $\hat{x}(t)$. These quantized states serve as the initial values for the plant model within the MPC at time t. The MPC subsequently computes optimized inputs $u(t)$, which are encrypted prior to transmission to the actuator. After the actuator receives the encrypted signals as input, the encrypted input is decrypted, leading to a quantized input $\hat{u}(t)$ that is applied to the process.

The above closed-loop design introduces two sources of errors. Firstly, a quantization error in the sensor-MPC communication link, resulting from the mapping of the state data from $\mathbb{R}$ to $\mathbb{Q}_{l_1,d}$. Additionally, the MPC-actuator communication link introduces an input quantization error caused by the conversion of input data from the set of real numbers $\mathbb{R}$ to $\mathbb{Q}_{l_1,d}$. These quantization errors are bounded and can be characterized by the mapping equation of Equation (5), ensuring that

$$|x(t) - \hat{x}(t)| \leq 2^{-d-1} \tag{9a}$$

$$|u(t) - \hat{u}(t)| \leq 2^{-d-1} \tag{9b}$$

where d is the quantization parameter used for mapping in Equation (5). Firstly, taking into account the impact of quantization-induced input errors, the dynamical model of the MPC employs a nonlinear system, represented by Equation (1), which can be expressed as follows:

$$\begin{aligned} \dot{x} = F(x, \hat{u}) &= f(x) + g(x)\hat{u} \\ &= f(x) + g(x)(u + e) \end{aligned} \tag{10}$$

where $e = \hat{u}(t) - u(t)$ and

$$|e| \leq 2^{-d-1} \tag{11}$$

Secondly, an error in the control input, $u = \Phi(x) \in U$, will emanate as the MPC receives $\hat{x}$ instead of the actual state x. This error will be bounded by the underlying equation, where $L_1 > 0$:

$$|\Phi(\hat{x}) - \Phi(x)| \leq L_1|\hat{x} - x| \leq L_1 2^{-d-1} \tag{12}$$

Reference [12] discusses and establishes the stability of the proposed control loop with encrypted data transfer, providing assurance for the closed-loop system stability even in the presence of encryption, under certain conditions.

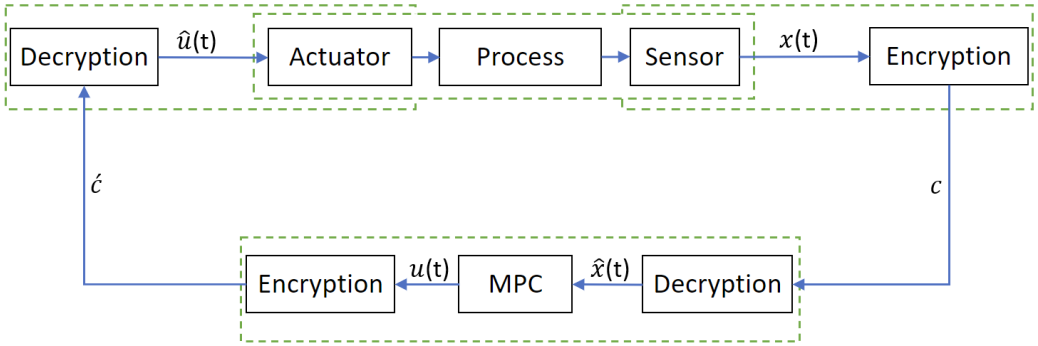

Figure 2. Illustration of the data transfer process in an encrypted MPC system.

Remark 1. *The error in the quantization operation occurs when the target value to be quantized is not found exactly in the set $\mathbb{Q}_{l_1,d}$, which consists of quantized values with a certain resolution determined by the quantization parameter, denoted as d. The resolution between elements in this set is given by 2^{-d}. To determine the upper bound of the error, let us focus on a specific value, denoted as x_1, that needs to be quantized. We assume that x_1 falls within the range of y_1 and $y_1 + 2^{-d}$, where y_1 and $y_1 + 2^{-d}$ represent quantized values in the set $\mathbb{Q}_{l_1,d}$. The quantization process involves comparing the distance between x_1 and y_1 with the distance between x_1 and $y_1 + 2^{-d}$. If the distance between x_1 and y_1 is smaller than the distance between x_1 and $y_1 + 2^{-d}$, then x_1 is mapped to y_1. Otherwise, it is mapped to $y_1 + 2^{-d}$. The error in quantization is then bounded by half the resolution, which is equal to $|y_1 + 2^{-d} - y_1|/2 = 2^{-d-1}$. This implies that the maximum difference between the quantized value $\hat{x}_1$, and the actual value x_1, is 2^{-d-1}.*

Encrypted Lyapunov-Based MPC

This section presents a formulation of the feedback MPC for the closed-loop design of the nonlinear system described by Equation (1), considering secure sensor–controller and controller–actuator communication links. Control actions will be applied to the nonlinear system using a sample-and-hold approach with a sampling period of Δ [18,19]. The proposed MPC formulation is outlined as follows:

$$\mathcal{J} = \min_{u \in S(\Delta)} \int_{t_k}^{t_{k+N}} L_2(\tilde{x}(t), u(t))dt \tag{13a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F(\tilde{x}(t), u(t)) = f(\tilde{x}) + g(\tilde{x})u \tag{13b}$$

$$u(t) \in U, \ \forall \ t \in [t_k, t_{k+N}) \tag{13c}$$

$$\tilde{x}(t_k) = \hat{x}(t_k) \tag{13d}$$

$$\dot{V}(\hat{x}(t_k), u) \leq \dot{V}(\hat{x}(t_k), \Phi(\hat{x}(t_k))), \text{ if } \hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{\min}} \tag{13e}$$

$$V(\tilde{x}(t)) \leq \rho_{\min}, \ \forall \ t \in [t_k, t_{k+N}), \text{ if } \hat{x}(t_k) \in \Omega_{\rho_{\min}} \tag{13f}$$

Within the framework of the Lyapunov-based MPC, referred to as LMPC, the predicted state trajectory is represented as $\tilde{x}$, the sampling time is denoted by Δ, and the prediction horizon encompasses a number of sampling periods indicated by N. The LMPC algorithm computes the optimal input sequence $u^*(t|t_k)$ for the entire prediction horizon $t \in [t_k, t_{k+N})$. The first input of this sequence is subsequently transmitted to the actuator for application to the system within the interval $t \in [t_k, t_{k+1})$.

In the encrypted LMPC design, the MPC uses quantized states $\hat{x}$ for predicting the state trajectory, Equation (13a) integrates the cost function over the entire prediction horizon, and it computes the optimized control inputs for the entire prediction horizon. However, the actuator only applies the control inputs corresponding to the first prediction horizon and repeats this process at each sampling instance. Equation (13b) represents the dynamic system model used by the LMPC. Equation (13c) represents the constraints imposed on the control inputs. The constraint in Equation (13d) initializes the plant model described in Equation (13b) with quantized states. If the state $x(t_k)$ at time t_k lies within the set $\Omega_{\hat{\rho}} \setminus \Omega_{\rho_{\min}}$, where $\rho_{\min}$ represents a level set of V in proximity to the origin, the Lyapunov constraint outlined in Equation (13e) steers the closed-loop state $x(t_k)$ of the nonlinear system presented in Equation (10) toward the origin. Once the closed-loop state $x(t_k)$ enters the region $\Omega_{\rho_{\min}}$, the constraint specified in Equation (13f) ensures that this state remains within $\Omega_{\rho_{\min}}$ throughout the entire prediction horizon.

4. Application to a Chemical Process Operating at an Unstable Steady State Using Aspen Plus Simulator

In this section, we demonstrate the application of the proposed encrypted LMPC to a large-scale chemical process. To begin, we construct two dynamic models for a chemical process. We develop the first dynamic model using Aspen Plus Dynamics V12, while the second model is based on first-principles modeling fundamentals. Aspen Plus Dynamics is a high-fidelity software that can be used for the detailed dynamic simulation of chemical processes in an operating region around a stable or unstable steady state, which is not possible in steady-state simulation software for chemical processes, and hence, can be considered as the closest representation of the actual process dynamic behavior. Furthermore, first-principles-based MPC computations can be performed on a computer in SCADA systems using Python. As a first-principles model can be derived for most processes even in the absence of data and can be simulated readily with available solvers, the Aspen Plus Dynamics model and first-principles-based Python code can be considered as a "standard metric" to quantify and analyze specific aspects of MPC. In this work, we use a distinct model to simulate the chemical process from the model incorporated into the LMPC to demonstrate the impact of quantization and compare it with the plant/model mismatch. We design both models without any input or state delays. Subsequently, closed-loop simulations are performed in the Aspen Plus Dynamics model using the first-principles model-based LMPC. Finally, we replace the LMPC with an encrypted LMPC, and closed-loop simulations are conducted and discussed.

4.1. Process Description

The process considered is the production of Ethylbenzene (EB) from Ethylene (E) and Benzene (B) as reactive raw materials. The main reaction, labeled as "primary", is a

second-order, exothermic, and irreversible reaction that occurs alongside two additional side reactions. This reaction scheme is illustrated in Equation (14) and takes place in two non-isothermal, well-mixed continuous stirred tank reactors (CSTR). The chemical reactions involved are as follows:

$$C_2H_4 + C_6H_6 \longrightarrow C_8H_{10} \quad \text{(primary)} \tag{14a}$$

$$C_2H_4 + C_8H_{10} \longrightarrow C_{10}H_{14} \tag{14b}$$

$$C_6H_6 + C_{10}H_{14} \longrightarrow 2\,C_8H_{10} \tag{14c}$$

The state variables are the concentrations of Ethylene, Benzene, Ethylbenzene and Di-Ethylbenzene and the reactor temperature, for each CSTR$_i$, $i = (1, 2)$, respectively, which in deviation terms is:
$x^T = [C_{E_1} - C_{E_{1s}},\ C_{B_1} - C_{B_{1s}},\ C_{EB_1} - C_{EB_{1s}},\ C_{DEB_1} - C_{DEB_{1s}},\ T_1 - T_{1s},\ C_{E_2} - C_{E_{2s}},\ C_{B_2} - C_{B_{2s}},\ C_{EB_2} - C_{EB_{2s}},\ C_{DEB_2} - C_{DEB_{2s}},\ T_2 - T_{2s}]$

The subscript "s" denotes the steady-state value. The rate of heat removal for each reactor $[Q_1 - Q_{1s},\ Q_2 - Q_{2s}]$ is the manipulated inputs to our process, which are bounded by the closed sets $[-10^4$ kW, 2×10^3 kW$]$ and $[-1.5 \times 10^4$ kW, 5×10^3 kW$]$, respectively.

The control objective is to maintain the operation of both the CSTRs at their unstable steady state under the encrypted LMPC using the quantized states and inputs in computation and actuation. Because the rate of heat removal for each CSTR is the manipulated input, the reactor temperature state variables are directly affected by it. However, the manipulated inputs do not directly influence the concentration states. Instead, they follow open-loop trajectories, gradually converging to their respective steady-state values as the reactor temperatures approach their steady-state values.

To identify the stability condition of the operating steady state, we conducted an open-loop simulation in Aspen Plus Dynamics. We initiated the simulation using the steady-state values as initial conditions, and the control inputs were held constant at their respective steady-state values (0 in deviation form) throughout the simulation. After running the simulation for 10 h of process time, the system states converged to a distinct stable steady state, providing clear evidence that the selected operating condition is an unstable steady state. The main reason behind choosing this unstable steady state as the operating condition was its ability to yield the highest amount of Ethylbenzene, our desired product, at steady state, at the outlet of the second CSTR.

4.2. Dynamic Model in Aspen Plus Dynamics

We develop the process model for this system using Aspen Plus and Aspen Plus Dynamics V12. These are high-fidelity simulators used for complex chemical process modeling. The two CSTRs are connected in series, such that the output of the first reactor affects the second reactor but not vice versa. Initially, the process model is created in Aspen Plus, and a steady-state simulation is performed and validated by examining material and energy balances. Subsequently, dynamic simulations of the process are conducted in Aspen Plus Dynamics, enabling a thorough analysis and control of its dynamic behavior. The construction of both the steady-state and dynamic models follows the following procedure in detail:

1. Inlet stream configuration: We enter the inlet stream components, concentrations, and temperatures into Aspen Plus and supply it to each reactor through Hexane solutions with flow rates F_1 and F_2. Using Hexane ensures the inlet flows remain in the liquid phase at the feeding temperature. C_E, C_B, C_{EB}, and C_{DEB} represent the concentrations of Ethylene, Benzene, Ethylbenzene, and Di-Ethylbenzene in the inlet stream, respectively. T_i, ρ_i, V_i, are the temperature, liquid density, and volume of CSTR$_i$, $i = 1, 2$. C_P represents the mass-specific heat capacity of the liquid mixture and is assumed to remain constant throughout the process in both reactors. Table 1 specifies the process parameters used. The subscript "o" denotes the state in the inlet stream, and "s" indicates the steady-state conditions.

2. Pressure drop selection: Valves play a crucial role in establishing a dynamic model for Aspen Plus Dynamics, as they serve as connectors between components and regulate fluid flow by controlling the pressure drop across the system. A suitable pressure drop specifies the flow direction, ensuring a smooth simulation run. In our model, valves v_1, v_2, v_3, and v_4 are assigned pressure drops of 5, 5, 2, and 14 bars, respectively.
3. Reaction and reactor specification: We define the reaction parameters and stoichiometry in Aspen Plus. All reactions mentioned in Equation (14) are selected in the kinetic specifications of both the CSTRs. We set the initial pressure of each CSTR to 15 bar and equip them with a heating/cooling jacket to provide or remove heat at a rate denoted by Q_i, where i represents the reactor number. The initial temperatures of the first and second CSTRs are 350 K and 400 K, respectively. These settings ensure that the reactants and products remain in the liquid phase throughout the process. After completing the reaction specification for both reactors, we carry out a steady-state simulation.
4. Reactor geometry: Before exporting the steady-state model from Aspen Plus to Aspen Plus Dynamics, it is necessary to define the reactor geometry. In our model, the vessels are of the vertical type with flat heads, and each CSTR has a length of ten meters.
5. Pressure verification: To ensure the accuracy of the dynamic model, perform a pressure check using the integrated Aspen Plus pressure checker. This step verifies that no errors arise during the dynamic process. Once the steady-state model successfully passes the pressure check, we export it to Aspen Plus Dynamics for further analysis and simulation.
6. Dynamic model initialization: Level controllers are added to each reactor to maintain them at the desired capacity. We perform a steady-state simulation to determine the steady-state values of the dynamic model. The values obtained are listed in Equation (14). Further, we specify the initial values of the states in both reactors for the dynamic simulation. Through an initialization run, we ensure the values entered are thermo-kinetically consistent with the model specifications.
7. Manipulated input configuration: For external control of the manipulated variables Q_1 and Q_2 (heat duty of reactor 1 and 2, respectively) during the dynamic simulation, the heating type of the reactors is switched to constant heat duty. With these adjustments, the dynamical process model is now fully established. Figure 3 depicts the corresponding model flow sheet.

Table 1. Parameter values, steady-state values, and model configuration of the Aspen model.

$T_{1o} = T_{2o} = 350\,\mathrm{K}$	$T_{1s} = 321.15\,\mathrm{K}$
$V_1 = V_2 = 60\,\mathrm{m}^3$	$T_{2s} = 442.99\,\mathrm{K}$
$F_1 = 43.2\,\mathrm{m}^3/\mathrm{h}$	$F_2 = 47.87\,\mathrm{m}^3/\mathrm{h}$
$C_{E_{o1}} = 4.43\,\mathrm{kmol}/\mathrm{m}^3$	$C_{E_{1s}} = 4.33\,\mathrm{kmol}/\mathrm{m}^3$
$C_{B_{o1}} = 5.54\,\mathrm{kmol}/\mathrm{m}^3$	$C_{B_{1s}} = 5.55\,\mathrm{kmol}/\mathrm{m}^3$
$C_{E_{o2}} = 4.02\,\mathrm{kmol}/\mathrm{m}^3$	$C_{E_{2s}} = 0.196\,\mathrm{kmol}/\mathrm{m}^3$
$C_{B_{o2}} = 5.02\,\mathrm{kmol}/\mathrm{m}^3$	$C_{B_{2s}} = 1.31\,\mathrm{kmol}/\mathrm{m}^3$
$C_{EB_{1s}} = 0.53\,\mathrm{kmol}/\mathrm{m}^3$	$C_{EB_{2s}} = 4.22\,\mathrm{kmol}/\mathrm{m}^3$
$C_{DEB_{1s}} = 8.76 \times 10^{-4}\,\mathrm{kmol}/\mathrm{m}^3$	$C_{DEB_{2s}} = 0.0078\,\mathrm{kmol}/\mathrm{m}^3$
$k_1 = 1.528 \times 10^6\,\mathrm{m}^3\,\mathrm{kmol}^{-1}\,\mathrm{s}^{-1}$	$E_1 = 71{,}160\,\mathrm{kJ}/\mathrm{kmol}$
$k_2 = 2.778 \times 10^4\,\mathrm{m}^3\,\mathrm{kmol}^{-1}\,\mathrm{s}^{-1}$	$E_2 = 83{,}680\,\mathrm{kJ}/\mathrm{kmol}$
$k_3 = 0.4167\,\mathrm{m}^3\,\mathrm{kmol}^{-1}\,\mathrm{s}^{-1}$	$E_3 = 62{,}760\,\mathrm{kJ}/\mathrm{kmol}$
$\rho_1 = 639.153\,\mathrm{kg}/\mathrm{m}^3$	$\rho_2 = 607.504\,\mathrm{kg}/\mathrm{m}^3$
$\Delta H_1 = -1.04 \times 10^5\,\mathrm{kJ}/\mathrm{kmol}$	$\Delta H_2 = -1.02 \times 10^5\,\mathrm{kJ}/\mathrm{kmol}$
$\Delta H_3 = -5.5 \times 10^2\,\mathrm{kJ}/\mathrm{kmol}$	$C_p = 2.411\,\mathrm{kJ}\,\mathrm{kg}^{-1}\,\mathrm{K}^{-1}$
$Q_{1s} = -1074.63\,\mathrm{kW}$	$Q_{2s} = -6768.83\,\mathrm{kW}$
$C_p = 2.411\,\mathrm{kJ}\,\mathrm{kg}^{-1}\,\mathrm{K}^{-1}$	$R = 8.314\,\mathrm{kJ}\,\mathrm{kmol}^{-1}\,\mathrm{K}^{-1}$
Heat transfer option	Dynamics
Temperature approach	77.33 K
Heat capacity of coolant	$4.2\,\mathrm{kJ}\,\mathrm{kg}^{-1}\,\mathrm{K}^{-1}$
Medium holdup	1000 kg

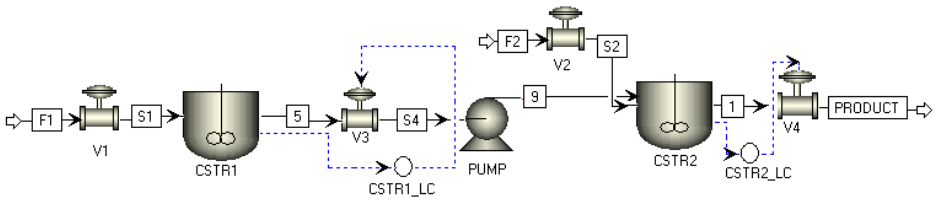

Figure 3. Aspen Plus Dynamics model flow sheet.

4.3. First-Principles Model Development

By applying the concepts of mass and energy balances, the first-principles model for the CSTRs is developed. Specifically, the dynamic model of the first CSTR is represented by the following ODEs:

$$\frac{dC_{E_1}}{dt} = \frac{(F_1 C_{Eo_1} - F_{out_1} C_{E_1})}{V_1} - r_{1,1} - r_{1,2} \tag{15a}$$

$$\frac{dC_{B_1}}{dt} = \frac{(F_1 C_{Bo_1} - F_{out_1} C_{B_1})}{V_1} - r_{1,1} - r_{1,3} \tag{15b}$$

$$\frac{dC_{EB_1}}{dt} = \frac{-F_{out_1} C_{EB_1}}{V_1} + r_{1,1} - r_{1,2} + 2r_{1,3} \tag{15c}$$

$$\frac{dC_{DEB_1}}{dt} = \frac{-F_{out_1} C_{DEB_1}}{V_1} + r_{1,2} - r_{1,3} \tag{15d}$$

$$\frac{dT_1}{dt} = \frac{(T_{1_o} F_1 - T_1 F_{out_1})}{V_1} + \sum_{j=1}^{3} \frac{-\Delta H_j}{\rho_1 C_p} r_{1,j} + \frac{Q_1}{\rho_1 C_p V_1} \tag{15e}$$

where $F_{out_1} = F_1$. The dynamic model of the second CSTR is represented by the following ODEs:

$$\frac{dC_{E_2}}{dt} = \frac{(F_2 C_{Eo_2} + F_{out_1} C_{E_1} - F_{out_2} C_{E_2})}{V_2} - r_{2,1} - r_{2,2} \tag{16a}$$

$$\frac{dC_{B_2}}{dt} = \frac{(F_2 C_{Bo_2} + F_{out_1} C_{B_1} - F_{out_2} C_{B_2})}{V_2} - r_{2,1} - r_{2,3} \tag{16b}$$

$$\frac{dC_{EB_2}}{dt} = \frac{F_{out_1} C_{EB_1} - F_{out_2} C_{EB_2}}{V_2} + r_{2,1} - r_{2,2} + 2r_{2,3} \tag{16c}$$

$$\frac{dC_{DEB_2}}{dt} = \frac{F_{out_1} C_{DEB_1} - F_{out_2} C_{DEB_2}}{V_2} + r_{2,2} - r_{2,3} \tag{16d}$$

$$\frac{dT_2}{dt} = \frac{(T_{2_o} F_2 - T_1 F_{out_1} - T_2 F_{out_2})}{V_2} + \sum_{j=1}^{3} \frac{-\Delta H_j}{\rho_2 C_p} r_{2,j} + \frac{Q_2}{\rho_2 C_p V_2} \tag{16e}$$

where $F_{out_2} = F_1 + F_2$, and the reaction rates are calculated by the following expressions:

$$r_{i,1} = k_1 e^{\frac{-E_1}{RT_i}} C_{E_i} C_{B_i} \tag{17a}$$

$$r_{i,2} = k_2 e^{\frac{-E_2}{RT_i}} C_{E_i} C_{EB_i} \quad i = 1, 2 \text{ (reactor index)} \tag{17b}$$

$$r_{i,3} = k_3 e^{\frac{-E_3}{RT_i}} C_{DEB_i} C_{B_i} \tag{17c}$$

Remark 2. *When constructing a dynamic model based on first-principles fundamentals involving multiple ordinary differential equations (ODEs), there may be multiple potential steady states. It is crucial to design the dynamic model in a manner that ensures convergence to the desired steady state. It should be noted that the steady states obtained from the first-principles model may differ from those obtained using the Aspen model. Therefore, our approach involves expressing the first-principles dynamic model equations in the form* $\dot{x} = F(x, u) - F(x_s, u_s) = f(x) - f(x_s) + g(x)u - g(x_s)u_s$.

Here, x_s and u_s correspond to the steady-state values of the state variables and control inputs, respectively. These values are determined by the Aspen model through simulation. Writing the equations in this form guarantees that the first-principles model will converge to the desired steady states obtained from the Aspen model, particularly when dealing with multiple distinct steady states.

4.4. Linking the Dynamic Models

To establish a seamless data transfer between the Aspen model (Aspen Plus Dynamics V12) and the first-principles model-based LMPC (Python code), we program a script in Aspen Plus Dynamics. This script reads the calculated control inputs, exported as text files by the Python code responsible for computing the control inputs. Additionally, it facilitates the export of the state variable values from Aspen Plus Dynamics as text files read by the Python code. This data exchange occurs at each sampling time, establishing a robust data transfer link between the Aspen model and the first-principles-based LMPC.

Remark 3. *As discussed in Section 4.1, the MPC model (first-principles based) used for predicting future states and optimizing control inputs differs from the Aspen dynamic model, where we apply the controller. To address this model mismatch, we analyze the combined and relative effects of the quantization errors, which arise from encryption–decryption and can further amplify the model mismatch error. Our analysis reveals that the quantization error is bounded by half the resolution (resolution/2). For instance, when the quantization parameter chosen is $d = 1$, the resolution is 0.5, and the upper bound of the error between the actual and quantized values is resolution/2 or 0.25. Hence, for higher quantization parameters, the impact of the quantization error on the overall model mismatch error is negligible. It is important to note that quantization introduces a bounded error in the states, thereby limiting the extent of the model mismatch error.*

4.5. Implementing the Encrypted LMPC

Before implementing encryption–decryption in a process, it is crucial to carefully choose the values: d_1, l_1, and l_2. After closely examining the maximum and minimum permissible values of the states and inputs, we determine the number of integer bits, $l_1 - d_1$. The largest value in the set $\mathbb{Q}_{l_1,d}$ is obtained using the formula $2^{l_1-d_1-1} - 2^{-d_1}$, while the smallest value is $-2^{l_1-d_1-1}$. The quantization parameter d_1 should be selected based on factors, such as the desired accuracy and the range of the state and input values. Additionally, a value for l_2 should be selected such that l_2 is greater than l_1. These steps complete the hyperparameter selection process.

After following the aforementioned steps, we determine that, for the example discussed in this section, $l_1 - d$ is calculated to be 15. The values of l_1 and d need to be selected accordingly. In the set $\mathbb{Q}_{l_1,d}$, rational numbers are separated by a resolution of 2^{-d}, meaning that a higher value of d leads to lower quantization errors. For simulation purposes, we vary the values of d from 1 to 8, resulting in l_1 ranging from 16 to 23. It is important to ensure that $l_2 > l_1$ for the bijective mapping, so we choose $l_2 = 30$. After determining all the quantization-related parameters, we proceed to quantize the states and inputs. Subsequently, we encrypt them using the Paillier Encryption algorithm. The implementation of Paillier Encryption is carried out using the "phe" module in Python, specifically PythonPaillier [20]. The first-principles model, described by Equations (15) and (16), serves as the process model in the LMPC framework. To solve the optimization problem, we utilize the Python module of the IPOPT software, version: ASL (20190605) [21].

Remark 4. *IPOPT, Interior Point OPTimizer, is a software tool designed specifically for solving nonlinear optimization problems. It employs an iterative method known as the interior point method, which focuses on finding the optimal solution by gradually moving toward the interior of the feasible region. To solve the optimization problem, IPOPT employs a series of iterations. In each iteration, it updates a sequence of points that satisfy the given constraints and improve the value of the objective function. This process involves calculating descent directions based on the gradient and Hessian of both the objective function and the constraints. IPOPT considers both the feasibility and optimality of the solution, striving to find a point that not only satisfies the constraints but also optimizes the*

objective function. Throughout the iterations, IPOPT utilizes a barrier function to handle inequality constraints and a penalty function to handle equality constraints. It also incorporates a line search procedure to determine the appropriate step length and employs backtracking techniques to ensure convergence toward the optimal solution. In our study, the nature of the MPC formulation leads to a non-convex optimization problem. This signifies that the optimum achieved through the IPOPT optimizer is a local optimum, rather than a global one. The optimization process begins with a designated starting input trajectory based on predicted values for the extended horizon (beyond the first input trajectory calculation) from the prior iteration. Furthermore, the optimizer is guided by a prescribed tolerance error and an upper limit on the number of iterations. The optimizer will persist in its pursuit of an improved solution until either of these conditions is met. If the optimizer is unable to calculate an optimal solution, the computed solution from the backup controller (P-controller) will be substituted for that specific sampling instance.

To implement encryption in a practical setting, it is crucial to ensure that the sampling time Δ exceeds the combined maximum of the encryption–decryption time required for all the states and control inputs, as well as the maximum time needed for computing the control action at each sampling instance for all the considered quantization parameters, denoted as d. This requirement can be expressed by the following equation:

$$\begin{aligned} &\Delta > \max(\text{Enc-Dec time}) + \max(\text{MPC computation time}) \\ &\forall\, d = \{1, 2, 3, 4, 5, 6, 7, 8\} \end{aligned} \tag{18}$$

During the implementation of the encrypted MPC design in the SCADA systems, where encrypted sensor measurements and control actions are transmitted through the network, the time spent on signal transmission is generally not substantial due to the rapid and efficient nature of networked communication. However, this efficiency comes at the risk of susceptibility to cyberattacks. To mitigate this potential vulnerability, this study encrypts these communication channels and assesses the repercussions of encryption. Consequently, the formula provided above does not incorporate the factor of signal transmission time as well as issues with asynchronous, delayed measurements that have been studied in past works [22,23]. The sampling time Δ is carefully selected as 30 s, considering the aforementioned condition to ensure proper implementation. The integration step h_c is chosen as $(10^{-2} \times \Delta)$ to evaluate the cost function of the LMPC through the first-principles model. The positive definite matrix P in the control Lyapunov function $V = x^T P x$ for this system is taken as $P = \text{diag}[200\ \ 500\ \ 2500\ \ 10\ \ 0.25\ \ 1000\ \ 1000\ \ 500\ \ 1\ \ 0.5]$ based on extensive simulations. A prediction horizon of $N = 6$ is employed in the LMPC framework. To ensure stability in the LMPC, we set the criterion $\rho_{min} = 2$ to determine when the states have reached stability. Additionally, a contractive constraint of the form $\dot{V} \leq -kV$ is utilized for Equation (13f), where the value of k is chosen as 0.15. The weight matrices Q_1 and Q_2 in the LMPC cost function are chosen as $Q_1 = \text{diag}[5\ \ 5\ \ 650\ \ 5\ \ 2.5\ \ 25\ \ 25\ \ 100\ \ 2\ \ 6]$ and $Q_2 = \text{diag}[5 \times 10^{-6}\ \ \ 1.25 \times 10^{-5}]$, respectively. The cost function is defined as $L_2(x(t), u(t)) = x^T Q_1 x + u^T Q_2 u$.

4.6. Utilizing MPC over Traditional Control

In this section, we substantiate the utilization of Model Predictive Control (MPC) by conducting a comparative analysis between the MPC and the simpler p-control strategy. P-control allows control actions to be computed directly in encrypted states, eliminating the requirement for decryption at the controller through complex multiplicative homomorphic algorithms, such as the ElGamal cryptosystem. The MPC strategy is a more advanced control method that uses a mathematical model of the system to predict future behavior and optimize control actions accordingly. It requires decryption at the controller to obtain the necessary information for prediction and multi-constrained, nonlinear optimization, which cannot be performed in an encrypted space.

Figure 4 showcases its enhanced performance, with lower undershoot and faster settling time observed for the temperature of CSTR 1. Further, the temperature of CSTR 2

exhibits a significant reduction in overshoot by almost 50% and converges over 1 h before the p-control, within a settling limit of 0.25 K. Moreover, the evaluation of the normalized sum of the controller cost function ($L_2(x(t), u(t))$) over the closed-loop simulation reinforces the advantage of MPC over p-control, by the respective values of 0.86 and 1. These findings underscore the necessity of adopting MPC, as it offers reduced overshoot and undershoot, a faster settling time of state variables, and enhanced cost efficiency.

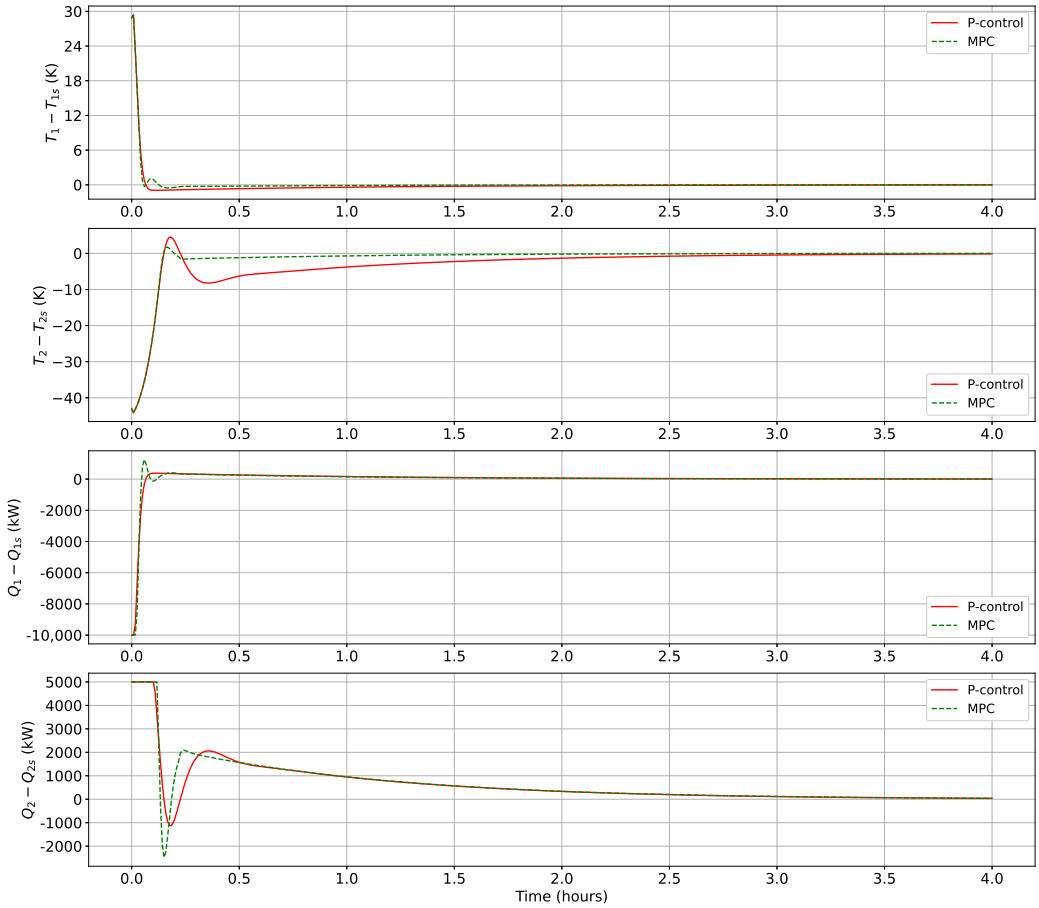

Figure 4. Temperature state and input profiles of p-control (red solid line) and MPC (green dashed line) strategies employed using the Aspen dynamic model.

Remark 5. *As mentioned earlier in Section 2.4, the Paillier cryptosystem is a partially homomorphic encryption scheme that does not support multiplication operations in an encrypted space. Therefore, in the above section, we mention using the ElGamal cryptosystem, which supports multiplicative homomorphism. Although the Paillier cryptosystem supports addition operations in an encrypted space, we do not utilize this property in our study. The Paillier cryptosystem is primarily selected for encryption due to its lower computational complexity compared to the ElGamal cryptosystem. This choice reduces the time and computational effort required for encryption–decryption processes.*

4.7. Simulation Results of the Encrypted LMPC

We apply the encrypted LMPC to the Aspen dynamic model, initialized from the following point:

$x_0 = [-1.11\,\text{kmol/m}^3 \;\; -1.16\,\text{kmol/m}^3 \;\; -0.3\,\text{kmol/m}^3 \;\; -8.76 \times 10^{-6}\,\text{kmol/m}^3 \;\; 28.85\,\text{K}$
$0.49\,\text{kmol/m}^3 \;\; 0.56\,\text{kmol/m}^3 \;\; -1.85\,\text{kmol/m}^3 \;\; -7.77 \times 10^{-6}\,\text{kmol/m}^3 \;\; -43\,\text{K}]$

We then observe the closed-loop simulation results for $d = 1, 4, 8$. A process time of 4 h allows both the states and control inputs to reach their respective steady-state values. Figures 5–7 display the temperature state and input profiles.

Remark 6. *As indicated in Section 4.1, the concentration states in the reactors exhibit open-loop trajectories as the reactor temperature converges to its steady-state value. Consequently, the presence or absence of encryption does not significantly affect these states because the manipulated input, i.e., the heat removed from the reactors, has no direct influence on the concentration states. Therefore, in this section, we focus solely on displaying the temperature states and control inputs, as encryption noticeably influences them.*

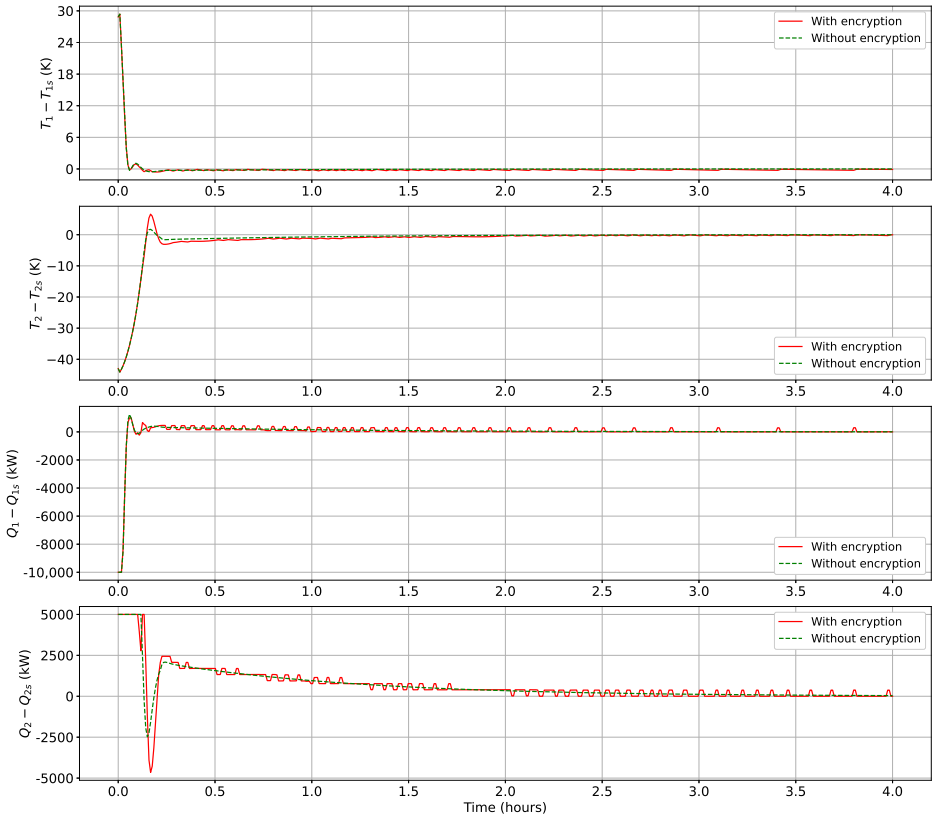

Figure 5. Temperature state and input profiles of the LMPC with encryption (red solid line) and without encryption (green dashed line) for the Aspen dynamic model, with $d = 1$.

For a quantization parameter of $d = 1$, it is evident that the state $T_1 - T_{1s}$ does not precisely converge to its steady-state value, instead exhibiting small oscillations around it throughout the 4 h process time. Also, the state $T_2 - T_{2s}$ demonstrates nearly double the overshoot with encryption and oscillates around the steady-state values, similar to the previous state. Further, quantized control inputs $Q_1 - Q_{1s}$ and $Q_2 - Q_{2s}$ experience significant oscillations under the encrypted MPC, rendering it incapable of effectively stabilizing the closed-loop system within a small neighborhood $\Omega_{\rho_{\min}}$ around the origin. Although, it does stabilize the system within the larger neighborhood $\Omega_{\hat{\rho}}$. This behavior can be attributed to the quantization error resulting from the quantization of the state measurements. Thus, we establish that errors due to quantization can be more significant than plant/model mismatch errors as the MPC without encryption and with a higher quantization parameter, $d = 8$, is stabilized within the small neighborhood $\Omega_{\rho_{\min}}$ around the origin. As indicated in Remark 7, the quantization error associated with the quantized control input can be deemed negligible. However, the quantization error emanating from the quantized states is significant given the range in which they lie during the closed-loop simulation. For $d = 1$, the quantized states are separated by a resolution of 2^{-1} or 0.5, leading to a high quantization error. When running simulations with the quantization parameter $d = 4$, we no longer observe oscillatory motions in the temperature states, and the magnitude of oscillations for the quantized inputs is much smaller compared to the case where $d = 1$. Furthermore, the amplitude of overshoot observed in the state variable

$T_2 - T_{2s}$ remains nearly unchanged when encryption is applied, and the system also reaches the steady state more rapidly.

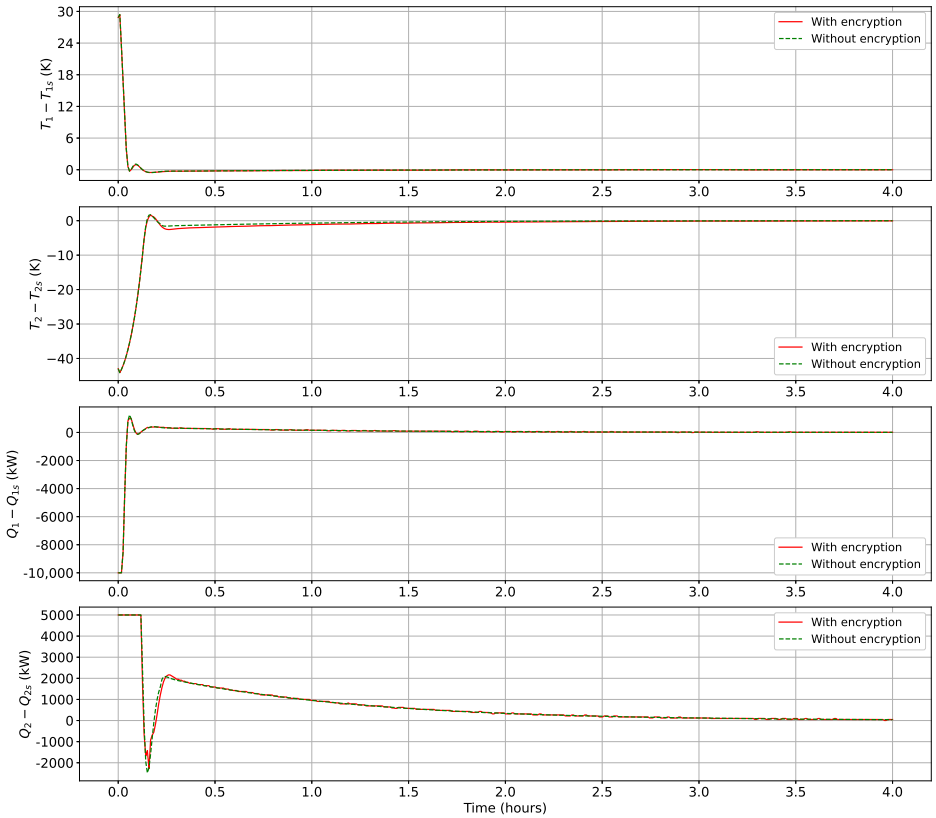

Figure 6. Temperature state and input profiles of the LMPC with encryption (red solid line) and without encryption (green dashed line) for the Aspen dynamic model, with $d = 4$.

It is important to note that as the quantization parameter increases, resulting in a lower resolution, the states and inputs converge more quickly and exhibit reduced oscillations. Therefore, a higher quantization parameter improves the convergence behavior and decreases fluctuations in the state and control input profiles. Specifically, when $d = 8$, the closed-loop trajectories of the temperature states and control inputs become nearly identical between the cases with encryption and without encryption. In other words, the impact of encryption on the system's behavior diminishes significantly as the quantization parameter increases, ultimately resulting in almost indistinguishable closed-loop trajectories for both scenarios.

Remark 7. *The total quantization error can be attributed to the state quantization rather than the control input quantization, because the magnitudes of the quantized control inputs generally fall within the order of magnitude three. For the case $d = 1$, representing the lowest quantization, the maximum permissible error in the control input calculated by the MPC (before encryption) and applied by the actuator (after decryption) is 0.25, corresponding to half of the resolution. This error is considered negligible compared to the overall control input. As a result, the error arising from the quantization of control inputs is insignificant, particularly for the specific example considered. However, it is crucial to acknowledge that if the control inputs have smaller magnitudes, the error resulting from the quantized control inputs would significantly impact the controller's performance.*

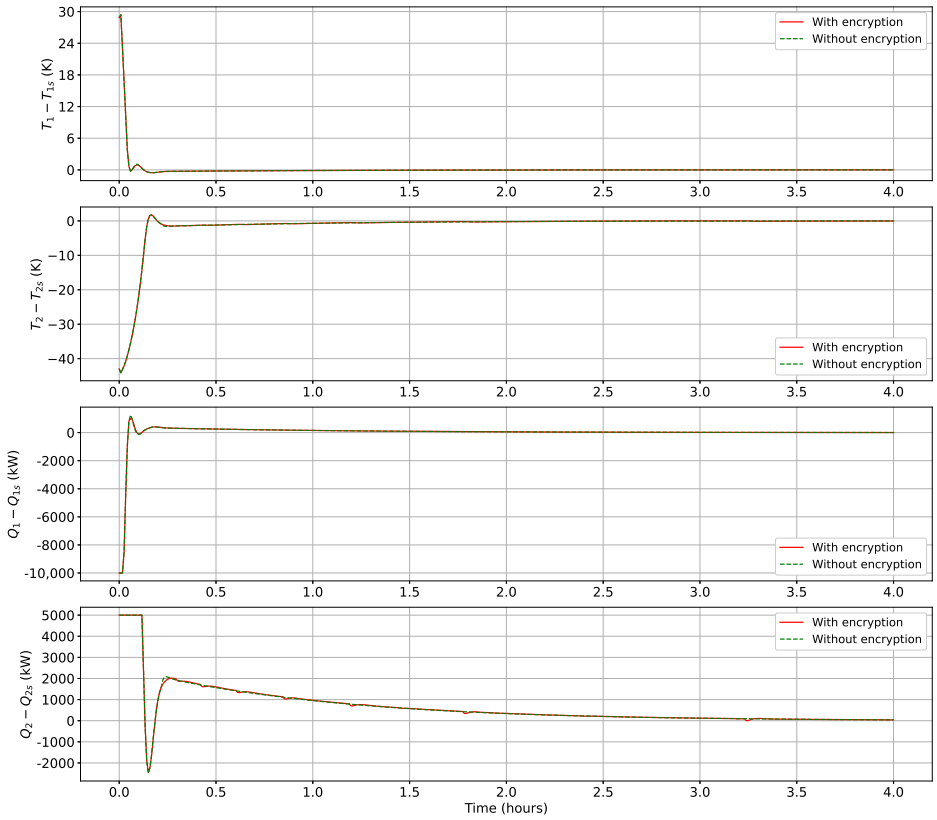

Figure 7. Temperature state and input profiles of the LMPC with encryption (red solid line) and without encryption (green dashed line) for the Aspen dynamic model, with $d = 8$.

5. Effect of the Quantization Parameter d and Encryption–Decryption on the Total Computational Time

This section discusses the impact of the quantization parameter, d, and encryption–decryption on the total control input calculation time. For an encrypted MPC, the total control input calculation time comprises two main components: the time required by the MPC to calculate the control action and the total time spent on encrypting–decrypting the state variables and control inputs.

5.1. Effect of the Quantization Parameter d on Computational Time

Table 2 provides an overview of the computation time required for the complete encryption–decryption process, considering a range of quantization parameters, $d = [1, 8]$. The table also offers a detailed breakdown of the time required for each sub-process involved. Analyzing Table 2, it becomes apparent that the computational time for the entire encryption–decryption process shows consistent values across the quantization parameters within the range $d = [1, 8]$.

However, as discussed in Section 4, a higher quantization parameter proves more advantageous for the LMPC. Specifically, for $d = 8$, the trajectories of the temperature states and control inputs closely resemble those without encryption. In contrast, for $d = 1$, there is a noticeable difference between the cases with and without encryption.

Furthermore, this table also reveals that the majority of the computational time is allocated to the encryption step, followed by the decryption step. Mapping the inputs to quantized states ($g_{l_1,d}$), bijective mapping ($f_{l_2,d}$) and inverse mapping ($f^{-1}_{l_2,d}$) contribute only a negligible fraction of the total time at each sampling instance. Although the computational time remains consistent across the quantization parameters, the number of search operations at each sampling instance increases linearly with the quantization parameter. This observation is presented in Table 3. Additionally, the time and number of

operations required to generate the set $\mathbb{Q}_{l_1,d}$ grow exponentially by increasing the quantization parameter, d. However, it is vital to note that this step is performed only once at the beginning of the process and is not repeated at each sampling instance. Consequently, selecting a higher quantization parameter remains favorable, as the operational time for encryption–decryption at each sampling instance remains unchanged, and a higher quantization parameter yields significantly improved results.

Table 2. Time required to encrypt–decrypt the 10 states and 2 inputs at a single sampling instance.

d	$g_{l_1,d}$ Time	$f_{l_2,d}$ Time	Enc. Time	Dec. Time	$f_{l_2,d}^{-1}$ Time	Total Time
1	4.8×10^{-4} s	2.6×10^{-4} s	2.49 s	0.72 s	2.9×10^{-4} s	3.204 s
2	4.5×10^{-4} s	2.9×10^{-4} s	2.48 s	0.71 s	3.1×10^{-4} s	3.190 s
3	4.7×10^{-4} s	2.7×10^{-4} s	2.48 s	0.7 s	2.8×10^{-4} s	3.179 s
4	4.8×10^{-4} s	2.9×10^{-4} s	2.48 s	0.71 s	2.8×10^{-4} s	3.182 s
5	5.3×10^{-4} s	2.7×10^{-4} s	2.5 s	0.71 s	2.8×10^{-4} s	3.214 s
6	5×10^{-4} s	2.9×10^{-4} s	2.47 s	0.71 s	3.2×10^{-4} s	3.182 s
7	5.1×10^{-4} s	3×10^{-4} s	2.49 s	0.71 s	3.3×10^{-4} s	3.194 s
8	5.4×10^{-4} s	2.9×10^{-4} s	2.5 s	0.73 s	3.1×10^{-4} s	3.225 s

Remark 8. *An alternative approach to mitigate the initial high computational time, especially when a higher quantization parameter d is selected, is to pre-generate the set $\mathbb{Q}_{l_1,d}$ before commencing the process operation with encryption–decryption on the hardware. By generating this set prior to the first sampling instance, we can avoid the need for additional time allocation during the actual control process. As mentioned in Section 5, as the quantization parameter increases, the time required to generate $\mathbb{Q}_{l_1,d}$ grows exponentially. Therefore, pre-generating the set is particularly beneficial in reducing the computational overhead during the initial sampling instance when dealing with larger quantization parameters. This approach allows for the utilization of higher quantization parameters without being hindered by the drawback of increased computational time in the first sampling instance.*

Table 3. Operations required for $g_{l_1,d}$, generating $\mathbb{Q}_{l_1,d}$, and time required to generate $\mathbb{Q}_{l_1,d}$.

d	Operations for $g_{l_1,d}$	Operations to Generate $\mathbb{Q}_{l_1,d}$	Time to Generate $\mathbb{Q}_{l_1,d}$
1	192	65,534	0.02 s
2	204	131,070	0.04 s
3	216	262,142	0.07 s
4	228	524,286	0.15 s
5	240	1,048,574	0.29 s
6	252	2,097,150	0.55 s
7	264	4,194,302	1.11 s
8	276	8,388,606	2.27 s

5.2. Effect of Encryption–Decryption on the Total Computational Time

Figure 8 shows that encryption–decryption takes approximately 45–65% of the total time required to calculate the control inputs for an encrypted LMPC, which is the sum of the time needed for MPC control action computation and encryption–decryption (of the 10 states and two control inputs). Moreover, this result is consistent over the quantization parameters $d = \{1, 2, 3, 4, 5, 6, 7, 8\}$. This substantiates the fact that the decision regarding

the choice of a quantization parameter does not necessarily result in a substantial alteration of the ratio between the time devoted to encryption–decryption and the total duration of MPC computation.

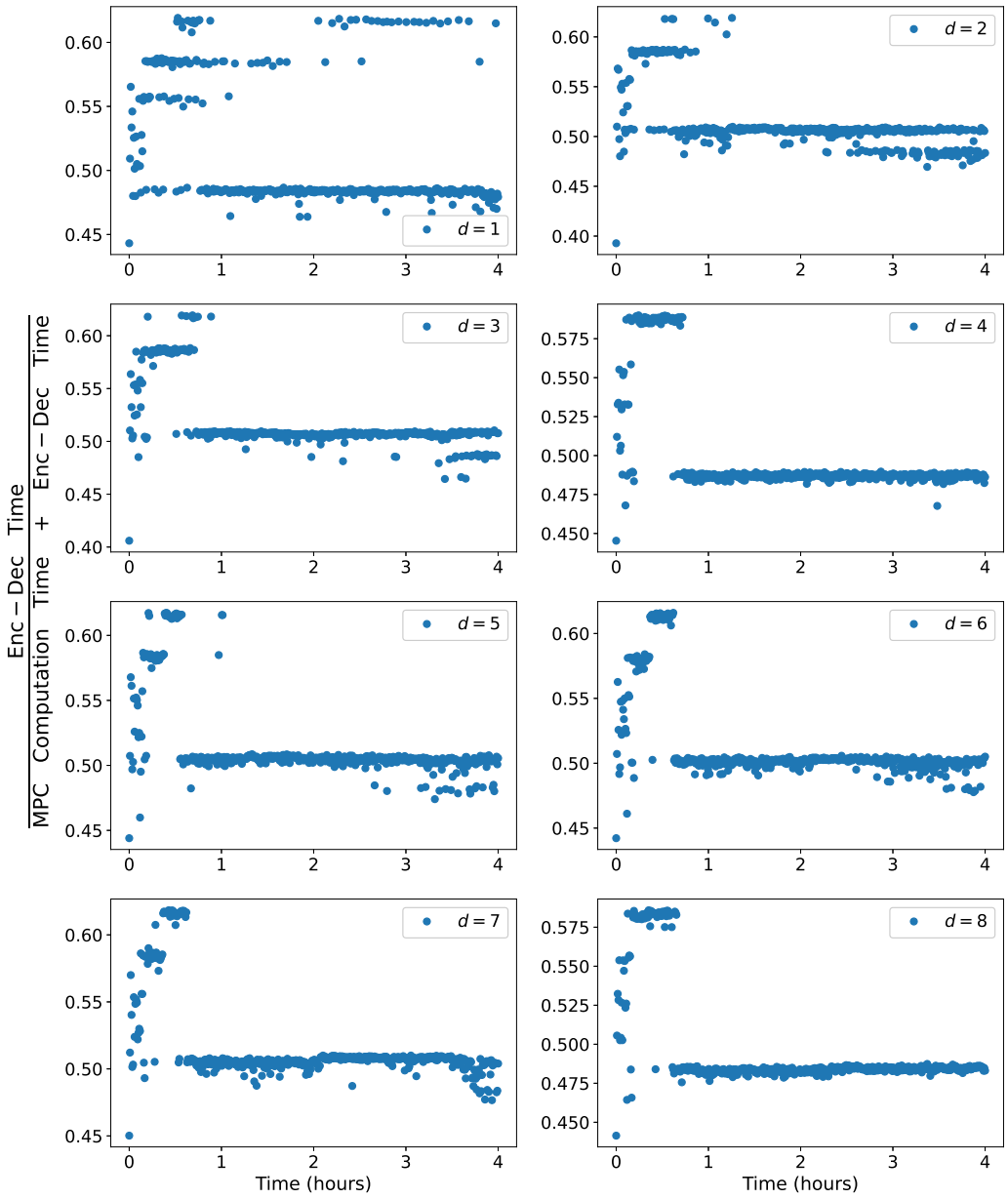

Figure 8. Ratio of the total time spent for encryption–decryption to the sum of the total time required for MPC computation and encryption–decryption at each sampling instance.

As previously discussed in Section 4.5, it is essential to select a sampling time, Δ, that exceeds the combined maximum duration of the encryption–decryption process and the MPC computation time for any given sampling instance. This criterion applies to all considered quantization parameters. For the example examined in this study, the minimum required sampling time was determined to be 9 s. Consequently, a sampling time of 30 s was selected, which exceeded the minimum requirement.

Remark 9. *Maintaining system stability and ensuring effective control requires avoiding excessively large sampling times, particularly in cases where the system operates at an unstable steady state and has bounded control inputs. Going beyond a certain threshold in sampling time can impede the ability of the controller to successfully regulate the system. To validate this concept, we*

conducted an experiment on the example discussed in Section 4. We applied LMPC control without encryption and increased the sampling time for the process in 30 s increments. The results showed that the controller achieved the desired steady state for a sampling time up to 2.5 min. However, extending the sampling time to 3 min prevented the controller from achieving the desired outcome. This observation emphasizes the significance of selecting an appropriate sampling time that ensures effective control action and system stability.

Remark 10. *In order to maintain manageable encryption–decryption times within an encrypted control system network, it is essential to choose computationally efficient cryptosystems, such as the Paillier cryptosystem. Cryptosystems like ElGamal and AES impose higher computational requirements on process control hardware, resulting in longer encryption and decryption times. Consequently, this leads to the need for longer sampling times. Further, in practical applications, it may be feasible to reduce the prediction horizon of the MPC for encrypted control as long as it does not significantly impact the performance of the controller. These adjustments enable shorter sampling times while still meeting encryption requirements.*

Remark 11. *When dealing with large-scale processes with hundreds or thousands of measurements, it would be advisable to employ a distributed SCADA architecture across multiple locations or nodes within the network. Furthermore, encryption of state measurements at the sensor can be performed in parallel rather than in series. When we report the encryption time in this paper, it is the total time needed for encrypting each sensor signal and control input in series, not in parallel. This could be performed in a parallel manner across multiple devices for larger systems to reduce the effective computational time needed.*

Remark 12. *To deal with asynchronous or delayed signals in an encrypted setting, the signals would be encrypted prior to transmission and decrypted upon receipt, with the actuator designed to apply control inputs in a sample-and-hold manner, whereby the preceding control input trajectory continues to be implemented until the recalculated input trajectory is received. Because quantization with encryption has a consistent computational duration, an appropriate sampling time would be chosen based on its knowledge and time needed to compute the control input, as demonstrated in Equation (18). However, because the formula given to decide the sampling time does not take into account the time spent for signal communication or signal delays, which are very specific to the process setting, sensors used, and communication channels established, the time spent between asynchronous measurements or for signal delays could be known or approximated to select an appropriate sampling time.*

6. Conclusions

In this work, we developed and applied an Encrypted Lyapunov-based Model Predictive Control (LMPC) scheme to a large-scale chemical process network involved in the production of Ethylbenzene. By employing the encrypted LMPC, we conducted closed-loop simulations for different quantization parameters and identified errors resulting from quantization. We illustrated that the effect of quantization could be more profound than plant/model mismatch when a low quantization parameter is chosen. To mitigate the impact of quantization, we proposed using a higher quantization parameter, specifically $d = 8$. Furthermore, through a comprehensive analysis of the duration of encryption–decryption at each sampling instance, we observed that the computational burden on the control input calculation time remained consistent across all tested quantization parameters. This finding supports the recommendation of employing a higher quantization parameter, as it not only minimizes the impact of quantization errors but also ensures secure communication between the sensor–controller and controller–actuator, thus enhancing system cybersecurity without compromising the performance of the controller. The current research necessitates MPC computations to be executed within a fully secure cyber–physical environment, aimed to thwart cyberattackers from compromising the decrypted plaintext input signals and control inputs computed by the MPC prior to encryption. An avenue for

future research could involve adapting the encrypted MPC architecture to operate within a less secure context. Additionally, another promising area for future investigation could entail implementing encrypted MPC while incorporating data reconciliation mechanisms amidst a cyberattack scenario. Notably, the works referenced [6,9,24,25] in this context have explored such aspects within non-encrypted settings.

Author Contributions: Conceptualization, Y.A.K., A.S., F.A., A.A. and P.D.C.; methodology, Y.A.K., A.S., F.A., A.A. and P.D.C.; software, Y.A.K., A.S., F.A. and A.A.; validation, Y.A.K., A.S., F.A. and A.A.; formal analysis, Y.A.K., A.S., F.A. and A.A.; investigation, Y.A.K., A.S., F.A. and A.A.; resources, P.D.C.; data curation, Y.A.K., A.S., F.A. and A.A.; writing—original draft preparation, Y.A.K., A.S. and F.A.; writing—review and editing, Y.A.K., A.S., F.A., A.A. and P.D.C.; supervision, P.D.C.; project administration, P.D.C.; funding acquisition, P.D.C. All authors have read and agreed to the published version of the manuscript.

Funding: Financial support from the National Science Foundation, CBET-2227241, is gratefully acknowledged.

Data Availability Statement: Data is available upon request to the corresponding author.

Conflicts of Interest: The authors declare that they have no conflict of interest regarding the publication of this research article.

References

1. Kushner, D. The real story of stuxnet. *IEEE Spectr.* **2013**, *50*, 48–53. [CrossRef]
2. Barrett, M.P. *Framework for Improving Critical Infrastructure Cybersecurity (Version 1.1)*; Cybersecurity Framework; NIST: Gaithersburg, ML, USA, 2018.
3. Huang, L.; Nguyen, X.; Garofalakis, M.; Hellerstein, J.M.; Jordan, M.I.; Joseph, A.D.; Taft, N. Communication-efficient online detection of network-wide anomalies. In Proceedings of the 26th IEEE International Conference on Computer Communications, Anchorage, AK, USA, 6–12 May 2007; pp. 134–142.
4. Omar, S.; Ngadi, A.; Jebur, H.H. Machine learning techniques for anomaly detection: an overview. *Int. J. Comput. Appl.* **2013**, *79*, 33–41. [CrossRef]
5. Agrawal, S.; Agrawal, J. Survey on anomaly detection using data mining techniques. *Procedia Comput. Sci.* **2015**, *60*, 708–713. [CrossRef]
6. Wu, Z.; Albalawi, F.; Zhang, J.; Zhang, Z.; Durand, H.; Christofides, P.D. Detecting and Handling Cyber-attacks in Model Predictive Control of Chemical Processes. *Mathematics* **2018**, *6*, 173. [CrossRef]
7. Chen, S.; Wu, Z.; Christofides, P.D. A cyber-secure control-detector architecture for nonlinear processes. *AIChE J.* **2020**, *66*, e16907. [CrossRef]
8. Wu, Z.; Chen, S.; Rincon, D.; Christofides, P.D. Post cyber-attack state reconstruction for nonlinear processes using machine learning. *Chem. Eng. Res. Des.* **2020**, *159*, 248–261. [CrossRef]
9. Durand, H. A nonlinear systems framework for cyberattack prevention for chemical process control systems. *Mathematics* **2018**, *6*, 169. [CrossRef]
10. Durand, H.; Wegener, M. Mitigating safety concerns and profit/production losses for chemical process control systems under cyberattacks via design/control methods. *Mathematics* **2020**, *8*, 499. [CrossRef]
11. Elgamal, T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory* **1985**, *31*, 469–472. [CrossRef]
12. Suryavanshi, A.; Alnajdi, A.; Alhajeri, M.; Abdullah, F.; Christofides, P.D. Encrypted model predictive control design for security to cyberattacks. *AIChE J.* **2023**, *69*, e18104. [CrossRef]
13. Wu, Z.; Tran, A.; Rincon, D.; Christofides, P.D. Machine Learning-Based Predictive Control of Nonlinear Processes. Part I: Theory. *AIChE J.* **2019**, *65*, e16729. [CrossRef]
14. Wu, Z.; Tran, A.; Rincon, D.; Christofides, P.D. Machine Learning-Based Predictive Control of Nonlinear Processes. Part II: Computational Implementation. *AIChE J.* **2019**, *65*, e16734. [CrossRef]
15. Lin, Y.; Sontag, E.D. A universal formula for stabilization with bounded controls. *Syst. Control Lett.* **1991**, *16*, 393–397. [CrossRef]
16. Paillier, P. Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology—EUROCRYPT'99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, 2–6 May 1999*; Springer: Berlin/Heidelberg, Germany, 1999; pp. 223–238.
17. Darup, M.S.; Redder, A.; Shames, I.; Farokhi, F.; Quevedo, D. Towards encrypted MPC for linear constrained systems. *IEEE Control Syst. Lett.* **2017**, *2*, 195–200. [CrossRef]
18. Heidarinejad, M.; Liu, J.; Christofides, P.D. Economic model predictive control of nonlinear process systems using Lyapunov techniques. *AIChE J.* **2012**, *58*, 855–870. [CrossRef]

19. Mhaskar, P.; El-Farra, N.H.; Christofides, P.D. Stabilization of nonlinear systems with state and control constraints using Lyapunov-based predictive control. *Syst. Control Lett.* **2006**, *55*, 650–659. [CrossRef]
20. CSIRO's Data61. Python Paillier Library. 2013. Available online: https://github.com/data61/python-paillier (accessed on 3 March 2023).
21. Wächter, A.; Biegler, L.T. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.* **2006**, *106*, 25–57. [CrossRef]
22. Liu, J.; Munoz de la Pena, D.; Ohran, B.J.; Christofides, P.D.; Davis, J.F. A two-tier control architecture for nonlinear process systems with continuous/asynchronous feedback. *Int. J. Control* **2010**, *83*, 257–272. [CrossRef]
23. Liu, J.; Chen, X.; de la Pena, D.M.M.; Christofides, P.D. Iterative distributed model predictive control of nonlinear systems: Handling asynchronous, delayed measurements. *IEEE Trans. Autom. Control* **2011**, *57*, 528–534.
24. Mercorelli, P. A fault detection and data reconciliation algorithm in technical processes with the help of Haar wavelets packets. *Algorithms* **2017**, *10*, 13. [CrossRef]
25. Schimmack, M.; Mercorelli, P. An adaptive derivative estimator for fault-detection using a dynamic system with a suboptimal parameter. *Algorithms* **2019**, *12*, 101. [CrossRef]