Annual Reviews in Control 56 (2023) 100913

Contents lists available at ScienceDirect

Annual Reviews in Control

journal homepage: www.elsevier.com/locate/arcontrol

Review article

A brief survey on encrypted control: From the first to the second generation and beyond

Nils Schlüter, Philipp Binfet, Moritz Schulze Darup [*]

Department of Mechanical Engineering, TU Dortmund University, Leonhard-Euler-Str. 2, Dortmund, 44227, Germany

ARTICLE INFO

Keywords:
Encrypted control
Homomorphic encryption
Secure multi-party computation

ABSTRACT

This article provides a comprehensive and illustrative presentation of the young field of encrypted control. In particular, we survey the evolution of encrypted controllers from their first appearance in 2015 until 2023 and derive a categorization into two generations mainly characterized by the utilized cryptographic methods. We further envision future developments and challenges of encrypted control. Throughout our presentation, we build less on technicalities but rather on intuitive tutorial-style explanations. This way, we intend to build a bridge from control engineering to cryptography and to make the interdisciplinary field of encrypted control more accessible.

1. Introduction and overview

The growing interconnectivity in control systems due to robust wireless communication and cloud usage paves the way for a future where data-driven (including machine learning) and distributed control as well as decision-making is poised to play a significant role in applications. While these areas are currently under active research and development, their potential impact is highly anticipated in cloud-based control, industry 4.0, intelligent transportation, robotics, and building automation. Generally speaking, interconnected control systems offer several distinct advantages when compared to traditional control systems. Among others, these include leveraging data, resource pooling and outsourcing, rapid scalability, ease of use, as well as maintainability.

For a good reason, one of the most influential concepts in control theory is safety, which typically comes in the flavors of stability, robustness, reliability, or resilience, which are main driving forces in research. With increasing connectivity, however, considering safety only in terms of classical control notions falls short. In fact, an increased connectivity comes with a pressing need to address cybersecurity threats in control systems. Typically, these threats are categorized into confidentiality, integrity, and availability (CIA) threats (Bishop et al., 2005). Confidentiality deals with the privacy of data. Integrity of data is achieved when it has not been modified in an unauthorized or undetected manner. Availability considers the accessibility of data and services. Ensuring privacy and integrity in a cyberphysical system (CPS) are more recent efforts, whereas loss of availability via denial-of-service attacks has been investigated earlier in networked control (see, e.g., Amin, Cárdenas, & Sastry, 2009; Cetinkaya, Ishii, & Hayakawa, 2019; De Persis & Tesi, 2015). Preventing attacks on the availability (instead of dealing with their results) is attributed to the field of computer science.

It may be surprising that even (seemingly) isolated CPS became targets of tailored malware attacks (Alladi, Chamola, & Zeadally, 2020; Hemsley, Fisher, et al., 2018), where data thefts and data erasures as well as unauthorized control access are common functionalities. Possible outcomes of such attacks include loss of safety, data, life, production, and competitive advantage as well as an impact on the society and the environment. Thus, a safe control system design should take cybersecurity threats into account.

Sparked by the aforementioned attacks, many research efforts based on control-related methods were undertaken (Chong, Sandberg, & Teixeira, 2019). When it comes to legacy control systems, which are especially vulnerable because they often lack basic security features due to long-lasting and rarely updated hardware, it is typically assumed that the communication network is unsecure. Namely, sensor and/or actuator signals can be eavesdropped and maliciously perturbed by an attacker. Since an attacker that just inflicts damage will be easily detected, more evolved strategies consider some sort of stealthiness. This gave rise to the notions of zero-dynamic attacks (Pasqualetti, Dörfler, & Bullo, 2013), covert attacks (Smith, 2011), or bias injection attacks (Teixeira, Shames, Sandberg, & Johansson, 2015). These differ in several aspects such as impact, signal access, and required knowledge but have in common that they will not trigger an anomaly detector. Another type of attack is replay (Mo & Sinopoli, 2009, 2012), where a signal is stored and later used to replace the correct signal, which

[*] Corresponding author.
E-mail address: moritz.schulzedarup@tu-dortmund.de (M. Schulze Darup).

https://doi.org/10.1016/j.arcontrol.2023.100913
Received 24 July 2023; Received in revised form 9 October 2023; Accepted 23 October 2023
Available online 18 November 2023
1367-5788/© 2023 Elsevier Ltd. All rights reserved.

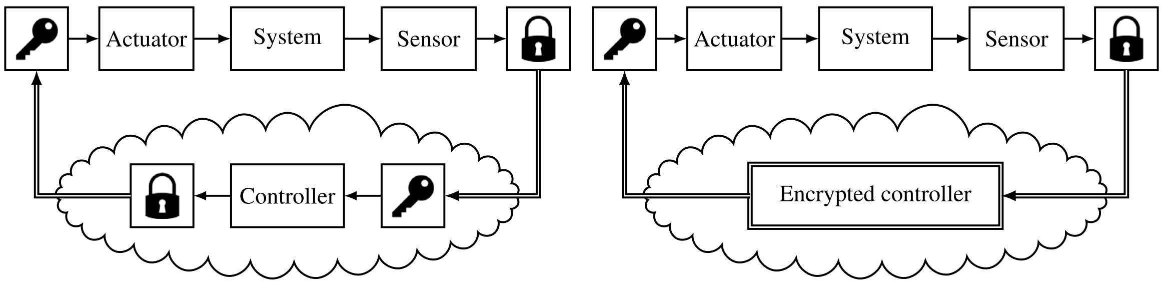

Fig. 1. Standard illustration (cf. Schulze Darup, Alexandru, Quevedo, and Pappas (2021, Fig. 1)) of a cloud-based control scheme with encrypted communications but insecure controller evaluation (left) and with encrypted communications *and* controller (right). Double-arrows and double-framing highlight encrypted data transmission and encrypted data processing, respectively.

is also possible with encrypted data. Because old data is replayed, this goes unnoticed by detectors. These attacks have been mitigated by constructing watermarks (Ferrari & Teixeira, 2017), that can also address stealthy attacks.

While it is insightful to study system theoretic viewpoints on cyber-attacks and defense mechanisms, one should naturally also make use of cryptographic tools to secure CPS. For instance, data confidentiality and integrity during communications can easily be ensured using standard key exchange protocols (Diffie & Hellman, 2022), encryption schemes (Daemen & Rijmen, 1999), and message authentication codes (MAC) (Katz & Lindell, 2020, p.135). These methods are well-established and, except for the key exchange (which is performed only once), have little computational cost. Furthermore, by considering cryptographic nonces (i.e., random numbers used once) or extending the messages by timestamps, MAC can also effectively prevent replay attacks.

Now, a characteristic property of networked control systems (NCS) is that data is processed externally (in order to, e.g., evaluate control laws) on not necessarily trustworthy platforms such as cloud-servers or neighboring agents. Traditionally, performing computations requires decrypting data and removing authentication codes (cf. Fig. 1 on the left). Hence, secure communication on its own is typically not sufficient to secure NCS. In fact, confidentiality and integrity of control-related data should ideally be ensured throughout the entire control loop. In terms of confidentiality, this is the main focus in encrypted control, which builds on tailored combinations of control algorithms and special cryptosystems with homomorphic properties (see Schulze Darup et al. (2021) for an early survey). In particular, these cryptosystems provide an encryption procedure preserving some of the algebraic structure such that some sort of computation on encrypted data is possible. Clearly, this allows to securely outsource computations and to subsequently decrypt the desired results in a trustworthy environment. For a cloud-based controller, this concept is illustrated on the right in Fig. 1. More generally, encrypted control also enables (among other applications) privacy-preserving control-as-a-service, processing of sensitive data (personal health data, companies' data), and the mitigation of attacks that require system knowledge.

The interdisciplinary approach of designing encrypted controllers requires deep knowledge of both networked control and suitable cryptosystems. Given that most readers will have a background in control, the paper mainly intends to build a bridge to cryptography. To this end, an initial overview of several methods for privacy-preserving computations is given in Table 1. The table first lists homomorphic encryption (HE), where schemes based on ring learning-with-errors (RLWE) are most competitive, and secure multi-party computation (SMPC). As apparent from the table, HE and SMPC stand out for high security guarantees (which will be specified later). In fact, under reasonable assumptions, it is impossible for an attacker to gain information about plaintexts based on ciphertexts (or shares) resulting from these schemes. In addition, both HE and SMPC can provide highly accurate computation results without compromising on security. It is this combination of beneficial features, which make HE and SMPC from our point of view particularly suited for secure control applications. In fact, while the remaining two options in Table 1, differential privacy (DP) and random affine transformations (RAT), are generally cheaper in terms of computational effort, they require trading off security against accuracy or have security issues in practical implementations, respectively. As a consequence, we focus on HE and SMPC in the remainder of this paper. Nevertheless, we next briefly discuss DP and RAT for completeness and differentiation.

DP has been proposed in Dwork (2006) for static-dataset analyses when an attacker has access to background information. The security notion in DP guarantees that the presence or absence of a dataset entry will not significantly affect the publicly visible computation output. Intuitively, privacy then comes from an attacker's inability to determine the contribution of any specific entry to the output. This is achieved by adding (non-trivially constructed) noise before releasing an output, where bigger noise provides more security. As a result, there is a trade-off between security and accuracy while, on the other hand, computations do not impose an overhead as it is the case in HE or SMPC. When applied to control, DP must be adapted for dynamic data-streams. In this context, Han and Pappas (2018) provide an introduction to the topic, whereas (Hassan, Rehmani, & Chen, 2019) give a broad overview of DP in CPS. In control related applications, DP is naturally considered a multi-party setup, where an agent's contribution corresponds to a dataset entry from above that is processed by a trusted entity or (Laplacian) noise is added to an agent's output followed by an aggregation step (Wang, Huang, Mitra, & Dullerud, 2017). This is why DP is often considered for consensus problems (Huang, Mitra, & Vaidya, 2015; Nozari, Tallapragada, & Cortés, 2016), federated learning (Wei et al., 2020), or distributed optimization (Hale & Egerstedt, 2015; Han, Topcu, & Pappas, 2017). Note that the use of DP for a small number of parties is restrictive, since the required noise drowns the result of the computation.

RAT are based on the simple idea to encrypt a vector by multiplying with a random matrix and adding a random vector. As a result, one essentially obtains high accuracy computations without any overhead or ciphertext size increase. RAT were first proposed for matrix products (Du & Zhan, 2002; Lei, Liao, Huang, & Heriniaina, 2014; Shan, Ren, Blanton, & Wang, 2018) but have also been applied in the context of model predictive control recently (Naseri, Lucia, & Youssef, 2022; Sultangazin & Tabuada, 2020). Furthermore, in Wang, Ren, and Wang (2011) and Xu and Zhu (2015) not only the confidentiality but also the integrity of the data is addressed. However, the cryptanalysis by Schlüter, Binfet, and Schulze Darup (2023) shows that they are only secure when implemented over real numbers. A practical implementation on a digital machine is not recommendable.

Lastly, we note that apart from such software-based solutions (listed in Table 1) hardware enclaves (or other trusted execution environments) can protect the data's confidentiality. In hardware enclaves, data is only decrypted and processed in a trusted environment, which protects it from all other software (even with full privilege) in a system. From a broader perspective, this may create the impression of

Table 1
Overview of approaches for privacy-preserving computations in CPS. The symbols "✓" and "✗" stand for yes/available and no/unavailable, respectively.

	Homomorphic encryption (RLWE)	SMPC (secret sharing, garbled circuits,...)	Differential privacy	Random affine transformations
Single party computation	✓	✗	✗	✓
High achievable security	✓	✓[a]	✓	✗
Low computational complexity	✗[b]	✓	✓	✓
Low communication overhead	✓[c]	✗	✓	✓
Memory-efficient ciphertexts	✗[d]	✓[e]	✓	✓
High achievable accuracy	✓[f]	✓	✗	✓
Any function computable	✓	✓	✓	✗[g]
Memory-efficient auxiliary data	✗[h]	✗[i]	✓	✓
Easy entry to the field	✗	✗	✗	✓

[a] If parties are not colluding to an extent that cannot be handled by the scheme;
[b] Single operations become slower for "deeper" computations;
[c] After transmitting the input data;
[d] Depends on the depth of computation;
[e] Increases linearly with the number of computing parties;
[f] Possible but increases the computational complexity significantly;
[g] Only used for specific applications;
[h] In many schemes, switching keys such as evaluation and automorphism keys are precomputed and stored;
[i] Often, large amounts of correlated randomness are precomputed and stored.

computing on encrypted data. Despite providing little overhead, there are several reasons why we do not consider hardware enclaves further in this survey. Namely, trust in the vendor and the provider is required, they suffer from side-channel attacks, and require specialized hardware.

Before stepping into a detailed analysis of encrypted controllers in the upcoming sections, let us briefly summarize the introduction for convenience. The safety of connected control systems crucially depends also on their security against cyberphysical threats. In modern control systems, the communication can be secured with well-established methods. Thus, mainly the execution of control algorithms on external platforms, that are either untrustworthy or vulnerable to attacks, is a wide-ranging problem. In this context, privacy is addressed in encrypted control by means of special frameworks for secure computation. In order to face these challenges sustainably, a high security guarantee is of paramount importance. In fact, there are countless examples of schemes that were broken (Katz & Lindell, 2020), which is disastrous when the schemes have been deployed at scale. To make sure that this requirement for security does not reduce the quality of operation (e.g., through low accuracy computations), we will limit the scope of the ensuing survey to encrypted controllers implemented using HE and SMPC in line with the reasoning given above.

The remainder of this paper is organized as follows. Sections 2 to 4 are dedicated to a survey of popular cryptosystems and related encrypted controllers. It will turn out that (the majority of) the designed controllers can be subdivided into two classes based on the applied cryptosystems. More precisely, many early realizations of encrypted controllers make use of so-called partially HE (PHE) offering only a very limited set of encrypted operations. We will associate this group with the first generation of encrypted control and discuss it in Section 2. The second generation of encrypted controllers typically makes use of more complex encrypted operations. In this context, we will further distinguish between controllers mainly building on fully HE (FHE[1]) and SMPC, and present the corresponding schemes in Sections 3 and 4, respectively. In both sections, we first introduce related cryptographic tools and then survey their usage for encrypted control. Finally, we envision different challenges, future developments, and applications of encrypted control in Section 5, and conclude the paper in Section 6.

[1] As specified in Sections 2 and 3, we focus on arithmetic operations (rather than binary) and consequently associate FHE with cryptosystems supporting (an unlimited number of) encrypted additions *and* multiplications. Often, applications only require a limited number of operations (especially multiplications) as provided by *leveled* FHE schemes.

Note that this survey differs from the early survey conducted by Schulze Darup et al. (2021) in that, here, the focus lies on the second generation of encrypted controllers, which was still under development during the writing of the early survey. Moreover, this survey intends to make the potential of (leveled) FHE more accessible to the control community, as it constitutes the backbone of many recent encrypted control schemes. This intention is similar to the recent work by Kim et al. (2022). However, we focus on a different leveled FHE scheme, and we take more controller types into account. Since some of the required cryptographic and number theoretic concepts may be new to many readers, we build on intuitive explanations and provide sidebars for more detailed information throughout the article.

Notation

The sets of complex, real, integer, and natural numbers are denoted by $\mathbb{C}$, $\mathbb{R}$, $\mathbb{Z}$, and $\mathbb{N}$ respectively. Central to this work is the set $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ of residue classes of the integers modulo $q \in \mathbb{N}$. In this regard, for $z \in \mathbb{Z}$, the modulo reduction $z \bmod q$ is a mapping from $\mathbb{Z}$ onto the representatives of $\mathbb{Z}_q$. Further, we write $a = b \pmod q$ to indicate that the modulo reduction is applied to both sides of the equation. The quotient ring $\mathcal{R}_q = \mathcal{R}/q\mathcal{R} = \mathbb{Z}_q[X]/(X^N + 1)$ contains polynomials $a(X)$ of (at most) order $N-1$ and coefficients in $\mathbb{Z}_q$ (a more detailed explanation is given later in Sidebar 2). Next, the brackets $\lfloor\cdot\rfloor$, $\lceil\cdot\rceil$, and $\lfloor\cdot\rceil$, refer to the (component-wise) floor, ceiling, and rounding to the nearest integer, respectively. Further, $(a \leq b) \in \{0, 1\}$ is the outcome of the comparison $a \leq b$. Hence, $(a \leq b)\, c$ results in c if $b \geq a$ and 0 otherwise, which is commonly used in cryptographic literature. Finally, the (rarely used) imaginary unit is denoted by i and the complex conjugation of $z \in \mathbb{C}$ is $\bar{z}$. Matrices $\boldsymbol{A}$ and vectors $\boldsymbol{a}$ are denoted with bold letters, where $\boldsymbol{A}_{i:j}$ refers to selecting the rows i to j from $\boldsymbol{A}$. We write $\mathsf{negl}(\kappa)$ to denote a function that is negligible in κ meaning that it decays faster than any inverse polynomial in κ.

2. First generation using PHE

The set of primitive operations supported by HE schemes is always very limited. In fact, even if only encrypted additions *and* multiplications are offered (and can be carried out arbitrarily often), an HE scheme is called fully homomorphic. The reason is that these operations are, theoretically, sufficient for functional completeness (in terms of logic circuits). However, early HE schemes such as ElGamal (1985) or Paillier (1999) solely provide encrypted additions *or* multiplications, respectively. While the computational capabilities are quite limited, these partially HE (PHE) schemes are relatively easy to apply and powerful enough to encrypt simple controllers such as linear state feedback. As a consequence, many early encrypted control schemes build on PHE. We briefly summarize the cryptographic basics in the following, and then survey the first generation of encrypted controllers.

2.1. Cryptographic basics

Any cryptosystem comes with two algorithms, namely an encryption and a decryption. First, the encryption takes a message z from the cryptosystem's message space $\mathcal{M}$, which typically is a subset of the integers such as $\mathbb{Z}_q$, as well as an encryption key ek and outputs a ciphertext $\mathsf{ct}(z) = \mathsf{Enc}_{\mathsf{ek}}(z)$ in the ciphertext space $\mathcal{C}$. Second, the decryption algorithm takes ct as well as a secret key sk and outputs $z = \mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}(z))$, where the reconstruction of z is a requirement for correctness. Depending on the nature of ek, one distinguishes between secret key schemes (or symmetric encryption), where $\mathsf{ek} = \mathsf{sk}$, and public key schemes (or asymmetric encryption), where ek is a public key pk, which can be distributed safely allowing anyone to encrypt messages. We refer to Katz and Lindell (2020) for details on the discussed cryptographic basics and further insights.

2.1.1. Partial homomorphic encryption

Notably, most HE schemes support public key encryption and also public evaluations of encrypted operations. In case of PHE, the provided operations are either encrypted additions or multiplications. More formally, additively HE schemes such as Paillier (1999)[2] offer an operation "$\oplus$", which allows the evaluation of encrypted additions according to

$$\mathsf{ct}(z_1 + z_2) = \mathsf{ct}(z_1) \oplus \mathsf{ct}(z_2) \tag{1}$$

for $z_i \in \mathcal{M}$. Analogously, multiplicatively HE schemes such as ElGamal (1985) or the (insecure textbook variant of the) RSA scheme (Rivest, Shamir, & Adleman, 1978) support encrypted multiplications "$\otimes$" via

$$\mathsf{ct}(z_1 z_2) = \mathsf{ct}(z_1) \otimes \mathsf{ct}(z_2). \tag{2}$$

For all mentioned schemes, evaluating the encrypted operations is enabled by virtue of $\mathsf{Enc}(\cdot)$ preserving algebraic structure.

While PHE schemes support either (1) or (2) but not both, further operations can still be derived. In fact, if (1) is available, one can always realize multiplications by a public positive integer c via

$$\mathsf{ct}(cz) = \underbrace{\mathsf{ct}(z) \oplus \cdots \oplus \mathsf{ct}(z)}_{c \text{ times}}. \tag{3}$$

Likewise, exponentiation by a public constant is possible if (2) is given. Remarkably, while the repeated addition in (3) provides the theoretical foundation, partially encrypted additions can typically be evaluated more efficiently by a single operation to which we refer with "$\odot$".

2.1.2. Security guarantees

The application of cryptosystems is only meaningful if a certain level of security is guaranteed. For HE schemes, the highest certificate that can be obtained is, by construction, security against chosen-plaintext attacks (see Fontaine and Galand (2007, Chapter 2.4) or Marcolla et al. (2022, Chapter 4.C) for details). In this attack scenario, the adversary has the ability to obtain encryptions $\mathsf{ct}(z)$ of an arbitrary message $z \in \mathcal{M}$ chosen by the adversary. While being a theoretical attack model in the first place, the scenario has real-world manifestations. For instance, the situation arises if public key encryption is used, if the attacker has access to the encryption as a black box, or if the attacker has background knowledge (in control, e.g., about multiple setpoints or reference values). The goal of the attacker is then to gain further information on, e.g., the secret key sk or new plaintexts. If achieving this is practically intractable, the cipher is secure under chosen plaintext-attacks.

Let us now specify what this means in more detail. Without loss of generality, the simplest message space $\mathcal{M} = \{0, 1\}$ is selected. In a game-based security proof, the adversary $\mathcal{A}$ is given an encryption $\mathsf{ct}(m)$ for some message $m \in \{0, 1\}$, which $\mathcal{A}$ has to guess. To this end, $\mathcal{A}$ is given a polynomial (in κ) number of encryptions and is allowed to carry out arbitrary computations for polynomial time before making its guess $\mathcal{A}(\mathsf{ct}(m))$. Under these assumptions, the scheme is secure under chosen plaintext attacks if

$$|\Pr[\mathcal{A}(\mathsf{ct}(0)) = 1] - \Pr[\mathcal{A}(\mathsf{ct}(1)) = 1]| \leq \mathsf{negl}(\kappa),$$

i.e., the adversary cannot distinguish between encryptions of 0 and encryptions of 1 with more than negligible probability. This security notion is therefore also referred to as indistinguishability under chosen-plaintext attacks (abbreviated as IND-CPA) or semantic security (which is related to simulation type security proofs). Intuitively, IND-CPA ensures that $\mathcal{A}$ cannot infer information about messages from ciphertexts because every ciphertext is with (almost) equal probability an encryption of a given message. By increasing κ, this status can be ensured against any $\mathcal{A}$ with the characteristics from above.

Both, ElGamal (1985) and Paillier (1999) are IND-CPA under the assumption that certain, well studied computational problems are hard (non-polynomial complexity). More precisely, Paillier (1999) builds on the decisional composite residuosity assumption whereas ElGamal (1985) is based on the decisional Diffie–Hellman assumption. See Cramer, Damgård, and Nielsen (2015) or Katz and Lindell (2020) for a more comprehensive introduction on security guarantees of cryptosystems.

2.1.3. Data encoding

Generally speaking, different cryptosystems rely on different realizations of message and ciphertext spaces $\mathcal{M}$ and $\mathcal{C}$, respectively. However, both $\mathcal{M}$ and $\mathcal{C}$ are typically finite sets. For instance, $\mathcal{M} := \mathbb{Z}_q$ with the standard representatives $\{0, 1, \ldots, q-1\}$ (and the corresponding modulo reduction $z \bmod q := z - \lfloor z/q \rfloor q$) is found in Paillier (1999) and many SMPC protocols (see Section 4). Using finite $\mathcal{M}$ and $\mathcal{C}$ has several advantages in the context of encryption, with security being a major consideration. In fact, finite sets allow the construction of encryption procedures which output almost uniformly distributed ciphertexts regardless of the input message. Moreover, finite integer sets brought forth many hard problems that can serve as a basis for a cryptosystem. In addition to the above, well-known examples include integer factorization, discrete logarithm, or the quadratic residuosity problem. Furthermore, integers are realized in the same way across (practically all) machines, which ensures reliability.

While message spaces such as $\mathcal{M} = \mathbb{Z}_q$ are advantageous from a cryptographic point of view, encrypting real-valued data, which is omnipresent in control and other domains, is not immediately supported. Thus, given a scalar $x \in \mathbb{R}$, an encoding of x into a message $z \in \mathbb{Z}_q$ is needed. Two approaches are evident to realize such an encoding. Namely, as a first step, x is either approximated with a fixed-point or floating-point number. Due to the finite size, one can associate the approximation with integers in $\mathbb{Z}_q$. Clearly, floats provide higher robustness and flexibility, which makes them the unspoken standard for numerical computations. However, their data-dependent exponent hinders an efficient implementation of encrypted additions (and subtractions). Aiming for fast encrypted operations, this inefficiency outweighs the benefits of floating point numbers. Hence, most applications of HE utilize fixed-point encodings, and we follow this trend here.

A straightforward way to realize (generalized) fixed-point encodings suitable for encrypted control is as follows. One specifies a (public) scaling factor $s \geq 1$ and defines the encoding by

$$z := \lceil sx \rfloor \bmod q. \tag{4}$$

[2] See Schulze Darup et al. (2021, p. 65) for a compact summary of the scheme.

Clearly, due to the rounding and the modulo operation, an exact decoding is usually intractable. However,

$$\mu(z) := z - (z \geq q/2)\, q \tag{5}$$

reflects a partial inverse of the modulo operation. In fact, $z = \mu(z \bmod q)$ for all $z \in \mathcal{Z}_q := [-\lfloor q/2 \rfloor, \lceil q/2 \rceil - 1] \cap \mathbb{Z}$. Hence, we find $x \approx \mu(z)/s$ for z as in (4) and all x satisfying $\lceil sx \rfloor \in \mathcal{Z}_q$. More precisely, for all such x, we easily deduce the decoding error $|x - \mu(z)/s| \leq 1/(2s)$. Importantly, this error becomes very large (at least q/s) whenever $\lceil sx \rfloor \notin \mathcal{Z}_q$ as a result of an overflow. Avoiding such overflows is crucial, since it will be impossible to detect them during subsequent encrypted computations. Clearly, overflows are less likely for small s and large q. However, small s may lead to inaccuracy and large q slightly increase the computational effort (since we often have to deal with numbers significantly larger than, say, 32 or 64 bit, which are common processor-native word sizes).

2.2. *Encrypted controllers with affine structures*

Given the partial homomorphisms and the data encoding from Sections 2.1.1 and 2.1.3, respectively, encrypting simple controllers becomes straightforward. In fact, it is easy to see that additively HE allows to carry out linear as well as affine transformations in an encrypted manner using the operations "$\oplus$" and "$\odot$". Clearly, this is all we need to encrypt static linear output (or state) feedback of the form

$$\boldsymbol{u}(k) = \boldsymbol{K}\boldsymbol{y}(k), \tag{6}$$

where $k \in \mathbb{N}$ denotes a time-step and $\boldsymbol{K} \in \mathbb{R}^{m \times l}$ the controller gains, whereas $\boldsymbol{u}(k)$ and $\boldsymbol{y}(k)$ stand for the control input and the system's output, respectively. This has first been observed by Kogiso and Fujita (2015) although they used the multiplicatively homomorphic ElGamal scheme to encrypt (6). After this pioneering work, many papers presented encrypted versions of various controller types. While the considered types were as diverse as predictive or distributed control, most early works build on the identification and encryption of affine controller structures. We associate this feature with the first generation of encrypted controllers and provide a summary of some illustrative works next.

2.2.1. *Static linear feedback*

The encryption of (6) has also been addressed in Farokhi, Shames, and Batterham (2016) and Fujita, Kogiso, Sawada, and Shin (2015), where the latter work makes use of the additively homomorphic Paillier scheme. Regarding the setup, both works (and Kogiso & Fujita, 2015) consider a cloud-based implementation of (6), where the cloud is assumed to be an honest-but-curios (or semi-honest) adversary. That is, the cloud will faithfully execute uploaded algorithms, but it may try to gather additional information or insights from the provided data. Such an assumption for the cloud is standard in cryptography. The encrypted realizations then aim for keeping the system outputs $\boldsymbol{y}(k)$ and control inputs $\boldsymbol{u}(k)$ private, whereas (a scaled version of) the controller matrix $\boldsymbol{K}$ is assumed to be known to the cloud.

In order to prepare the encrypted implementation, we initially apply the integer encoding from Section 2.1.3. Scaling of the controller matrix and system outputs in combination with the modulo reduction then leads to the transformed control law

$$\boldsymbol{v}(k) := \lceil s\boldsymbol{K} \rfloor \lceil s\boldsymbol{y}(k) \rfloor \bmod q. \tag{7}$$

The approximate control inputs can be recovered from $\boldsymbol{v}(k)$ via $\boldsymbol{u}(k) \approx \mu(\boldsymbol{v}(k))/s^2$ as long as $\lceil s\boldsymbol{K} \rfloor \lceil s\boldsymbol{y}(k) \rfloor \in \mathcal{Z}_q^m$. Based on (7), an encrypted implementation using additively HE is straightforward. To see this, we first note that

$$\lceil s\boldsymbol{K} \rfloor \lceil s\boldsymbol{y}(k) \rfloor = (\lceil s\boldsymbol{K} \rfloor \bmod q)(\lceil s\boldsymbol{y}(k) \rfloor \bmod q) \pmod q$$

according to standard modulo arithmetic. In each time-step, $\boldsymbol{y}(k)$ can now be measured, scaled, rounded, and encrypted at the (smart) sensor (see Fig. 1.b). The resulting ciphertexts $\mathbf{ct}(\lceil s\boldsymbol{y}(k) \rfloor \bmod q)$ are subsequently sent to the cloud, where encryptions of $\boldsymbol{v}(k)$ are computed via

$$\mathsf{ct}(v_i(k)) = \bigoplus_{j=1}^{l} \left(\lceil sK_{ij} \rfloor \bmod q \right) \odot \mathsf{ct}\left(\lceil sy_j(k) \rfloor \bmod q \right)$$

with "$\bigoplus$" reflecting the sum-version of "$\oplus$". The results are sent to the actuator, where a decryption, modulo inversion, and rescaling according to

$$\mu\left(\mathsf{Dec}_{\mathsf{sk}}\left(\mathbf{ct}(\boldsymbol{v}(k))\right)\right)/s^2$$

leads to the desired control actions. Apart from the encrypted implementation itself, Farokhi et al. (2016) discuss robustness guarantees despite quantization errors resulting from the integer encoding. Moreover, they point out that one could also achieve privacy of $\boldsymbol{K}$ if $\boldsymbol{y}(k)$ is not confidential by swapping the roles of the factors $\lceil s\boldsymbol{K}_{ij} \rfloor$ and $\lceil s\boldsymbol{y}_j \rfloor$ within the partially encrypted multiplications. Finally, Lin, Farokhi, Shames, and Nešić (2018) investigate the application of encrypted linear feedback to nonlinear systems.

2.2.2. *Model predictive control*

Another more advanced type of controller, which has been considered early on for an encrypted implementation using additively HE, is model predictive control (MPC). Central to classical MPC is the recurring solution of the optimal control problem

$$\begin{aligned} \min_{\substack{\tilde{\boldsymbol{x}}(0),\dots,\tilde{\boldsymbol{x}}(p) \\ \tilde{\boldsymbol{u}}(0),\dots,\tilde{\boldsymbol{u}}(p-1)}} \quad & \|\tilde{\boldsymbol{x}}(p)\|_{\boldsymbol{S}}^2 + \sum_{\tau=0}^{p-1} \|\tilde{\boldsymbol{x}}(\tau)\|_{\boldsymbol{Q}}^2 + \|\tilde{\boldsymbol{u}}(\tau)\|_{\boldsymbol{R}}^2 \\ \text{s.t.} \quad & \tilde{\boldsymbol{x}}(0) = \boldsymbol{x}(k),\ \tilde{\boldsymbol{x}}(p) \in \mathcal{T}, \\ & \tilde{\boldsymbol{x}}(\tau+1) = \boldsymbol{A}\,\tilde{\boldsymbol{x}}(\tau) + \boldsymbol{B}\tilde{\boldsymbol{u}}(\tau), \quad \forall \tau \in \{0,\dots,p-1\}, \\ & (\tilde{\boldsymbol{x}}(\tau), \tilde{\boldsymbol{u}}(\tau)) \in \mathcal{X} \times \mathcal{U}, \quad \forall \tau \in \{0,\dots,p-1\} \end{aligned} \tag{8}$$

in every time-step for the current state $\boldsymbol{x}(k)$. Here, $p \in \mathbb{N}$ is the prediction horizon, $\boldsymbol{A}, \boldsymbol{B}$ reflect the model dynamics, $\boldsymbol{S}, \boldsymbol{Q}, \boldsymbol{R}$ are weighting matrices, and the sets $\mathcal{U}, \mathcal{X}, \mathcal{T}$ describe input, state, and terminal constraints, respectively. The solution of (8) then determines the control action via $\boldsymbol{u}(k) = \tilde{\boldsymbol{u}}(0)$. For more details on MPC, we refer to Rawlings, Mayne, and Diehl (2017).

Now, it might seem surprising that encrypted MPC has been considered so early, since the involved optimal control problem makes it significantly more complex than linear feedback and since the affine structures (required for an effective implementation via PHE) may not be obvious. However, under the standard assumptions that $\boldsymbol{S}, \boldsymbol{Q}, \boldsymbol{R}$ are positive definite and that $\mathcal{U}, \mathcal{X}, \mathcal{T}$ are convex polyhedrons, it has been shown by Bemporad, Morari, Dua, and Pistikopoulos (2002) that the optimal control law $\boldsymbol{g}$ is piecewise affine (PWA), i.e., of the form

$$\boldsymbol{g}(\boldsymbol{x}) = \begin{cases} \boldsymbol{K}_1 \boldsymbol{x} + \boldsymbol{w}_1 & \text{if} \quad \boldsymbol{x} \in \mathcal{P}_1, \\ \quad\vdots & \qquad\quad \vdots \\ \boldsymbol{K}_\theta \boldsymbol{x} + \boldsymbol{w}_\theta & \text{if} \quad \boldsymbol{x} \in \mathcal{P}_\theta \end{cases} \tag{9}$$

with polyhedral regions $\mathcal{P}_i \subseteq \mathcal{X}$. Encrypting the affine segments can be carried out analogously to the linear feedback case in Section 2.2.1. In fact, we only have to provide $\mathbf{ct}(\lceil s^2 \boldsymbol{w}_i \rfloor \bmod q)$ to the cloud and add this via "$\oplus$" to $\mathbf{ct}(\lceil s\boldsymbol{K}_i \rfloor \lceil s\boldsymbol{x} \rfloor \bmod q)$. The remaining issue is, however, to select the correct segment i, satisfying $\boldsymbol{x}(k) \in \mathcal{P}_i$. In Schulze Darup, Redder, Shames, Farokhi, and Quevedo (2018), this step is outsourced to the sensor since an encrypted selection of i is quite challenging due to the involved conditional expressions. Then, in each time-step, i is either provided to the cloud, where the corresponding encrypted segment is evaluated, or to the actuator, which requires the cloud to evaluate all segments and forward all intermediate results to the actuator, where only the relevant one is decrypted (see Schulze Darup et al. (2021, Fig. 3.a) for an illustration). While the setup is obviously suboptimal, it provided a proof of concept that encrypted MPC is possible via its explicit solution.

Notably, PHE-based implementations of MPC have also been proposed for the implicit, i.e., solver-based solution, of (8). An early work in this regard is due to Shoukry et al. (2016), who considered the more general case of privacy-preserving quadratic optimization. Their work as well as the more MPC-specific extension by Alexandru, Morari, and Pappas (2018) build on an encrypted gradient ascent method applied to the dual of (8), which is a quadratic program of the form

$$\max_{\lambda} -\frac{1}{2}\lambda^\top \boldsymbol{H}_d \lambda - (\boldsymbol{F}_d \boldsymbol{x}(k) + \boldsymbol{c}_d)^\top \lambda \quad \text{s.t.} \quad \lambda \geq \boldsymbol{0}. \tag{10}$$

Clearly, the gradient to the objective in (10) is affine in λ such that a projected gradient step leads to

$$\lambda(k+1) = \max\{\boldsymbol{0}, \lambda(k) - \eta\left(\boldsymbol{H}_d \lambda(k) + \boldsymbol{F}_d \boldsymbol{x}(k) + \boldsymbol{c}_d\right)\}, \tag{11}$$

where η denotes the step width and where the max-expression, which is understood element-wise, reflects the projection onto the positive orthant. Remarkably, after convergence to the optimal Lagrange multipliers λ^*, one can easily reconstruct the optimal control action $\tilde{\boldsymbol{u}}^*(0)$ according to Boyd and Vandenberghe (2004, Chap. 5). However, one faces the problem of evaluating the (non-polynomial) max-expression when it comes to an encryption of (11). An additional and more subtle problem is related to the evaluation of (11) over $\mathbb{Z}_q$, where expressions like $\lfloor s\eta \boldsymbol{H}_d \rceil \lfloor s\lambda(k) \rceil \approx s^2 \eta \boldsymbol{H}_d \lambda(k)$ occur. Such multiplications increase the scaling associated with $\lambda(k)$ in every iteration. Without countermeasures, this will (almost certainly) lead to an overflow of $\mathbb{Z}_q$, which would cause a large and possibly fatal error.

In order to address both problem, Alexandru et al. (2018) and Shoukry et al. (2016) use re-encryptions with help of a third party.[3] While the former work employs a (real number) multiplicative blinding, the latter addresses the resulting security issues by additive blinding, i.e, $z + a$ (without modulo) where $z \in \mathcal{M}$, a is selected uniformly at random from $(0, \max\{\mathcal{M}\} + 2^\kappa)$, and κ is the security parameter. This allows for a formal security proof based on (statistical) indistinguishability ((22) in Section 4.1) from uniform randomness. Then, based on blinded data (which is not in danger of a modulo q wrap around), one can efficiently evaluate the comparisons required for (11) and reduce the scaling of $\lambda(k)$ via a division by s and subsequent rounding. These problems are analogously faced in Schulze Darup, Redder, and Quevedo (2018), where real-time iterations based on (primal) projected gradient descent for an MPC with input constraints are considered.

2.2.3. Distributed control

Affine structures can also be exploited for encrypted distributed control. Here, we consider multi-agent systems, which intend to solve a cooperative control task in a decentralized manner. For M agents with linear system dynamics and quadratic objectives (which may, e.g., reflect flocking), one can design structured linear feedback of the form

$$\begin{pmatrix} \boldsymbol{u}_1(k) \\ \vdots \\ \boldsymbol{u}_M(k) \end{pmatrix} = \begin{pmatrix} \boldsymbol{K}_{11} & \cdots & \boldsymbol{K}_{1M} \\ \vdots & \ddots & \vdots \\ \boldsymbol{K}_{M1} & \cdots & \boldsymbol{K}_{MM} \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_1(k) \\ \vdots \\ \boldsymbol{x}_M(k) \end{pmatrix},$$

where the block $\boldsymbol{K}_{ij}$ is constrained to form a zero-matrix whenever the agents i and j are not communicating (Lin, Fardad, & Jovanovic, 2011). In other words, the local control laws can be formulated as

$$\boldsymbol{u}_i(k) = \boldsymbol{K}_{ii} \boldsymbol{x}_i(k) + \sum_{j \in \mathcal{N}_i} \boldsymbol{K}_{ij} \boldsymbol{x}_j(k) \tag{12}$$

with $\mathcal{N}_i$ denoting the set of neighbors of agent i. An encrypted implementation of (12) has first been proposed in Schulze Darup, Redder, and Quevedo (2019) under the assumption of honest-but-curious agents. Then, the implementation aims for preserving the privacy of the individual states $\boldsymbol{x}_i(k)$. To this end, ciphertexts $\mathbf{ct}(\lfloor s\boldsymbol{K}_{ij} \rceil \mod q)$ encrypted using a secret key sk_i are made available to agent j. During runtime, agent j computes an encrypted version of $\boldsymbol{K}_{ij}\boldsymbol{x}_j(k)$ via the operations "$\odot$" and "$\oplus$" and sends the result to agent i, where it is decrypted before evaluating (12). Clearly, agent i may still be able to infer information on $\boldsymbol{x}_j(k)$ based on $\boldsymbol{v}(k) := \lfloor s\boldsymbol{K}_{ij} \rceil \lfloor s\boldsymbol{x}_j(k) \rceil \mod q$ (by solving a certain observability problem). However, this issue can be mitigated, to some extent, by implementing the improved scheme in Alexandru, Schulze Darup, and Pappas (2019).

Another stream of research deals with PHE-based privacy in consensus algorithms. These are structurally similar to (12) but recursive in nature and, e.g., of the form

$$\boldsymbol{x}_i(k+1) = \boldsymbol{x}_i(k) + \varepsilon \sum_{j \in \mathcal{N}_i} a_{ij}(\boldsymbol{x}_j(k) - \boldsymbol{x}_i(k)). \tag{13}$$

Still, approaches similar to the one above are used to guarantee privacy between the agents. In fact, the iteration issue is again resolved by the use of a trusted third party in Hadjicostis (2018), Hadjicostis and Domínguez-García (2020), Kishida (2018) and Ruan, Gao, and Wang (2019). Two additional works propose slightly different strategies. First, Zhang, Li, Wang, Louati, and Chen (2022) considers a decomposition of each participant's state into a private and exchangeable part. Second, Schlüter, Binfet, Kim, and Schulze Darup (2022) addresses the iterations by means of resets that do not rely on a trusted-third party.

3. Second generation using leveled FHE

While the variety of encrypted controllers from the first generation shows that PHE combined with tailored controller reformulations provides a solid framework for encrypted control, recurring challenges such as not natively supported operations or iterations firmly ask for more flexible and powerful cryptosystems. As indicated in the introduction and in Section 2, suitable cryptosystems exist in the form of (leveled) FHE and SMPC. However, the hurdle of applying them in the context of encrypted control is significantly higher than with PHE (even though applying PHE itself is not trivial). Two reasons are dominant in this context. First, (leveled) FHE and SMPC schemes are typically significantly more complex than PHE schemes such as Paillier or ElGamal, as they often rely on sophisticated algebraic structures or more advanced cryptographic concepts. Second, this complexity cannot be neglected for applications since available software packages such as SEAL (Chen, Laine, & Player, 2017), HElib (Halevi & Shoup, 2020), Concrete[4] (Chillotti, Joye, Ligier, Orfila, & Tap, 2020), OpenFHE (Al Badawi et al., 2022), or MP-SPDZ (Keller, 2020) still require expert knowledge for an efficient usage.

This article intends to lower this hurdle for readers new to the field. To this end, we will not only survey the second generation of encrypted controllers, which make use of these more complex cryptosystems, but we will also provide a detailed introduction to one of the most competitive HE schemes, namely CKKS (Cheon, Kim, Kim, & Song, 2017). We combine this tailored introduction with an own implementation,[5] which we share for educational purposes.

The remaining section is organized as follows. We discuss advanced HE, which largely builds on the learning with errors (LWE) problem, in Section 3.1. Advanced encrypted controllers, which rely on (leveled) FHE, are summarized in Section 3.2. Schemes based on SMPC are addressed further below in Section 4.

[3] Due to the two involved parties, the schemes can also be considered as SMPC. We still count them towards the first generation, since the PHE parts are central from our point of view.

[4] The Rust library originally termed "Concrete" is now referred to as "TFHE-rs" by the developers.

[5] The source code can be viewed at and downloaded from https://github.com/Control-and-Cyberphysical-Systems/ckks-demo.

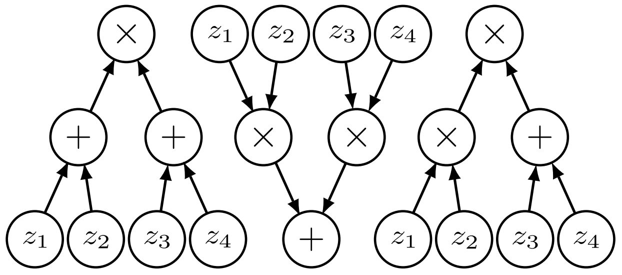

Fig. 2. Arithmetic circuits corresponding to (from left to right) the three expressions $(z_1+z_2)(z_3+z_4)$, $(z_1z_2)+(z_3z_4)$, and $(z_1z_2)(z_3+z_4)$. The multiplicative depth of the circuits is 1, 1, and 2, respectively.

3.1. *Advanced homomorphic encryption*

As briefly discussed in the beginning of Section 2, (leveled) FHE mainly differs from PHE schemes in that not only encrypted additions *or* multiplications are available but both (or comparable gates for binary cryptosystems). Still, counting Rivest et al. (1978) as the first PHE scheme, it took more than three decades until Gentry (2009) proposed the first FHE scheme. At this point, we have to detail a feature that we only briefly touched upon before. In fact, for a proper FHE scheme, it is required that arithmetic (or Boolean) circuits of arbitrary depth can be securely evaluated. If only a finite depth is supported, we call a scheme *leveled* FHE, where the level typically refers to the maximum multiplicative depth of supported arithmetic circuits (i.e., the maximum number of multiplications along a branch in the corresponding tree structure as illustrated in Fig. 2). As specified below, the main reason of a finite number of available levels is noise accumulation during encrypted operations. In fact, we will see in Section 3.1.1 that noise is a crucial element for most (leveled) FHE schemes.

Interestingly, leveled FHE can serve as a basis for proper FHE schemes if their supported levels allow for a procedure called *bootstrapping*, which is used to reduce noise and "refresh" ciphertexts. For different cryptosystems, bootstrapping can take different form. For instance, in CKKS, bootstrapping increases the ciphertext modulus which is decreasing during encrypted computations (as specified later in Sidebar 5). As a result, further encrypted operations are enabled. Hence, carrying out bootstrapping regularly allows evaluating circuits of arbitrary depth since unregulated noise accumulation is avoided.

Unfortunately, existing bootstrapping procedures for arithmetic schemes such as BFV (Fan & Vercauteren, 2012), BGV (Brakerski, Gentry, & Vaikuntanathan, 2014), and CKKS (Cheon et al., 2017) are computationally costly. Still, cryptosystems with efficient bootstrapping routines exist, e.g., FHEW/DM (Ducas & Micciancio, 2015) and TFHE/CGGI (Chillotti, Gama, Georgieva, & Izabachéne, 2020), but they are restricted to binary (or small integer) messages. In encrypted control, we typically rely on arithmetic operations with messages of moderate size (e.g., 16 or 32-bit numbers), bootstrapping will usually interfere with real-time requirements, and a finite multiplicative depth is often acceptable. As a consequence, we will concentrate on leveled FHE supporting arithmetic operations in the following. Nonetheless, we will briefly discuss potential use cases of schemes such as TFHE/CGGI for encrypted control in Section 5.1.1.

3.1.1. *The learning with errors problem*

The previous discussion shows that there is a huge variety of (leveled) FHE schemes. However, while different in style, they commonly build on variants of the so-called learning with errors problem (LWE) introduced by Regev (2005). To generate an instance of the problem, the first step is to sample a secret vector **sk** uniformly at random from $\mathcal{Z}_q^N$ (but other distributions are possible). We note at this point that $\mathcal{Z}_q$ (introduced in 2.1.3) is another representation of $\mathbb{Z}_q$. In order to use this representation, which is beneficial in the context of LWE, we have to redefine the modulo reduction and the partial inverse from (5) according to $z \bmod q := z - \lfloor z/q + 1/2 \rfloor\, q$ and $\mu(z) := z$, respectively, in order to obtain the same features as for the representation $\{0, 1, \dots, q-1\}$ above. We further note that the secret vector is denoted as **sk**, since it will later act as a secret key. Next, imagine an oracle from which you can receive tuples $(\boldsymbol{a}^\top, b)$, constructed as follows. The vector $\boldsymbol{a}$ is likewise sampled uniformly at random from $\mathcal{Z}_q^N$ and $b := \boldsymbol{a}^\top \mathbf{sk} + e \bmod q$ with the (small) error e being sampled from a discrete Gaussian distribution[6] $\mathcal{D}_\sigma$ over $\mathbb{Z}$ with standard deviation σ. Now, the fundamental LWE assumption is that, given an arbitrary number of tuples $(\boldsymbol{a}_i^\top, b_i)$, it is practically intractable to find **sk**.

At first sight, this assumption might seem questionable (especially to control engineers). In fact, it might be tempting to approach a solution with methods from linear algebra or optimization. In order to understand, why such approaches are not expedient, we applied them to an LWE instance in Sidebar 1 (Albrecht et al. (2018), Blum et al. (2003), Hassibi and Vikalo (2002)). In short, the hardness of LWE comes from the fact that it is defined over a lattice $\mathcal{Z}_q^N$ instead of $\mathbb{R}^N$. Of course, just showing that several algorithms fail does not replace a rigorous proof of LWE's hardness, which is, to some extent, an open problem. Nonetheless, let us summarize several reasons why LWE is widely believed to be hard (Regev, 2010). First, and most importantly, LWE has been shown to be as hard as worst-case lattice problems, which are believed to be NP-hard, by Regev (2009). Second, despite being "in the wild" for almost two decades and attracting significant attention, the best-known algorithms for LWE operate in exponential time, and even quantum algorithms do not appear to offer improvements (Albrecht et al., 2015). Third, LWE is a natural extension of the learning parity with noise problem, which can be expressed as the task of decoding from random linear binary codes. Consequently, advancements on LWE could lead to significant breakthroughs in coding theory.

3.1.2. *A basic LWE cryptosystem*

We next show how the LWE problem can be used to design a cryptosystem. For simplicity, we will discuss a (symmetric) secret key scheme because public key variants comprise a slightly more complex encryption. This scheme serves as an introduction, but also as an intermediate step towards the more complex CKKS scheme summarized in Section 3.1.3. Now, to set up the scheme, the secret key **sk** is sampled from $\mathcal{Z}_q^N$ as above. To encrypt a message z, a fresh LWE tuple $(\boldsymbol{a}^\top, b)$ is generated, and the encryption is performed via

$$\mathbf{ct}(z) = \mathsf{Enc}_{\mathbf{sk}}(z) := (\boldsymbol{a}^\top, b + z) \bmod q = (\boldsymbol{a}^\top, b). \tag{14}$$

Thus, within the ciphertext $\mathbf{ct}(z) \in \mathcal{Z}_q^{1\times(N+1)}$, the message z is buried in an LWE instance. The decryption is the inner product of $\mathbf{ct}(z)$ and $(-\mathbf{sk}, 1)$ leading to

$$\mathsf{Dec}_{\mathbf{sk}}(\mathbf{ct}(z)) = b - \boldsymbol{a}^\top \mathbf{sk} \bmod q = z + e \bmod q. \tag{15}$$

Evidently, the decryption returns the message perturbed by the error e. Since the errors are small, this is often acceptable as exploited in the CKKS cryptosystem. Alternatively, if exact decryptions are desired, one can encrypt δz instead of z using some large $\delta \in \mathbb{N}$, divide the outcome of (15) by δ, and round the result to the nearest integer. This procedure leads to the exact z given that $\delta > 2|e|$ (and that overflows are excluded). Such an approach is for instance used in the BFV and FHEW/DM cryptosystems (Ducas & Micciancio, 2015; Fan & Vercauteren, 2012), respectively.

Because the ciphertexts $\mathbf{ct}(z_1) = (\boldsymbol{a}_1^\top, b_1)$ and $\mathbf{ct}(z_2) = (\boldsymbol{a}_2^\top, b_2)$ are linear in the key **sk** and the plaintexts, encrypted additions are realized by (literally) adding the ciphertexts according to

$$\mathbf{ct}(z_1) \oplus \mathbf{ct}(z_2) = (\boldsymbol{a}_1^\top + \boldsymbol{a}_2^\top, b_1 + b_2) \bmod q.$$

[6] Alternatively, e is sampled from a bounded uniform distribution, although Gaussian distributions are allegedly more popular.

Sidebar 1: An intuition about the hardness of LWE

Let us discuss some seemingly promising solution approaches for LWE instances. To this end, tuples $(\boldsymbol{a}_i^\top, b_i)$ are given, which can be concatenated (row-wise) into $(\boldsymbol{A}_{\text{LWE}}, \boldsymbol{b})$. Note that a guess for $\mathbf{sk}$ can be verified because a correct guess leads to a small $\boldsymbol{b} - \boldsymbol{A}_{\text{LWE}}\mathbf{sk} \bmod q$ as a result of the small errors e_i (e.g., $e_i \in \{-1, 0, 1\}$). For a numeric example and more details, see Regev (2010, Sect. 1).
Gaussian elimination. First, an assumption about the errors e_i in each of the equations $b_i = \boldsymbol{a}_i^\top \mathbf{sk} + e_i \pmod q$ is required. To this end, we apply the certainty equivalence principle (ignoring e_i) and build the linear equation system $\boldsymbol{A}_{\text{LWE},1:N}\mathbf{sk} = \boldsymbol{b}_{1:N} \pmod q$ using N tuples. Then, Gaussian elimination linearly combines the equations, where division is replaced by the multiplicative inverse modulo q over the field $\mathcal{Z}_q$. However, combining equations amplifies the error, such that ignoring it is hopeless for sufficiently large N. Besides, one may presume $e_i \in \{-1, 0, 1\}$ and enumerate all 3^N combinations which also fails spectacularly for large enough N (and σ).
Least-squares. For the solution of (standard) linear equation systems with errors, a least-squares formulation is well-suited. Thus, we consider $\arg\min_{\mathbf{sk}\in\mathcal{Z}_q^N} \|\boldsymbol{b} - \boldsymbol{A}_{\text{LWE}}\mathbf{sk}\|_2^2$ as our next approach. In fact, the resulting $\mathbf{sk}$ is the (with high probability unique and) correct secret. The only problem is that solving integer least-square problems is NP-hard (see Hassibi and Vikalo (2002) and references therein) such that large N (and q) makes a solution intractable.
Combinations. Another solution builds on constructing tuples that allow for a simpler treatment (Blum, Kalai, & Wasserman, 2003). For instance, the linear combination $\sum_i \alpha_i \boldsymbol{a}_i = (\gamma, 0, \ldots, 0)^\top$ allows for the approximation $\mathrm{sk}_1 = \gamma^{-1}\sum_i \alpha_i(b_i - e_i) \approx \gamma^{-1}\sum_i \alpha_i b_i$ if $\sum_i \alpha_i b_i \gg \sum_i \alpha_i e_i$. Then, one proceeds similarly for the remaining components of $\mathbf{sk}$. The issues with this approach are twofold. On the one hand, it requires exponential time to obtain suitable samples. On the other hand, the secret is somewhat sensitive to perturbation.
From our short exposition, we learn that the main security parameter is N. We also note that the "signal-to-noise ratio" q/e must be small for a high security (noise-free LWE is trivial). Moreover, since several $(\boldsymbol{a}_i^\top, b_i)$ can be linearly combined to "new" samples with a slightly larger error, the hardness of LWE is essentially independent of the number of samples. In practice, LWE-based schemes come with a predetermined choice for σ. Then, one selects q based on the computations that have to be evaluated homomorphically and increases N such that a desired security threshold is exceeded. Typically, this leads to at least $N \geq 128$. Note that small N should be treated with care. In this context, the LWE estimator (Albrecht, Player, & Scott, 2015) is helpful because state-of-the-art attacks are considered. More conservative recommendations can be found in Albrecht et al. (2018).

The correctness can easily be verified by noting that

$$\mathsf{Dec}_{\mathsf{sk}}(\mathbf{ct}(z_1) \oplus \mathbf{ct}(z_2)) = z_1 + z_2 + e_1 + e_2 \bmod q.$$

Analogously, it is straightforward to show that partially encrypted additions of and multiplications by a public constant $c \in \mathcal{Z}_q$ are enabled via

$$c \boxplus \mathbf{ct}(z) = \mathbf{ct}(c + z) = (\boldsymbol{a}^\top, c + b) \bmod q \quad \text{and}$$
$$c \odot \mathbf{ct}(z) = \mathbf{ct}(cz) \quad = (c\boldsymbol{a}^\top, cb) \bmod q,$$

respectively. So far, the scheme does not provide more operations than Paillier. In fact, it is easy to see that also Paillier supports partially encrypted additions (in addition to "$\oplus$" and "$\odot$"). Nevertheless, the presented scheme can also support encrypted multiplications. However, realizations are more complex, and we postpone them to the introduction of CKKS below. Instead, we briefly comment on the security of the scheme at hand.

Regarding the security, mainly two effects play a role. First, $\boldsymbol{a}^\top\mathbf{sk} \bmod q$ results in almost uniformly random numbers in $\mathcal{Z}_q$ for suitable (N, q) such that $b = \boldsymbol{a}^\top\mathbf{sk} + e + z \bmod q$ is (almost) uniformly random regardless of the message z. From an attacker's perspective, b could be an encryption of any $z \in \mathcal{M}$ and, thus, does not reveal information. Another viewpoint would be that an attacker is not able to distinguish b from (truly) uniform randomness, which (of course) does not contain information about the message. However, an attacker does not only obtain b but $\mathbf{ct}(z) = (\boldsymbol{a}^\top, b)$. Then, a chosen-plaintext attacker, who has access to ciphertexts $\mathbf{ct}(z)$ and corresponding plaintexts, can readily compute $b - z = \boldsymbol{a}^\top\mathbf{sk} + e \bmod q$. Consequently, a chosen-plaintext attacker is faced with solving an LWE instance for $\mathbf{sk}$, resulting in an IND-CPA construction as defined in Section 2.1.2. For a rigorous security proof, we refer to the literature (Regev, 2009). Remarkably, LWE-based cryptosystems offer a very simple and structure-preserving encryption while guaranteeing security based on the LWE assumption.

3.1.3. Transitioning to CKKS

We are now ready to provide a step-by-step introduction to the CKKS cryptosystem (Cheon et al., 2017), which is often considered state-of-the-art when it comes to encrypted arithmetic and which also provides the basis of many encrypted controllers. As a first step, we note that there exists a ring variant (RLWE) of the LWE problem that enables a more efficient implementation. That is, instead of considering elements in $\mathcal{Z}_q$, quotient rings $\mathcal{R}_q = \mathcal{Z}_q[X]/(X^N + 1)$ are utilized (see Lyubashevsky, Peikert, & Regev, 2013) that are detailed in Sidebar 2. Consequently, ciphertexts become polynomials with integer coefficients and a fixed order. Fortunately, the modification does not degrade the security, as there are no better attacks available against RLWE than against LWE (Marcolla et al., 2022).

The incentive to use such an algebraic abstraction comes from the possibility to encrypt N messages (compared to one as in (14)) simultaneously in the coefficients of a single polynomial and a reduction in the multiplication complexity from approximately $\mathcal{O}(N^2)$ to $\mathcal{O}(N\log(N))$ via a number theoretic transformation (a generalization of a discrete Fourier transformation to finite fields (Pollard, 1971)). Finally, so-called residue number variants exist, where the ring $\mathcal{R}_q$ is decomposed into smaller rings $\mathcal{R}_{q_i}$ with $q_i < q$, in which more efficient and independent computations are possible (Cheon, Han, Kim, Kim, & Song, 2019).

Next, reconsider the basic LWE scheme from Section 3.1.2 and note that it is straightforwardly converted into an RLWE scheme by reinterpreting the vectors $\boldsymbol{a}, \mathbf{sk} \in \mathcal{Z}_q^N$ as the coefficients of the polynomials $a, \mathrm{sk} \in \mathcal{R}_q$ and drawing each coefficient of $e \in \mathcal{R}_q$ from a discrete Gaussian distribution. With this at hand, a ciphertext of $z \in \mathcal{R}_q$ becomes $\mathrm{ct}(z) = (a, b) \in \mathcal{R}_q^2$, where $b = a\,\mathrm{sk} + e + z \bmod q$. It is readily shown that the decryption (15), encrypted addition, and public operations work analogously. However, a slight adaption is made in the context of CKKS. Instead of drawing the coefficients of sk from $\mathcal{Z}_q^N$ one uses (ternary values) $\mathrm{sk} \in \{-1, 0, 1\}^N$. Here, we note that more efficient attacks on ternary secrets are available (especially when they are sparse (Cheon, Hhan, Hong, & Son, 2019; Curtis & Player, 2019)). Nonetheless, with adequate parameter choices, solving a corresponding RLWE instance is still intractable.

With these preparations at hand, we turn our focus to encrypted multiplications, which are, due to their nonlinearity, more intricate than additions. Still, they can be realized via two different approaches in RLWE-based cryptosystems. The approach for CKKS is summarized in Sidebar 3. At this point, an attentive reader may have realized

Sidebar 2: Quotient rings

The elements of the quotient ring $\mathcal{R}_q = \mathcal{R}/q\mathcal{R} = \mathcal{Z}_q[X]/(X^N+1)$ are polynomials of the form $a(X) = \sum_{i=0}^{N-1} \tilde{a}_i X^i \in \mathcal{R}_q$, where denoting the coefficients by $\tilde{a}_i$ avoids ambiguity with the ith polynomial $a_i(X) \in \mathcal{R}_q$. Importantly, the polynomial $a(X)$ is the remainder of the division by the irreducible polynomial X^N+1 (indicated by $\mathcal{Z}[X]/(X^N+1)$), where $\tilde{a}_i$ are subsequently reduced modulo q (indicated by $\mathcal{Z}_q[X]$). For brevity, we write $a = a(X)$ in the following and indicate the reduction by X^N+1 as well as q via $a \bmod q$.

Next, $\mathcal{R}_q$ is equipped with an addition and a multiplication between $a, b \in \mathcal{R}_q$, by means of

$$a + b := \sum_{i=0}^{N-1} (\tilde{a}_i + \tilde{b}_i) X^i \bmod q \quad \text{and} \tag{16}$$

$$ab := \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (-1)^{\lfloor (i+j)/N \rfloor} \tilde{a}_i \tilde{b}_j X^{i+j \,(\mathrm{mod}\, N)} \bmod q,$$

where the latter expression is derived by using $X^N = -1$ (mod (X^N+1)) and compactly evaluates the remainder of the polynomial multiplication and subsequent division by X^N+1. Implementation-wise, $a, b \in \mathcal{R}_q$ are naturally identified with their coefficient vector, i.e., $\tilde{\boldsymbol{a}} := (\tilde{a}_0, \ldots, \tilde{a}_{N-1})^\top \in \mathcal{Z}_q^N$.

Example. Consider $(q, N) = (10, 3)$, and polynomials with the coefficients $\tilde{\boldsymbol{a}} = (1, 2, -3)^\top$ and $\tilde{\boldsymbol{b}} = (4, -5, 6)^\top$. For $a + b$ we first obtain $5 - 3X - 7X^2$ which reduces to $-5 - 3X + 3X^2 \bmod q \in \mathcal{R}_q$. Similarly, ab first evaluates to $4 + 3X + 4X^2 - 3X^3 + 2X^4$ which, after the reductions, results in $-3 + 1X + 4X^2 \bmod q \in \mathcal{R}_q$. This and more examples can be found in our code[5].

that, based on $z_1, z_2 \in \mathcal{R}_q$, it is clear that $z_1 + z_2$ is a component-wise addition, whereas $z_1 z_2$ is a (negative wrapped) convolution of the coefficient vectors $\tilde{\boldsymbol{z}}_1$ and $\tilde{\boldsymbol{z}}_2$, in the following denoted as $\tilde{\boldsymbol{z}}_1 \circledast \tilde{\boldsymbol{z}}_2$, which is rarely a desired basic operation. As a workaround, one could use only one coefficient in $\tilde{\boldsymbol{z}}_i$ and set the remaining ones to zero in order to obtain a (standard) multiplication in a straightforward manner. This way, however, the benefit of having possibly N coefficient-slots for parallel operations in $\mathcal{R}_q$ is lost.

Fortunately, there exists a more complex encoding $\mathsf{Ecd}(\boldsymbol{x})$ for vectors $\boldsymbol{x} \in \mathbb{C}^{N/2}$ such that

$$\mathsf{Ecd}(\boldsymbol{x}_1) + \mathsf{Ecd}(\boldsymbol{x}_2) = \mathsf{Ecd}(\boldsymbol{x}_1 + \boldsymbol{x}_2)$$
$$\mathsf{Ecd}(\boldsymbol{x}_1) \circledast \mathsf{Ecd}(\boldsymbol{x}_2) = \mathsf{Ecd}(\boldsymbol{x}_1 \circ \boldsymbol{x}_2).$$

In other words, it preserves component-wise addition while the convolution corresponds to a component-wise multiplication of the encoded vectors (denoted by $\circ$). Thus, this enables encrypted SIMD (Single Instruction, Multiple Data) operations, because one homomorphic operation evaluates multiple slots in parallel. Now, the main idea to realize $\mathsf{Ecd}(\cdot)$ is applying a discrete Fourier-like transformation on $\boldsymbol{x}_1, \boldsymbol{x}_2$, which then results in the desired features. This is presented in Sidebar 4. Finally, another characteristic of CKKS is a rescaling procedure. As specified in Sidebar 5, by reducing the modulus of a ciphertext this procedure enables to perform more multiplications before running the risk of an overflow.

3.2. LWE-based encrypted control

(R)LWE-based cryptosystems have been intensively used for encrypting more demanding controller types. We survey representative realizations next. In this context, it may surprise at first sight that linear dynamic controllers (as addressed in Section 3.2.1) play a major role. However, they can be seen as a prototype for the aforementioned iteration problem. Hence, efficient implementations of these controllers are of special interest also for more complex iteration-based controllers. Moreover, we briefly consider LWE-based encryptions of estimators and data-driven controllers in Section 3.2.2.

3.2.1. Linear dynamic controllers

Linear dynamic (output feedback) controllers can be written in the form

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{y}(k) \tag{17a}$$
$$\boldsymbol{u}(k) = \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{y}(k), \tag{17b}$$

where $\boldsymbol{x}(k) \in \mathbb{R}^n$ now reflects the controller's state. This controller type is of practical interest since, e.g., PID controllers and observer-based feedback can be cast into (17). Moreover, they provide a natural extension to static linear feedback for which efficient design methods exist (Scherer & Weiland, 2015, Chap. 4). Now, in order to encrypt (17), an integer variant is required, where

$$\boldsymbol{z}(k+1) = \lceil s\boldsymbol{A} \rfloor\, \boldsymbol{z}(k) + \lceil s\boldsymbol{B} \rfloor\, \lceil s^{k+1} \boldsymbol{y}(k) \rfloor \bmod q \tag{18}$$
$$\boldsymbol{v}(k) = \lceil s\boldsymbol{C} \rfloor\, \boldsymbol{z}(k) + \lceil s\boldsymbol{D} \rfloor\, \lceil s^{k+1} \boldsymbol{y}(k) \rfloor \bmod q \tag{19}$$

with $\boldsymbol{z}(k) = \lceil s\boldsymbol{x}(k) \rfloor \bmod q$, is a possible choice. Here, the reconstruction $\boldsymbol{u}(k) \approx \boldsymbol{v}(k)/s^{k+2}$ holds as long as $\lceil s\boldsymbol{C} \rfloor\, \boldsymbol{z}(k) + \lceil s\boldsymbol{D} \rfloor\, \lceil s^{k+1}\boldsymbol{y}(k) \rfloor \in \mathcal{Z}_q^m$.

In encrypted control, (18) is of special interest because it represents a fundamental problem when it comes to encrypting iterative algorithms, where the number of iterations is unknown a priori. To see this, let us focus on the autonomous part $\boldsymbol{z}(k+1) = \lceil s\boldsymbol{A} \rfloor\, \boldsymbol{z}(k)$ and observe that the scaling of $\boldsymbol{z}(k)$ increases with every iteration. Thus, without countermeasures, an overflow of $\mathcal{Z}_q$ will almost certainly occur after a finite number of time-steps. In this case, the computed control input $\boldsymbol{v}(k)/s^{k+2}$ would be (highly) erroneous, putting the control performance and closed-loop stability at risk.

Several ways to deal with this problem, without (heavily) relying on a third party, have been proposed. They all coincide in that a reset of the scaling factor associated with $\boldsymbol{z}(k)$ is central, but the realizations differ. Clearly, such a reset could be realized using bootstrapping. However, we already discussed in Section 3.1 that bootstrapping is usually no option for real-time critical control. Kim et al. (2016) solve this issue by considering multiple controllers in parallel, where the orchestration of the controllers is organized such that always one can provide a control action while the others are being reset via bootstrapping. Notably, Kim et al. (2016) provide an early contribution to encrypted control, but use advanced FHE. Hence, our classification in generations is not always compatible with the evolution in time. Now, instead of regularly performing bootstrapping one can also reset $\boldsymbol{z}(k)$ itself to a predefined value, e.g., $\boldsymbol{0}$, and newly assign the scaling, which is proposed in Murguia, Farokhi, and Shames (2020). However, a reset often causes a large deviation from the controller's converged state such that a performance loss is encountered.

Next, one may observe that $\boldsymbol{A} \in \mathcal{Z}_q^{n\times n}$ circumvents overflows of $\mathcal{Z}_q$ since it allows choosing $s = 1$, as noted by Cheon, Han, Kim, Kim, and Shim (2018). However, Schlüter and Schulze Darup (2021) analyzed the restriction to integers and found that Schur stability then requires nilpotency. This eventually renders (17) into a finite impulse response (FIR) controller of the form

$$\boldsymbol{u}(k) = \sum_{i=0}^{n-1} \boldsymbol{F}_i \boldsymbol{y}(k-i). \tag{20}$$

Clearly, (20) avoids the overflow problem completely, because no iteration takes place, and it is "encryption-friendly" (Schlüter, Neuhaus, & Schulze Darup, 2021). For simplicity, assume $nl = N$. Then, observe that the (standard) convolution, required for (20), is evaluated in the X^{N-1} coefficient during the multiplication of ring elements in $\mathcal{R}_q$. Thus,

Sidebar 3: Encrypted multiplications in CKKS

CKKS adopts the ideas presented in Fan and Vercauteren (2012, Sect. 4), which work as follows. We start by considering the multiplied messages $z_1 z_2$ and work our way backwards to the corresponding ciphertext. Thus, we start with $z_1 z_2 \approx \mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}(z_1))\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}(z_2))$, which evaluates to

$$(b_1 - a_1\mathsf{sk})(b_2 - a_2\mathsf{sk}) = b_1 b_2 - \left(b_1 a_2 + a_1 b_2\right)\mathsf{sk} + a_1 a_2 \mathsf{sk}^2 = d_0 + d_1\mathsf{sk} + d_2\mathsf{sk}^2 \pmod q.$$

It is straightforward to compute the ciphertext $(d_0, d_1, d_2) \in \mathcal{R}_q^3$ but it has one additional element and sk^2 is required for decryption. Obviously, this approach is not sustainable if many multiplications are performed. Instead, the idea is to find a ciphertext $d_2' \in \mathcal{R}_q^2$ that evaluates to $d_2\mathsf{sk}^2$ when it is decrypted with sk. This way, $(-d_1, d_0) + d_2'$ can be used without the aforementioned problems. Fortunately, it is (relatively) straightforward to find a suitable d_2'. To this end, one prepares a so-called evaluation key $\mathsf{evk} = (a_{\mathsf{evk}}, b_{\mathsf{evk}})$ during the key generation phase, where $b_{\mathsf{evk}} = a_{\mathsf{evk}}\mathsf{sk} + e_{\mathsf{evk}} + P\,\mathsf{sk}^2 \in \mathcal{R}_{Pq}$. The random and error polynomial $a_{\mathsf{evk}}, e_{\mathsf{evk}} \in \mathcal{R}_{Pq}$ are generated like a, e but their coefficients come from a bigger set due to the large and positive integer P. Next, observe that

$$\mathsf{Dec}_{\mathsf{sk}}\left(P^{-1} d_2 \mathsf{evk}\right) = P^{-1} d_2 a_{\mathsf{evk}}\mathsf{sk} + P^{-1} d_2 e_{\mathsf{evk}} + d_2\mathsf{sk}^2 - P^{-1} d_2 a_{\mathsf{evk}}\mathsf{sk} = d_2\mathsf{sk}^2 + P^{-1} d_2 e_{\mathsf{evk}} \pmod q$$

as desired. The crucial role of the tiny P^{-1} is to keep the error term $P^{-1} d_2 e_{\mathsf{evk}}$ small, but also note that the modulus is reduced from Pq to q (which becomes clear in Sidebar 5). Without P^{-1}, the error is likely gigantic because of $d_2 = a_1 a_2 \bmod q$. Then, evk allows us to realize an encrypted multiplication of $\mathsf{ct}(z_1)$ and $\mathsf{ct}(z_2)$ by

$$\mathsf{ct}(z_1) \otimes \mathsf{ct}(z_2) = \left(-d_1, d_0\right) + \left\lfloor P^{-1} d_2 \mathsf{evk}\right\rceil \bmod q = \left(-d_1 + \left\lfloor P^{-1} d_2 a_{\mathsf{evk}}\right\rceil, d_0 + \left\lfloor P^{-1} d_2 b_{\mathsf{evk}}\right\rceil\right) \bmod q,$$

where rounding ensures integer coefficients. Then, by construction $\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}(z_1) \otimes \mathsf{ct}(z_2))$ results in $d_0 + d_1\mathsf{sk} + \mathsf{Dec}_{sk}(P^{-1} d_2 \mathsf{evk}) \approx z_1 z_2 \pmod q$. A numeric illustration of ciphertext operations can be found in our code[5].

Sidebar 4: The intuition behind the CKKS encoding

A detailed explanation of $\mathsf{Ecd}(\cdot)$ is out of the scope of this paper. We refer the interested reader to Lyubashevsky et al. (2013, Section 2.5.2) and instead focus on an illustration. Here, it is easier to start backwards by defining the decoding $\boldsymbol{x}_{\mathrm{Dcd}} = \mathsf{Dcd}(\tilde{\boldsymbol{z}}) = \boldsymbol{V}\tilde{\boldsymbol{z}}$ first. It maps coefficients $\tilde{z}_i$ of polynomials z from $\mathcal{R}_q$ to $\mathbb{C}^{N/2}$, where N has to be a power of 2. In particular, the Vandermonde matrix $\boldsymbol{V}$ is specified by $x_{\mathrm{Dcd},i} = \sum_{j=0}^{N-1} \tilde{z}_j \omega_M^{(4i+1)j}$ where $\omega_M = \exp(\mathrm{i}2\pi/M)$ is a primitive Mth root of unity and $i \in \mathbb{N} \cap [0, N/2)$. Due to symmetry constraints, this procedure can fill up to $N/2$ of the coefficient slots in $\mathcal{R}_q$ with complex numbers (essentially two real numbers). Conversely, the encoding is $\boldsymbol{x}_{\mathrm{Ecd}} = \mathsf{Ecd}(\boldsymbol{x}) \in \mathbb{R}^N$, which, based on the coefficients, can be understood as a mapping from $\mathbb{C}^{N/2}$ to $\mathbb{R}^N$. The missing link in going from $\mathbb{R}^N$ to coefficients in $\mathcal{R}_q$ is (again) $\tilde{\boldsymbol{z}} = \lfloor s\boldsymbol{x}_{\mathrm{Ecd}}\rceil \bmod q$ as in (4). Finally, by inverting the decoding and with $\boldsymbol{x}' = (\boldsymbol{x}^\top, \overline{\boldsymbol{x}}^\top)^\top$, we find

$$\boldsymbol{x}_{\mathrm{Ecd}} = \mathsf{Ecd}(\boldsymbol{x}) = \frac{1}{N}\begin{pmatrix}\boldsymbol{V}^\top & \overline{\boldsymbol{V}^\top}\end{pmatrix}\boldsymbol{x}' \quad \text{which is verified by} \quad \mathsf{Dcd}(\tilde{\boldsymbol{z}}) = \boldsymbol{V}\lfloor s\boldsymbol{x}_{\mathrm{Ecd}}\rceil \approx s\begin{pmatrix}\boldsymbol{I} & \boldsymbol{0}\end{pmatrix}\boldsymbol{x}' = s\boldsymbol{x}$$

presupposed $sx_i + N/s \in \mathcal{Z}_q$ for all $i \in N/2$ with the decoding error $\|\boldsymbol{V}\|_\infty/(2s) = N/s$.
Example. We consider $(q, N, s) = (10^4, 4, 10)$ and the (real) vectors $\boldsymbol{x}_1 = (1.11, 2.22)^\top$ and $\boldsymbol{x}_2 = (3.33, -4.44)^\top$. Then, for $\boldsymbol{x}_{\mathrm{Ecd},i} = \mathsf{Ecd}(\boldsymbol{x}_i)$, we find $\boldsymbol{x}_{\mathrm{Ecd},1} = (1.67, 0.39, 0, -0.39)^\top$ and $\boldsymbol{x}_{\mathrm{Ecd},2} = (-0.56, -2.74, 0, 2.74)^\top$. Next, $\tilde{\boldsymbol{z}}_i = \lfloor s\boldsymbol{x}_{\mathrm{Ecd},i}\rceil \bmod q$ are the coefficients for $z_1, z_2 \in \mathcal{R}_q$, such that we find $z_1 + z_2 = 11 - 23X + 23X^3 \bmod q$ and $z_1 z_2 = -318 - 483X + 483X^3 \bmod q$. Lastly, $\mathsf{Dcd}(z_1 + z_2) = (43.53, -21.53)^\top$ and $\mathsf{Dcd}(z_1 z_2) = (365.07, -1001.07)^\top$ which approximate $s(\boldsymbol{x}_1 + \boldsymbol{x}_2) = (44.4, -22.2)^\top$ and $s^2\boldsymbol{x}_1 \circ \boldsymbol{x}_2 = (360.63, -985.68)^\top$. Note that especially during the multiplication, many quantized values are involved, which leads to an additional error. However, we omit specifying computation errors here. A numeric illustration of this encoding can be found in our code[5].

m parallel RLWE-ciphertext multiplications (2) suffice to evaluate (20). To this end, the message polynomial coefficients are the rows of

$$\left(\lfloor s\boldsymbol{F}_0\rceil, \lfloor s\boldsymbol{F}_1\rceil, \ldots, \lfloor s\boldsymbol{F}_{n-1}\rceil\right) \bmod q \quad \text{and the vector}$$

$$(\lfloor s\boldsymbol{y}(k)^\top\rceil, \lfloor s\boldsymbol{y}(k-1)^\top\rceil, \ldots, \lfloor s\boldsymbol{y}(k-n-1)^\top\rceil)^\top \bmod q.$$

Moreover, the amount of data for a subsequent communication can approximately be halved by extracting an LWE ciphertext from the results according to $b = \tilde{b}_{N-1}$ and

$$\boldsymbol{a} = (\tilde{a}_{N-1}, \tilde{a}_{N-2}, \ldots, \tilde{a}_0)^\top, \mathbf{sk} = (\tilde{\mathsf{sk}}_0, \tilde{\mathsf{sk}}_1, \ldots \tilde{\mathsf{sk}}_{N-1})^\top.$$

Finally, although both consider a finite amount of information, (20) often performs better than a reset controller since the reset controller periodically re-enters a convergence phase.

The benefit of integer $\boldsymbol{A}$ can also be exploited in different ways. First, Kim, Shim, and Han (2023) propose to augment (17a) with $\boldsymbol{E}\boldsymbol{u}(k) - \boldsymbol{E}\boldsymbol{u}(k)$. Together with (17b), this results in

$$\boldsymbol{x}(k+1) = (\boldsymbol{A} - \boldsymbol{E}\boldsymbol{C})\boldsymbol{x}(k) + (\boldsymbol{B} - \boldsymbol{E}\boldsymbol{D})\boldsymbol{y}(k) + \boldsymbol{E}\boldsymbol{u}(k), \tag{21}$$

which obviously requires to feedback $\boldsymbol{u}(k)$. However, this feedback now allows adjusting the eigenvalues of the controller matrix $\boldsymbol{A} - \boldsymbol{E}\boldsymbol{C}$ via $\boldsymbol{E}$. Together with a state transformation, one can finally obtain $\boldsymbol{T}(\boldsymbol{A} - \boldsymbol{E}\boldsymbol{C})\boldsymbol{T}^{-1} \in \mathcal{Z}^{n \times n}$ as desired. Considering the integer-based variant of (21), it becomes however clear that $\lfloor s\boldsymbol{u}(k)\rceil \bmod q$ and the controller output $\boldsymbol{v}(k) = \lfloor s\boldsymbol{C}\rceil\lfloor s^2\boldsymbol{x}(k)\rceil \bmod q$ have different scalings. Thus, a rescaling is performed at a trusted third party. Nonetheless, in comparison to rescaling $\boldsymbol{z}(k)$ as in (18), the communication effort is typically reduced due to $m < n$. Second, algebraic integers (roots of monic polynomials with integer coefficients) can be exploited (Kim, Shim, Sandberg, & Johansson, 2021). To this end, (17) is additively decomposed into a stable and unstable part. For the stable part, a FIR approximation is used. In contrast, each unstable eigenvalue λ is approximated with an algebraic integer $\lambda \approx \varphi$ such that $\varphi^r \in \mathbb{Z}$, where $r \in \mathbb{N}$ controls the approximation quality. With this at hand, a time-varying transformation can be used to derive an equivalent controller that has an integer state matrix.

Sidebar 5: Ciphertext rescaling

To prepare the rescaling, instead of q, consider the modulus $q_\ell = p^\ell q_0$ at level $\ell \in \mathbb{N}$, where q_0 is called base modulus and $p = q_\ell / q_{\ell-1}$. Then, the rescaling operation RS acts on a ciphertext $\mathsf{ct}(z) \in \mathcal{R}^2_{q_\ell}$ at level ℓ and outputs

$$\mathsf{RS}(\mathsf{ct}(z)) = \lfloor p^{-1}\mathsf{ct}(z) \rceil = \lfloor (p^{-1}a, p^{-1}b) \rceil = (a_{\mathsf{RS}}, b_{\mathsf{RS}}) \pmod{q_{\ell-1}}.$$

An obvious (but unwanted) effect is that $\mathsf{RS}(\mathsf{ct}(z)) \in \mathcal{R}^2_{q_{\ell-1}}$ is now valid under $q_{\ell-1}$, i.e, the modulus has been reduced by p^{-1}. However, in order to see the purpose of RS, we first introduce w as the polynomial of remainders modulo q_ℓ, i.e., $a\,\mathsf{sk} + e + z \bmod q = a\,\mathsf{sk} + e + z - wq_\ell$. Then, we obtain $b_{\mathsf{RS}} = \lfloor p^{-1}b \rceil = \lfloor p^{-1}(a\,\mathsf{sk} + e + z - wq_\ell) \rceil$. Next, the rounding errors $\epsilon_a := \lfloor p^{-1}a \rceil - p^{-1}a = a_{\mathsf{RS}} - p^{-1}a$ and $\epsilon_z := \lfloor p^{-1}(z+e) \rceil - p^{-1}(z+e)$ lead to

$$b_{\mathsf{RS}} = \lfloor (\lfloor p^{-1}a \rceil - \epsilon_a)\,\mathsf{sk} + \lfloor p^{-1}(z+e) \rceil - \epsilon_z - wq_{\ell-1} \rceil = a_{\mathsf{RS}}\mathsf{sk} + \lfloor p^{-1}(z+e) \rceil + \lfloor -\epsilon_a \mathsf{sk} - \epsilon_z \rceil \pmod{q_{\ell-1}}.$$

Here, the term of interest is $\lfloor p^{-1}(z+e) \rceil$ which shows that RS can reduce the scaling of z and the error e by p^{-1} while the additional error $\lfloor -\epsilon_a \mathsf{sk} - \epsilon_z \rceil$ is made small by choosing ternary coefficients for sk. The rescaling technique was introduced in Brakerski et al. (2014, Sect. 1.4) and is used in the context presented here in Cheon et al. (2017, Sect. 1). Furthermore, a numeric illustration can be found in our code[5]. The attentive reader may still wonder why RS is useful because $\lfloor p^{-1}z \rceil$ and the message space $p^{-1}|\mathcal{M}| \approx q_{\ell-1}/2$ are reduced by the same amount. In this context, let us investigate an encrypted computation of z^k, where k is a power of 2 and $\lfloor sx \rceil \mod q$ is encoded in one coefficient of $z \in \mathcal{R}_q$. Then, naively computing $\mathsf{ct}(z) \otimes \mathsf{ct}(z)$ k-times (or with binary exponentiation) allows the evaluation of $k \leq \log_s(q_\ell/2)$ multiplications until $\lfloor sx \rceil^k \notin \mathcal{Z}_{q_\ell}$. However, performing RS with $p = s$ after each multiplication yields $\mathsf{ct}(z^k) = \mathsf{RS}(\mathsf{ct}(z^{k/2}) \otimes \mathsf{ct}(z^{k/2}))$, where $\mathsf{ct}(z^{k/2}) = \mathsf{RS}(\mathsf{ct}(z^{k/4}) \otimes \mathsf{ct}(z^{k/4}))$ etc., which requires $\log_2(k)$ rescalings/reductions of q_ℓ. Thus, one can execute $k \approx 2^{\log_s(q/2)}$ multiplications (exponentially more) until the base modulus q_0 is reached, which marks the level where further multiplications might cause an overflow. The price of RS is that operations between ciphertexts are only meaningful if they share the same level ℓ which can be addressed by RS or a simple modulus reduction. It should be noted that RS changes the plaintext to (approximately) $\lfloor p^{-1}z \rceil$ whereas a modulo reduction preserves it exactly.

Due to the structural similarity of Luenberger observers or asymptotic Kalman filters to (17) the major hurdle is again the iteration. Here, Kim and Shim (2019) build on integer matrices enabled by the design freedom of the observer feedback while Alexandru and Pappas (2019) use a third-party re-encryption with masking and a specialized homomorphic encryption scheme for the evaluation.

3.2.2. Data-driven control schemes

Data-driven schemes are of special interest, because they fit well in the context of NCS and naturally enable service-based controller evaluation and design. The first encrypted data-driven scheme is found in Alexandru, Tsiamis, and Pappas (2020), where a CKKS-based implementation of a data-driven linear quadratic regulator (LQR), which builds on Coulson, Lygeros, and Dörfler (2019), is proposed. Mainly, the analytic solution of the corresponding quadratic program is exploited, i.e., it has shallow multiplicative depth and serves as a basis for an online adaption strategy. In particular, incorporating new measurements amounts to rank one updates $\boldsymbol{\mu}\boldsymbol{v}^\top$ of the Hessian $\boldsymbol{H}$, which can be computed by the well-known Sherman–Morrison formula

$$\left(\boldsymbol{H} + \boldsymbol{\mu}\boldsymbol{v}^\top\right)^{-1} = \boldsymbol{H}^{-1} - \frac{\boldsymbol{H}^{-1}\boldsymbol{\mu}\boldsymbol{v}^\top\boldsymbol{H}^{-1}}{1 + \boldsymbol{v}^\top\boldsymbol{H}^{-1}\boldsymbol{\mu}}.$$

The remaining scalar inversion of $1 + \boldsymbol{v}^\top\boldsymbol{H}^{-1}\boldsymbol{\mu}$ is, however, performed at the client's side.

Next, Alexandru, Tsiamis, and Pappas (2021) shows an implementation of an encrypted least-square method with an L_1 regularization (also known as Lasso) which can be used for a data-driven LQR. The solution strategy is based on iterations of the alternating direction method of multipliers (ADMM), approximating the corresponding soft-threshold function on a normalized interval by Chebyshev polynomials, and bootstrapping. The latter is required because ADMM typically requires too many iterations for a practical leveled implementation. Due to the relatively deep computations, both of the aforementioned methods require significant evaluation time such that they are applied to temperature control for building automation.

The first private controller design is proposed in Schlüter and Schulze Darup (2022) by means of a leveled CKKS scheme. In particular, privacy-preserving PID controller tuning for black-box systems is considered. Due to its encryption-friendliness, the authors build on an extremum seeking controller which generates suitable search directions for optimal controller parameters by perturbations. However, designing a reliable extremum seeker without plant knowledge seems impossible, which is why several heuristics to ensure a robust behavior are introduced.

Lastly, reinforcement learning techniques to synthesize a control policy have also been implemented via HE. Because large amounts of data are typically necessary in reinforcement learning, an outsourced storage, learning, and policy evaluation is attractive. The works by Suh and Tanaka (2021a, 2021b) implement the linear updates for, e.g., TD(0) and SARSA(0) using leveled CKKS. Remaining challenges are the evaluation of $\max\{\cdot\}$ expressions and the requirement of bootstrapping, which are currently addressed by means of a third party.

4. Second generation using SMPC

It remains to address those encrypted controllers from the second generation, which mainly rely on SMPC. To do so, we follow the same procedure as in the previous sections. Hence, we first provide an overview of the cryptographic basics and instantiations of SMPC in Sections 4.1 and 4.2, respectively, and then discuss secure control schemes using SMPC in Section 4.3.

4.1. Overview of secure multi-party computation

In a nutshell, SMPC deals with jointly computing a function among M different parties who want to keep their inputs private. The threat models in SMPC have two orthogonal dimensions, namely the parties' behavior and their number. Honest parties follow an SMPC protocol faithfully and do not try to infer information, whereas corrupted parties (mainly) come in two flavors, i.e., *semi-honest* (also known as *honest-but-curious*) and *malicious*. While both semi-honest and malicious parties try to infer information, the former follow the protocol, whereas the latter can arbitrarily deviate from it, e.g., by sending false data. Note that protocols with malicious security can be obtained by enriching a semi-honest protocol with, e.g., special authentication techniques (Damgård et al., 2013), which makes them more costly. In case malicious behavior is detected, the corresponding party is typically exposed, and the protocol is aborted.

Next, if a protocol is secure against $t \in \{1, 2, \dots, M-1\}$ corrupted parties, we write (t, M) which indicates that it is impossible for t dishonest parties to recover secret information via collusion (or by any other means) which will become clear soon. However, $t+1$ colluding parties can recover the secret information trivially. Thus, in comparison to the security of HE schemes (see Section 2.1.2) which rely on the hardness of computational problems, SMPC builds on a non-collusion assumption (and sometimes astronomically unlikely events). Clearly, increasing the threshold t makes breaking the security harder, where $t \geq n/2$ refers to *dishonest majorities* and $t < n/2$ to *honest majorities*. If a $(M-1, M)$ protocol is used and self-trust is given, the security can be ensured in the semi-honest case.

In the context of SMPC, security is more precisely defined and proven using the so-called real-ideal paradigm (Evans, Kolesnikov, & Rosulek, 2018). This is basically a thought experiment, in which a given protocol is executed in an ideal world where a completely trustworthy third party securely receives the parties' inputs, evaluates the function, and securely distributes the results. In this idealization, it is easy to reason about security because a given party's *view* V_i^{ideal} ($i \in \{1, 2, \dots, M\}$), i.e., all data that the party ever gets to see during execution of the protocol, consists only of its own input and output. In the real world, however, no such trusted party exists. Instead, the parties exchange messages with each other, so that each party's view V_i^{real} is extended compared to the ideal world by all messages received during the execution of the protocol and the party's source of randomness. To prove the protocol secure, one has to show that there exist efficient algorithms S_i (called simulators) that can generate something that is indistinguishable from V_i^{real}, based on their own source of randomness and V_i^{ideal}. Clearly, if that is possible, V_i^{real} does not contain secret information.

In particular, many SMPC protocols guarantee *statistical indistinguishability*, which is defined via the statistical distance (Cramer et al., 2015, Ch. 2)

$$D(X, Y) = \frac{1}{2} \sum_{d \in D} |\Pr[X = d] - \Pr[Y = d]|, \tag{22}$$

where X, Y are random variables and D is their common range. Then, X and Y are called statistically indistinguishable if their statistical distance is negligible, i.e.,

$$D(X, Y) \leq \mathsf{negl}(\kappa)$$

for the security parameter κ. Under the simplifying assumption of only one semi-honest party, this idea can be directly applied to a given SMPC protocol by letting X, Y capture the distributions of V_i^{real} and $S_i(V_i^{\text{ideal}})$, respectively. While extending the definition to multiple corrupted parties is straightforward, treating malicious adversaries is more involved which is why we omit it here and instead refer to the discussion in Evans et al. (2018, Sect. 2.3) or Lindell (2017).

In conclusion, a realistic threat model is of paramount importance for the security of an SMPC scheme because it decides between confidentiality and a complete loss of privacy. The latter is particularly likely when the assumptions about collusions or the parties' behavior are inappropriate. In comparison, HE does not suffer from such assumptions and may be considered more robust.

For a thorough introduction to SMPC, security notions, and more advanced topics, we refer to Boneh and Shoup (2023), Cramer et al. (2015), and Evans et al. (2018).

4.2. Instantiations of secure multi-party computation

There are different approaches available in order to realize SMPC, which have specific advantages and disadvantages, mostly in terms of communication overhead. These approaches can also be combined, which can lead to more efficient overall solutions. Here, we make distinctions in terms of the computation domain, i.e., binary or arithmetic, and the primitives involved for some popular protocols.

Binary circuits are often computed with garbled circuits (GC) (Bellare, Hoang, & Rogaway, 2012), GMW (Goldreich, 2004), or BGW (Ben-Or, Goldwasser, & Wigderson, 1988). GC is a two-party protocol and secure against semi-honest parties. The idea in GC is that one party (the garbler) represents a function in terms of Boolean gates/small lookup tables, but it replaces the input and outputs with random labels. The resulting, typically very large "garbled" circuit and the garbler's input labels are then sent to the other party (the evaluator) for evaluation. Finally, the evaluator's input labels are obtained from the garbler by using a protocol called oblivious transfer (Rabin, 2005), which ensures that the evaluator only learns the correct labels and the garbler does not learn the selection of the evaluator. For security reasons, one cannot (or only partially) reuse a GC. Now, the generation of a GC is typically more intensive than its evaluation and the communication cost is constant but depends on the circuit size. Especially when bandwidth is limited, the communication constitutes the bottleneck. In comparison to GC, GMW and BGW naturally extend to many parties, and the number of communication rounds (how often a communication is invoked) is dependent on the circuit's depth. However, we skip details about GMW because it is often slow (oblivious transfer is used gate-wise) and BGW because it is based on secret sharing, which we will cover soon.

If arithmetic needs to be performed, the size of a corresponding binary circuit is often prohibitive, such that (linear) secret sharing (SS) schemes (Benaloh & Leichter, 1990; Cramer et al., 2015; Shamir, 1979) offer a more efficient solution. These provide (at first) a protocol for additions and multiplications which can also evaluate logic gates for $q = 2$ and require only cheap local computations. In comparison to GC, small amounts of data are sent, but the number of communication rounds depends on the circuit's depth. Thus, network latency is a major concern in the case of SS. Furthermore, it offers flexibility in terms of the number of corrupted parties and can be extended to provide security against malicious parties quite efficiently. As a result, competitive SMPC protocols often build on SS as a primitive, which is why we chose to illustrate an additive SS scheme in the next section.

Finally, HE can also be used to realize SMPC. This can be helpful if the communication is prohibitive, but the setup is still distributed. For instance, Mouchet, Troncoso-Pastoriza, Bossuat, and Hubaux (2021) describe a scheme, where the secret key is additively shared among the parties, which results in a $(M-1, M)$ scheme. However, also threshold schemes (t, M) are possible. The parties then compute locally and only interact for bootstrapping, which is cheaper than in a single-party setup but requires communication. Furthermore, there also exist proxy re-encryption schemes, where the idea is to collect ciphertexts from different parties encrypted under different secret keys and to convert them into a cryptosystem with a shared public key. From here on, the computation can be performed by a single party. Essential to this approach is a key-switching functionality that can, e.g., be realized in LWE cryptosystems very similarly to the multiplication shown in Sidebar 3.

4.2.1. Additive secret sharing

Consider M parties and secret numbers $z_i \in \mathbb{Z}_q$, where standard representatives are commonly used and $i, M \in \mathbb{N}$. Then, each party can generate shares $z_i^{(j)}$ of its secret number z_i such that

$$z_i = z_i^{(1)} + z_i^{(2)} + \cdots + z_i^{(M)} \bmod q = \sum_{j=1}^{M} z_i^{(j)} \bmod q \tag{23}$$

where the $z_i^{(j)}$, $j \in \{1, \dots, M\} \setminus \{k\}$, are drawn uniformly at random from $\mathbb{Z}_q$. Then, the remaining share is

$$z_i^{(k)} = z_i - \sum_{j=1, j \neq k}^{M} z_i^{(j)} \bmod q.$$

Importantly, one can show that $z_i^{(k)}$ is also uniformly random due to the distribution of the other shares and the modulo reduction. Therefore, applying the reconstruction (23) to any subset of $\{1,\dots,M\}$ with $M-1$ or less shares results in a uniformly random number. The indistinguishability of these numbers from uniformly random ones leads to the conclusion that the combination of up to $M-1$ shares contains no information ($D=0$ in (22)) Consequently, one can safely distribute the shares among the other parties, where we assume that the jth party receives $\{z_1^{(j)}, z_2^{(j)}, \dots, z_M^{(j)}\}$. Only when M shares are combined, the secret can be recovered by (23). As a result, additive secret sharing provides security against $(M-1, M)$ semi-honest parties.

Next, additive secret sharing readily supports linear operations such as additions and public multiplications. In this context, the shorthand notation $[\![z_i]\!] := (z_i^{(1)}, z_i^{(2)}, \dots, z_i^{(M)})$ is helpful. There, operations on $[\![z_i]\!]$ are understood component-wise, i.e., locally at each party, if not stated otherwise. With this at hand, an addition of secrets $z_3 = z_1 + z_2$ can be compactly written based on the shares $[\![z_1]\!]$ and $[\![z_2]\!]$ in terms of

$$[\![z_3]\!] := [\![z_1]\!] + [\![z_2]\!] \mod q. \tag{24}$$

By means of (23), one can verify that $[\![z_3]\!] = [\![z_1+z_2]\!]$ are indeed shares of z_3. Besides, additions and multiplications with a public $c \in \mathbb{Z}_q$ can be defined via

$$\begin{aligned} c + [\![z_1]\!] &:= (z_1^{(1)} + c, z_1^{(2)}, \dots, z_1^{(M)}) \mod q, \\ c\,[\![z_1]\!] &:= (c\, z_1^{(1)}, c\, z_1^{(2)}, \dots, c\, z_1^{(M)}) \mod q. \end{aligned}$$

Remarkably, these operations do not require communication between the parties, whereas multiplying secrets does. To see this, note that $z_1 z_2 = (z_1^{(1)} + \dots + z_1^{(n)})(z_2^{(1)} + \dots + z_2^{(n)}) \mod q$ creates coupling terms $z_1^{(i)} z_2^{(j)}$ with $i \neq j$ that can neither be computed by party i nor by party j alone. In other words, this information must somehow be communicated without revealing a share to any party.

Interestingly, this is possible with "multiplication triples" (Beaver, 1991; Du & Atallah, 2001). In particular, these are $[\![\alpha]\!]$, $[\![\beta]\!]$, and $[\![\gamma]\!]$, where α, β are uniformly random and $\gamma = \alpha\beta \mod q$. A multiplication $z_4 = z_1 z_2$, based on the distributed shares $[\![z_1]\!]$ and $[\![z_2]\!]$, is then realized as follows. First, $[\![\delta]\!] = [\![z_1]\!] - [\![\alpha]\!] \mod q$ and $[\![\epsilon]\!] = [\![z_2]\!] - [\![\beta]\!] \mod q$ are computed. Then, δ and ϵ, which are by construction uniformly random in $\mathbb{Z}_q$, are revealed by exchanging $[\![\delta]\!]$ and $[\![\epsilon]\!]$ and using the reconstruction (23). Finally, the multiplication is performed by the local computation

$$[\![z_4]\!] = [\![\gamma]\!] + \delta\,[\![\beta]\!] + \epsilon\,[\![\alpha]\!] + \delta\epsilon \mod q. \tag{25}$$

Again, one can verify by reconstructing z_4 from $[\![z_4]\!]$ that $[\![z_4]\!] = [\![z_1\, z_2]\!]$. Overall, each party performs cheap local computations and must broadcast two elements of $\mathbb{Z}_q$ in one communication round per multiplication.

Whenever possible, the generation of multiplication triples and other correlated randomness, that we omit specifying here, during an online phase is avoided because they require (costly) oblivious transfer or homomorphic encryption protocols, e.g., Keller, Orsini, and Scholl (2016) or Cheon et al. (2017). Fortunately, they can be precomputed (see Cramer et al. (2015, Chap. 8) for an optimized solution) because they do not depend on the input data. Alternatively, an additional party (dealer), that is especially devised to provide only these triples, can be used. Other methods for realizing multiplications on shares are discussed briefly in Sidebar 6.

Sidebar 6: Further secret sharing schemes

In the case of honest majorities, Shamir's secret sharing (Shamir, 1979) and replicated secret sharing (Benaloh & Leichter, 1990) can multiply without triples and offer efficient scalar products by means of a resharing protocol. The former is based on polynomials of degree t, such that a secret can be recovered by an interpolation with $t+1$ shares. This also allows detecting whether a minority of parties are sending erroneous shares. The latter method, is based on additive secret sharing, but every party receives more than one share of every secret. Thus, this scheme requires at least three parties (self-trust is insufficient) and encrypted communications.

4.3. Secure control with the help of SMPC

As apparent from Section 4.2.1, SMPC provides an immediate alternative to FHE, as privacy-preserving additions and multiplications are likewise supported. Meanwhile, non-polynomial functions can be realized based on GC or secret sharing protocols. Still, one has to take into account that, while the computational effort is typically reduced compared to HE, the communication effort may be extensive. In this regard, a comparison of HE and GC for private anomaly detection in linear controllers is presented in Alexandru et al. (2022). Namely, the communication intense phase for FHE can be transacted in a preprocessing phase, where evaluation keys are transmitted while the circuit transmission constitutes the bottleneck for GC.

As a consequence, depending on the control scheme at hand, not only pure HE- or SMPC-based implementation are of interest but also hybrid forms. We discuss some illustrative schemes with semi-honest security in the following. Note that some of the schemes mentioned up until this point, e.g., those using a third party, are strictly speaking also SMPC solutions.

4.3.1. Fully encrypted predictive controllers

While encrypted MPC has already been addressed by first generation schemes in Section 2.2.2, we recall that the prototyping implementations showed some weaknesses in terms of unencrypted operations, which were outsourced to trusted third parties. Now, exploiting the features of SMPC, e.g., allows to also securely evaluate the max-expression in (11). Hence, one could revisit the partially encrypted solver-based MPC implementations discussed above and derive fully encrypted variants. However, being able to evaluate max-expressions also opens more exciting directions. In fact, we can reconsider the explicit PWA control law (9) and perform a decomposition into two convex PWA functions (Kripfganz & Schulze, 1987). Since the current segment of convex PWA functions simply reflects the one with the largest function value among all segments, this leads to a reformulation of the form

$$\boldsymbol{g}(\boldsymbol{x}) = \max\{\boldsymbol{\mathcal{K}}\boldsymbol{x} + \boldsymbol{\beta}\} - \max\{\boldsymbol{\mathcal{L}}\boldsymbol{x} + \boldsymbol{\gamma}\},$$

where the involved matrices and vectors are typically large. Nevertheless, the resulting structure can be securely implemented by combining additive secret sharing with garbled circuits (Tjell, Schlüter, Binfet, & Schulze Darup, 2021). This results in a protocol is $(1,2)$ secure against semi-honest adversaries. A robust formulation which takes quantization into account is found in Schlüter and Schulze Darup (2020).

We would like to mention Alexandru et al. (2021) here again (see Section 3.2.2), as it deals with a sparse data predictive control problem and the solution is based on multi-party leveled HE (Mouchet et al., 2021) with a computationally less demanding bootstrapping. In fact, the bootstrapping time only grows slowly with increasing problem dimension, such that encrypted matrix multiplications become the most expensive computation in the algorithm.

4.3.2. Nonlinear controllers and observers

Having the ability to evaluate additions and multiplications in a privacy-preserving manner naturally brings encrypted polynomial control to the table. This has first been addressed in Schulze Darup (2020) based on a hybrid two-party implementation exploiting PHE and additive secret sharing. By involving PHE, secure multiplications were realized without relying on Beaver triples. However, this resulted in relatively slow computations. An improved implementation has been proposed by Schlor, Hertneck, Wildhagen, and Allgöwer (2021), based on a (replicated) secret sharing scheme. More precisely, three parties are utilized to evaluate a degree t polynomial, where none of them is allowed to collude. This way, local multiplications are enabled presupposed the reconstruction is performed at the actuator. For further multiplications, a resharing protocol is required. A third perspective on nonlinear encrypted control is presented by Kim, Farokhi, Shames, and Shim (2021), where controllers of the form

$$u(k) = \Phi(u(k-1), u(k-2), \dots, y(k-1), y(k-2), \dots)$$

are considered. Here, Φ is only required to be representable by encrypted operations, while previous results are rescaled with the help of a re-encryption.

Gonzalez-Serrano, Amor-Martın, and Casamayon-Anton (2014) implement an Extended Kalman Filter using a combination of the Paillier cryptosystem and SMPC, where everything but the nonlinear prediction is kept private. More specifically, their algorithm employs as building blocks secure two-party protocols for multiplication with truncation, comparison, division, and matrix inversion which are realized based on additive masking (mentioned in Section 2.2.2). Comparison of two numbers is achieved by extracting the most significant bit of their difference, while division is implemented in terms of the three-step approach described in Section 5.1. A matrix inversion is then built on top of the other primitives using an LU decomposition. Although the paper does not exploit a special formulation from the control perspective, there is clearly a variety of functions enabled by SMPC in spite of this being an early work. Using similar techniques, Tjell, Cascudo, and Wisniewski (2019) implements a recursive least-square method. We note that Alexandru et al. (2018), which is discussed in Section 2.2.2, could also be included in this category. Lastly, Tjell and Wisniewski (2021) propose a real number SS scheme with an upper bound on the information leakage. An application to adaptive Kalman filtering shows the effectiveness of the approach.

4.3.3. *Distributed optimization*

Another research direction is that of distributed optimization based control, in which privacy-preserving distributed optimization algorithms are needed at the core. Tjell and Wisniewski (2019) address the case of a joint optimization problem of multiple distributed subsystems that are connected through linear constraints in the form

$$\min_{\boldsymbol{x}_1,\dots,\boldsymbol{x}_{\mathcal{N}}} \sum_{i=1}^{\mathcal{N}} f_i(\boldsymbol{x}_i) \quad \text{s.t.} \quad \sum_{i=1}^{\mathcal{N}} \boldsymbol{D}_i \boldsymbol{x}_i - \boldsymbol{e} = \boldsymbol{0}.$$

They introduce a set of computing parties that solve the optimization problem with SS in two different ways, namely via ADMM and dual decomposition, which lead to a simple iteration formula.

Another ADMM-based approach (for a slightly different optimization problem) is presented by Binfet, Adamek, Schlüter, and Schulze Darup (2023), where public-key leveled FHE schemes are utilized for proxy re-encryption in the context of formation control. This way the communication effort can be reduced at the expense of more costly local computations. In particular, a third party provides an instantiation of the cryptosystem, where none of the other parties has access to the secret key. This global instantiation is used to homomorphically carry out all computations during the execution of the ADMM algorithm. In order for a given party to eventually gain access to its results in plaintext, the encrypted optimizer is sent to one of the other parties to evaluate a key switching operation which results in a new ciphertext encrypting the same value that can be decrypted by the intended party.

5. Towards the third generation

Surely, there will always be an additional cost associated with security, but current generations of encrypted controllers are still lagging relatively far behind their plaintext variants. Furthermore, cryptography-based secure control has almost solely dealt with the confidentiality of process data, but not with its integrity. We catch up regarding these two points, in this section. Lastly, we point out promising novel applications of encrypted control.

5.1. *Towards elementary functions*

During Sections 3 and 4, we encountered limitations of current leveled FHE and SMPC schemes used for arithmetic. Namely, (at first) they provide only finite computational depth and multivariate polynomials. Thus, non-polynomial functions have to be approximated by means of polynomials, which is often non-trivial even for seemingly simple computations. While approaches to address the computational depth are closely related to the scheme at hand, polynomial approximations can be discussed more generically.

In this context, a popular choice is a minimax approximation (or Chebyshev polynomials) instead of a Taylor series expansion because the worst-case approximation error is known a priori. For efficiency reasons, one can aim for the lowest order polynomial that still satisfies a predetermined precision. When it comes to polynomial evaluation, the number of private multiplications dominates the complexity in FHE and SMPC. Therefore, the algorithm by Paterson and Stockmeyer (1973), which minimizes the number of non-scalar multiplications to $\mathcal{O}(\sqrt{n})$ by recursively evaluating smaller degree polynomials, is the optimal choice in comparison to a naive evaluation or Horner's method requiring $\mathcal{O}(n^2)$ and $\mathcal{O}(n)$ non-scalar multiplications, respectively. In the context of control, Chebyshev polynomials are deployed in Alexandru et al. (2022, 2021), where the former also uses the Paterson–Stockmeyer algorithm.

Nonetheless, depending on the function type and the size of the domain, approximating a function by a polynomial can be inefficient (or inaccurate). In these cases, a power series, lookup tables, or a three-step approach are commonly used in computer arithmetic (Muller, 2006, Section 4). In the latter, the input is first reduced to a predetermined (typically small) domain. Then, based on precision requirements an approximation by means of a polynomial, lookup table for values or polynomial coefficients, and/or an iteration method (such as Newton, Goldschmidt, Wilkes iteration) is utilized. Finally, the result is transformed back. By means of these approaches, it is possible to construct state-of-the-art implementations of elementary functions such as $\mathrm{sgn}(x)$, $1/x$, $\sqrt{x}$, $\sin(x)$ and, based on that for instance $|x| = \mathrm{sgn}(x)x = \sqrt{x^2}$, $\exp(x)$, $\log(x)$, $\arctan(x)$, $\max\{x, y\} = ((x+y) + |x-y|)/2$, $\mathrm{comp}(x, y) = (\mathrm{sgn}(x-y)+1)/2$ but also $\boldsymbol{A}^{+}$ (see Söderström & Stewart, 1974) or Cordic (Muller, 2006, Section 7). Having reached this stage, it can be seen that it is possible to rebuild more complex algorithms and to compile arbitrary computations, which will rapidly enhance the scalability of encrypted solutions.

5.1.1. *Via homomorphic encryption*

In the context of leveled FHE schemes (such as CKKS), a relatively deep computation as required for basic functions can become a burden. The reason is that a large ciphertext modulus is required, which in turns makes a larger ring dimension necessary for security reasons. Consequently, the overall scheme becomes sluggish. Clearly, this can be addressed by more efficient bootstrapping procedures for leveled FHE, which are highly anticipated. A fruitful endeavor might therefore be the search for more efficient but less general constructions tailored for control applications. Furthermore, representing an elementary function with the least amount of multiplications is important for the overall performance. In this context, trading off some accuracy for speed by using less levels might be of interest. Here, control engineers can contribute by analyzing the overall error and how it affects a closed-loop.

When it comes to complex functions or large domains, it seems that solutions based on lookup tables are out of scope because (according to our knowledge) no practical evaluation method exist for them in leveled schemes. Thus, power series or domain reductions are more common (see, e.g., Cheon, Kim, Kim, Lee, & Lee, 2019; Hong, Park, Cho, Choe, & Cheon, 2022). While the former often requires a large amount of multiplications, the latter suffers from uncertainties regarding the message magnitude. If the magnitude of the message is known, however, domain reduction, and subsequent polynomial approximation and/or iterations seem to be an effective approach. When this is not the case, control engineers could contribute by analyzing/ensuring the stability of such iterations or by tailored constructions to enable a suitable domain reduction.

For completeness, we note that the (non-leveled) TFHE/CGGI scheme (Chillotti, Joye, Ligier, Orfila, & Tap, 2020) already supports basic elementary functions and furthermore a practical bootstrapping routine which allows for a simultaneous function evaluation. However, without getting into the details here, only small integers (less than 8 bit) are supported, the ciphertexts are larger, and computational complexity is higher compared to leveled schemes. Therefore, their application in encrypted control is limited. Currently, extensions to larger integers based on radix or CRT decompositions are under active research (Bergerat et al., 2023, Sects. 2.3 and 4). Nonetheless, discrete event systems can be addressed with TFHE/CGGI schemes due to their binary forms. Besides, Boura, Gama, Georgieva, and Jetchev (2020) combines the strengths of different LWE-based cryptosystems by switching between them based on an overarching algebraic structure. This can lead to an improved performance, if the cost of switching can be compensated by more efficient evaluations. Finally, apart from theoretical improvements, that increased the speed of HE by several orders of magnitude over the past decade, optimized code, compilers, and specialized hardware can bridge the last mile. In fact, with help of field programmable gate arrays, Stobbe, Keijzer, and Ferrari (2022) showed that a robot controller can achieve practical execution times.

5.1.2. *Via secure multi-party computation*

The landscape in SMPC is different from HE and depends on the primitive. For instance, in SS based schemes there already exist truncation routines that compute, e.g., $[\![\lfloor z/s \rfloor]\!]$ based on $[\![z]\!]$ quite efficiently such that an arbitrary amount of private multiplications is enabled. Here, the works Catrina and Saxena (2010) (or similarly Dalskov, Escudero, & Keller, 2019) and Mohassel and Zhang (2017) stand out. Assuming preprocessing, the former realizes a truncation protocol with 1 communication round by means of additive masking at the cost of statistical instead of perfect security (still high enough in practice) and larger ciphertexts. The latter truncation protocol can even be performed locally. However, in addition to the cost from before, the protocol may fail unexpectedly with a large error, which may be unsuitable for control applications.

Furthermore, comparisons $[\![z_1 \leq z_2]\!]$ can be realized (for instance, based on Catrina & De Hoogh, 2010; Catrina & Saxena, 2010, Damgård, Fitzi, Kiltz, Nielsen, & Toft, 2006 or Damgård, Geisler, and Krøigaard (2007)). Thus, regarding computations, elementary functions and more complex algorithms can be realized with suitable running times for certain applications if the setup (mostly in terms of latency and threat model) permits it (see Aly & Smart, 2019, for benchmarks). A useful toolbox in this context is MP-SPDZ (Keller, 2020), where several schemes are available, and the communication rounds are optimized automatically. Benchmarks for SS schemes and GC applied to different computations can also be found in Keller (2020).

Finally, the standard SMPC security model considers equally (un)trustworthy parties. However, real applications commonly encompass a far more nuanced set of performance and security constraints, which enables a tailored threat model for control with performance benefits. For instance, an asymmetric trust model with honest majority for computation outsourcing may be suitable.

5.2. *Towards integrity*

In practice, integrity is often ensured by access control or identity/attribute-based policies. In that case, a protection against unauthorized adversarial data changes can be achieved. But when it comes to computation outsourcing, these techniques require a trustworthy external party. When it comes to untrustworthy environments, we mainly dealt with confidentiality, i.e., protecting the privacy of the data by means of cryptographic tools, so far. However, these tools are malleable by principle, i.e, malicious attackers can alter the ciphertext (even though they remain oblivious about the underlying plaintext), which is critical in many applications but enabled computation outsourcing in the first place. In this context, a stealthy attack for control systems is proposed in Alisic, Kim, and Sandberg (2023). There, it is assumed that a basic LWE scheme is used for encryption. The attack then exploits the idea of combining ciphertext, as briefly discussed in Sidebar 1. Therefore, additional means to ensure integrity are needed.

A generic method that ensures the integrity of control algorithms is non-trivial to construct. Mainly because the computational cost on the system side should be strictly less than the computational cost of the algorithm itself. Otherwise, outsourcing it would be questionable. With this in mind, control-related approaches, such as those reviewed in Section 1, seem not expedient. Yet, optimization and (possibly) mechanism design constitute exceptions. In the former, the integrity of a result is easily verifiable by the KKT conditions. The latter builds on designing rewards for a computation party such that sending correct data is the most desirable outcome for it (according to the revelation principle, see Hurwicz and Reiter (2006)). Rationality must, however, be assumed.

Nonetheless, cryptographic tools that can ensure integrity of generic computations are actively being developed and entail additional overhead. We will provide a brief overview in the following because these tools are entire research fields of their own which build on constructions that are different from what we have encountered in this survey. Moreover, they are rarely found in encrypted control literature yet.

5.2.1. *In homomorphic encryption setups*

Analogously to HE, there exist homomorphic authentication techniques, where homomorphic signatures are attached to the data (Gorbunov, Vaikuntanathan, & Wichs, 2015). A valid signature corresponds to the correct execution of the computation and can be verified. Furthermore, symmetric-key based solutions such as homomorphic MACs are available (Gennaro & Wichs, 2013). Unfortunately, homomorphic authentication suffers similarly to HE from high computational complexity (and partially from security issues). In particular, the verification effort and the signature size grow with the complexity of the computation. With a restriction to linear operations, an authentication fast enough for a linear controller can be achieved (Cheon, Han, Hong, et al., 2018).

Another (often more efficient) approach is called verifiable computations. In contrast to before, where a valid signature or MAC can only be obtained by correct computations, the cloud now proves a statement, with help of an argument π based on probabilistic tests and cryptographic primitives, to the client that a computation was correctly performed while it is not possible to cheat. This way, one can authenticate a computation by creating π such that it proves that a signed input belongs to a computation output. Again, linear controllers have been addressed in this context. A tailored method via group-based cryptography can be found in Cheon, Kim, Kim, Lee, and Shim (2020). Another promising tool of this kind is zero-knowledge proofs (zk), where we refer to Sun et al. (2021) for an introduction. Here, non-interactive variants, such as zk-SNARKs (e.g., Parno, Howell, Gentry, & Raykova, 2016) stand out due to their computational and communication efficiency. In fact, many other zk proofs produce heavy communication overhead. In the context of RLWE, there exist the recent zk-SNARK variant by Ganesh, Nitulescu, and Soria-Vazquez (2021) which is tailored for computations over polynomial rings. Of course, zk are also of use in the context of SMPC, where each party can prove they are following the protocol without revealing their private inputs.

5.2.2. In secure multi-party computation setups

Again, by means of specialized protocols against malicious parties which are built on semi-honest ones, one can ensure integrity. Let us again put our focus on SS based protocols and not repeat aforementioned techniques. Then, an early and influential construction is presented by Damgård, Pastro, Smart, and Zakarias (2012). This scheme, commonly called "SPDZ", is based on an additive SS scheme and thus secure against dishonest majorities. In addition, SPDZ verifies the correctness of a party's computation with an efficient (additive) secretly shared global random MAC. The MAC relates a secret with the global key and has additive properties such that it behaves like a normal share. Furthermore, it enables the parties to verify their computation mutually. Here, SMPC enables a much simpler construction compared to HE.

Next, apart from the online phase, one also has to take the preprocessing into account. State-of-the-art protocols for offline and online phases with malicious security in the dishonest and honest majority case are Chida et al. (2018) and Keller et al. (2016), respectively. Note that many malicious protocols still build on SPDZ or adaptions of it. Finally, Dalskov, Escudero, and Keller (2020) shows that if for every corrupted party four honest parties are guaranteed, then the computation output can always be delivered.

5.3. Towards further applications

In many cases, cloud-services are already deployed for data storage and computation services, which is enabled by the fulfillment of legal restrictions and trust. Meanwhile, the significant overhead generated by current cryptographic solutions make their usage unattractive in cases where cloud-usage is already possible. Therefore, applications that are enabled by confidentiality and integrity in the realm of control are of interest, until further improvements are made.

However, the landscape looks different when data disclosure has serious implications. In this context, applications related to personal data stand out. Consider, for example, sensor fusion of cardiorespiratory signals as in Linschmann, Leonhardt, and Hoog Antink (2020) (health data), intelligent transportation systems (geolocation data), predictive control for epidemics as in Köhler et al. (2021) (health data), or smart grids (metering data). Moreover, cooperative planning tasks, such as a supply chain optimizations or resource allocation across multiple companies, are conceivable. There, each company profits from participating but is not running the risk of leaking data. Finally, one can think of service-based controller designs (as introduced in Schlüter & Schulze Darup, 2022), which makes state-of-the-art controllers more accessible with wide-ranging implications for the performance of closed-loop systems.

6. Conclusions

In control, safety is classically addressed by concepts such as stability or robustness. Considering the rising deployment of large networked control systems handling vast amounts of data, it is becoming equally important to address cybersecurity threats effectively. Specifically, ensuring the confidentiality and integrity of process data is crucial.

To establish a basis for meeting these needs, we first provided an overview of the current state of encrypted control in this article. In this context, we classified existing schemes into two generations depending on the dominant cryptographic tools. Although this classification may not be sharp in all cases, it reflects the efforts towards more powerful, efficient, and secure encrypted controllers. As a next step towards secure networked control systems, we envisioned promising research directions for the next generation of encrypted controllers. Finally, in order to pave the way for control engineers to enter the field, we built on illustrative presentations and left out many technicalities. In particular, we provided a step-by-step explanation and tutorial to the CKKS cryptosystem, which can currently be considered as the most competitive HE scheme for arithmetic operations.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

We share code for educational purposes in the GitHub repository referenced in footnote 5.

Acknowledgments

Financial support by the German Research Foundation (DFG) and the Daimler and Benz Foundation, Germany under the grants SCHU 2940/4-1 and 32-08/19, respectively, is gratefully acknowledged.

References

Al Badawi, A., Bates, J., Bergamaschi, F., Cousins, D. B., Erabelli, S., Genise, N., et al. (2022). OpenFHE: Open-source fully homomorphic encryption library. In *Proceedings of the 10th workshop on encrypted computing & applied homomorphic cryptography* (pp. 53–63).

Albrecht, M., Chase, M., Chen, H., Ding, J., Goldwasser, S., Gorbunov, S., et al. (2018). *Homomorphic encryption security standard*: *Technical Report*, Toronto, Canada: HomomorphicEncryption.org.

Albrecht, M. R., Player, R., & Scott, S. (2015). On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, *9*, 169–203.

Alexandru, A. B., Burbano, L., Ćeliktuğ, M. F., Gomez, J., Cardenas, A. A., Kantarcioglu, M., et al. (2022). Private anomaly detection in linear controllers: Garbled circuits vs. Homomorphic encryption. In *Proceedings of the 61st conference on decision and control* (pp. 7746–7753). IEEE.

Alexandru, A. B., Morari, M., & Pappas, G. J. (2018). Cloud-based MPC with encrypted data. In *Proceedings of the 57th conference on decision and control* (pp. 5014–5019). IEEE.

Alexandru, A. B., & Pappas, G. J. (2019). Encrypted LQG using labeled homomorphic encryption. In *Proceedings of the 10th ACM/IEEE conference on cyber-physical systems* (pp. 129–140).

Alexandru, A. B., Schulze Darup, M., & Pappas, G. J. (2019). Encrypted cooperative control revisited. In *Proceedings of the 58th conference on decision and control* (pp. 7196–7202). IEEE.

Alexandru, A. B., Tsiamis, A., & Pappas, G. J. (2020). Towards private data-driven control. In *Proceedings of the 59th conference on decision and control* (pp. 5449–5456). IEEE.

Alexandru, A. B., Tsiamis, A., & Pappas, G. J. (2021). Encrypted distributed Lasso for sparse data predictive control. In *Proceedings of the 60th conference on decision and control* (pp. 4901–4906). IEEE.

Alisic, R., Kim, J., & Sandberg, H. (2023). Model-free undetectable attacks on linear systems using LWE-based encryption. *IEEE Control Systems Letters*, *7*, 1249–1254.

Alladi, T., Chamola, V., & Zeadally, S. (2020). Industrial control systems: Cyberattack trends and countermeasures. *Computer Communications*, *155*, 1–8.

Aly, A., & Smart, N. P. (2019). Benchmarking privacy preserving scientific operations. In *Proceedings of the 17th conference on applied cryptography and network security* (pp. 509–529). Springer.

Amin, S., Cárdenas, A. A., & Sastry, S. S. (2009). Safe and secure networked control systems under denial-of-service attacks. In *Proceedings of the 12th conference on hybrid systems: Computation and control* (pp. 31–45). Springer.

Beaver, D. (1991). Efficient multiparty protocols using circuit randomization. In *Proceedings of the international cryptology conference* (pp. 420–432). Springer.

Bellare, M., Hoang, V. T., & Rogaway, P. (2012). Foundations of garbled circuits. In *Proceedings of the conference on computer and communications security* (pp. 784–796).

Bemporad, A., Morari, M., Dua, V., & Pistikopoulos, E. N. (2002). The explicit linear quadratic regulator for constrained systems. *Automatica*, *38*, 3–20.

Ben-Or, M., Goldwasser, S., & Wigderson, A. (1988). Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *Proceedings of the 20th symposium on theory of computing* (pp. 1–10). New York, NY, USA: ACM.

Benaloh, J., & Leichter, J. (1990). Generalized secret sharing and monotone functions. In *Advances in cryptology — CRYPTO' 88* (pp. 27–35). Springer New York.

Bergerat, L., Boudi, A., Bourgerie, Q., Chillotti, I., Ligier, D., Orfila, J. -B., et al. (2023). Parameter optimization and larger precision for (T)FHE. *Journal of Cryptology*, *36*, 28.

Binfet, P., Adamek, J., Schlüter, N., & Schulze Darup, M. (2023). Towards privacy-preserving cooperative control via encrypted distributed optimization. *Automatisierungstechnik*, *71*, 736–747.

Bishop, M., et al. (2005). *Introduction to computer security, volume 50*. Addison-Wesley Boston.

Blum, A., Kalai, A., & Wasserman, H. (2003). Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM*, *50*, 506–519.

Boneh, D., & Shoup, V. (2023). *A graduate course in applied cryptography*. http://toc.cryptobook.us/.

Boura, C., Gama, N., Georgieva, M., & Jetchev, D. (2020). Chimera: Combining ring-LWE-based fully homomorphic encryption schemes. *Journal of Mathematical Cryptology*, *14*, 316–338.

Boyd, S. P., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Brakerski, Z., Gentry, C., & Vaikuntanathan, V. (2014). (Leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory*, *6*, 1–36.

Catrina, O., & De Hoogh, S. (2010). Improved primitives for secure multiparty integer computation. In *Security and cryptography for networks: 7th international conference* (pp. 182–199). Springer.

Catrina, O., & Saxena, A. (2010). Secure computation with fixed-point numbers. In *Proceedings of the conference on financial cryptography and data security* (pp. 35–50). Springer.

Cetinkaya, A., Ishii, H., & Hayakawa, T. (2019). An overview on denial-of-service attacks in control systems: Attack models and security analyses. *Entropy*, *21*, 210.

Chen, H., Laine, K., & Player, R. (2017). Simple encrypted arithmetic library – SEAL v2.1. In *Financial cryptography and data security* (pp. 3–18). Springer.

Cheon, J. H., Han, K., Hong, S. -M., Kim, H. J., Kim, J., Kim, S., et al. (2018). Toward a secure drone system: Flying with real-time homomorphic authenticated encryption. *IEEE Access*, *6*, 24325–24339.

Cheon, J. H., Han, K., Kim, H., Kim, J., & Shim, H. (2018). Need for controllers having integer coefficients in homomorphically encrypted dynamic system. In *Proceedings of the 57th IEEE conference on decision and control* (pp. 5020–5025). IEEE.

Cheon, J. H., Han, K., Kim, A., Kim, M., & Song, Y. (2019). A full RNS variant of approximate homomorphic encryption. In *Selected areas in cryptography – SAC 2018* (pp. 347–368). Springer.

Cheon, J. H., Hhan, M., Hong, S., & Son, Y. (2019). A hybrid of dual and meet-in-the-middle attack on sparse and ternary secret LWE. *IEEE Access*, *7*, 89497–89506.

Cheon, J. H., Kim, D., Kim, D., Lee, H. H., & Lee, K. (2019). Numerical method for comparison on homomorphically encrypted numbers. In S. D. Galbraith, & S. Moriai (Eds.), *Advances in cryptology – ASIACRYPT 2019*. Springer International Publishing.

Cheon, J. H., Kim, D., Kim, J., Lee, S., & Shim, H. (2020). Authenticated computation of control signal from dynamic controllers. In *Proceedings of the 59th conference on decision and control* (pp. 3249–3254). IEEE.

Cheon, J. H., Kim, A., Kim, M., & Song, Y. (2017). Homomorphic encryption for arithmetic of approximate numbers. In *Proceedings of the conference on theory and application of cryptology and information security* (pp. 409–437). Springer.

Chida, K., Genkin, D., Hamada, K., Ikarashi, D., Kikuchi, R., Lindell, Y., et al. (2018). Fast large-scale honest-majority MPC for malicious adversaries. Cryptology ePrint Archive, Paper 2018/570.

Chillotti, I., Gama, N., Georgieva, M., & Izabachène, M. (2020). TFHE: Fast fully homomorphic encryption over the torus. *Journal of Cryptology*, *33*, 34–91.

Chillotti, I., Joye, M., Ligier, D., Orfila, J. -B., & Tap, S. (2020). CONCRETE: Concrete operates on ciphertexts rapidly by extending TFHE. In *WAHC 8th workshop on encrypted computing & applied homomorphic cryptography*.

Chong, M. S., Sandberg, H., & Teixeira, A. M. H. (2019). A tutorial introduction to security and privacy for cyber-physical systems. In *Proceedings of the 18th European control conference* (pp. 968–978). IEEE.

Coulson, J., Lygeros, J., & Dörfler, F. (2019). Data-enabled predictive control: In the shallows of the DeePC. In *Proceedings of the 18th European control conference* (pp. 307–312). IEEE.

Cramer, R., Damgård, I. B., & Nielsen, J. B. (2015). *Secure multiparty computation*. Cambridge University Press.

Curtis, B. R., & Player, R. (2019). On the feasibility and impact of standardising sparse-secret LWE parameter sets for homomorphic encryption. In *Proceedings of the 7th ACM workshop on encrypted computing & applied homomorphic cryptography* (pp. 1–10). ACM.

Daemen, J., & Rijmen, V. (1999). *AES proposal: Rijndael*. MD, USA: Gaithersburg.

Dalskov, A., Escudero, D., & Keller, M. (2019). Secure evaluation of quantized neural networks. arXiv preprint arXiv:1910.12435.

Dalskov, A., Escudero, D., & Keller, M. (2020). Fantastic four: Honest-majority four-party secure computation with malicious security. Cryptology ePrint Archive, Paper 2020/1330.

Damgård, I., Fitzi, M., Kiltz, E., Nielsen, J. B., & Toft, T. (2006). Unconditionally secure constant-rounds multi-party computation for equality, comparison, bits and exponentiation. In *Theory of cryptography conference* (pp. 285–304). Springer.

Damgård, I., Geisler, M., & Krøigaard, M. (2007). Efficient and secure comparison for on-line auctions. In *Australasian conference on information security and privacy* (pp. 416–430). Springer.

Damgård, I., Keller, M., Larraia, E., Pastro, V., Scholl, P., & Smart, N. P. (2013). Practical covertly secure MPC for dishonest majority–or: Breaking the SPDZ limits. In *Computer security–ESORICS 2013* (pp. 1–18). Springer.

Damgård, I., Pastro, V., Smart, N., & Zakarias, S. (2012). Multiparty computation from somewhat homomorphic encryption. In *Annual cryptology conference* (pp. 643–662). Springer.

De Persis, C., & Tesi, P. (2015). Input-to-state stabilizing control under denial-of-service. *IEEE Transactions on Automatic Control*, *60*, 2930–2944.

Diffie, W., & Hellman, M. E. (2022). New directions in cryptography. In *Democratizing cryptography: The work of Whitfield Diffie and Martin Hellman* (pp. 365–390).

Du, W., & Atallah, M. J. (2001). Secure multi-party computation problems and their applications: A review and open problems. In *Proceedings of the workshop on new security paradigms* (pp. 13–22).

Du, W., & Zhan, Z. (2002). A practical approach to solve secure multi-party computation problems. In *Proceedings of the 2002 workshop on new security paradigms* (pp. 127–135).

Ducas, L., & Micciancio, D. (2015). FHEW: Bootstrapping homomorphic encryption in less than a second. In *Advances in cryptology–EUROCRYPT 2015* (pp. 617–640). Springer.

Dwork, C. (2006). Differential privacy. In *Proceedings of the 33rd colloquium on automata, languages and programming* (pp. 1–12). Springer.

ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, *31*, 469–472.

Evans, D., Kolesnikov, V., & Rosulek, M. (2018). A pragmatic introduction to secure multi-party computation. *Foundations and Trends in Privacy and Security*, *2*, 70–246.

Fan, J., & Vercauteren, F. (2012). Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Paper 2012/144.

Farokhi, F., Shames, I., & Batterham, N. (2016). Secure and private cloud-based control using semi-homomorphic encryption. *IFAC-PapersOnLine*, *49*, 163–168.

Ferrari, R. M. G., & Teixeira, A. M. H. (2017). Detection and isolation of replay attacks through sensor watermarking. *IFAC-PapersOnLine*, *50*, 7363–7368.

Fontaine, C., & Galand, F. (2007). A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security*, *2007*, 1–10.

Fujita, T., Kogiso, K., Sawada, K., & Shin, S. (2015). Security enhancements of networked control systems using RSA public-key cryptosystem. In *Proceedings of the 10th Asian control conference* (pp. 1–6). IEEE.

Ganesh, C., Nitulescu, A., & Soria-Vazquez, E. (2021). Rinocchio: SNARKs for ring arithmetic. Cryptology ePrint Archive, Paper 2021/322.

Gennaro, R., & Wichs, D. (2013). Fully homomorphic message authenticators. In *Proceedings of the conference on theory and application of cryptology and information security* (pp. 301–320). Springer.

Gentry, C. (2009). Fully homomorphic encryption scheme using ideal lattices. In *Proceedings of the 41st ACM symposium on theory of computing* (pp. 169–178). Association for Computing Machinery.

Goldreich, O. (2004). *Foundations of cryptography: Volume 2, basic applications*. Cambridge University Press.

Gonzalez-Serrano, F. J., Amor-Martın, A., & Casamayon-Anton, J. (2014). State estimation using an extended Kalman filter with privacy-protected observed inputs. In *Workshop on information forensics and security* (pp. 54–59). IEEE.

Gorbunov, S., Vaikuntanathan, V., & Wichs, D. (2015). Leveled fully homomorphic signatures from standard lattices. In *Proceedings of the 47th symposium on theory of computing* (pp. 469–477). ACM.

Hadjicostis, C. N. (2018). Privary preserving distributed average consensus via homomorphic encryption. In *Proceedings of the 57th conference on decision and control* (pp. 1259–1263). IEEE.

Hadjicostis, C. N., & Domínguez-García, A. D. (2020). Privacy-preserving distributed averaging via homomorphically encrypted ratio consensus. *IEEE Transactions on Automatic Control*, *65*, 3887–3894.

Hale, M. T., & Egerstedt, M. (2015). Differentially private cloud-based multi-agent optimization with constraints. In *Proceedings of the American control conference* (pp. 1235–1240).

Halevi, S., & Shoup, V. (2020). Design and implementation of HElib: A homomorphic encryption library. Cryptology ePrint Archive, Paper 2020/1481.

Han, S., & Pappas, G. J. (2018). Privacy in control and dynamical systems. *Annual Review of Control, Robotics, and Autonomous Systems*, *1*, 309–332.

Han, S., Topcu, U., & Pappas, G. J. (2017). Differentially private distributed constrained optimization. *IEEE Transactions on Automatic Control*, *62*, 50–64.

Hassan, M. U., Rehmani, M. H., & Chen, J. (2019). Differential privacy techniques for cyber physical systems: A survey. *IEEE Communications Surveys & Tutorials*, *22*, 746–789.

Hassibi, B., & Vikalo, H. (2002). On the expected complexity of integer least-squares problems. In *Proceedings of the conference on acoustics, speech, and signal processing*: *Vol. 2*, (pp. II–1497). IEEE.

Hemsley, K. E., Fisher, E., et al. (2018). *History of industrial control system cyber incidents*: *Technical Report*, Idaho Falls: Idaho National Laboratory.

Hong, S., Park, J. H., Cho, W., Choe, H., & Cheon, J. H. (2022). Secure tumor classification by shallow neural network using homomorphic encryption. *BMC Genomics*, *23*, 1–19.

Huang, Z., Mitra, S., & Vaidya, N. (2015). Differentially private distributed optimization. In *Proceedings of the 16th international conference on distributed computing and networking* (pp. 1–10).

Hurwicz, L., & Reiter, S. (2006). *Designing economic mechanisms*. Cambridge University Press.

Katz, J., & Lindell, Y. (2020). *Introduction to modern cryptography*. CRC Press.

Keller, M. (2020). MP-SPDZ: A versatile framework for multi-party computation. In *Proceedings of the SIGSAC conference on computer and communications security* (pp. 1575–1590). ACM.

Keller, M., Orsini, E., & Scholl, P. (2016). MASCOT: Faster malicious arithmetic secure computation with oblivious transfer. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* (pp. 830–842).

Kim, J., Farokhi, F., Shames, I., & Shim, H. (2021). Toward nonlinear dynamic control over encrypted data for infinite time horizon. arXiv:2110.06270.

Kim, J., Kim, D., Song, Y., Shim, H., Sandberg, H., & Johansson, K. H. (2022). Comparison of encrypted control approaches and tutorial on dynamic systems using Learning With Errors-based homomorphic encryption. *Annual Reviews in Control*.

Kim, J., Lee, C., Shim, H., Cheon, J. H., Kim, A., Kim, M., et al. (2016). Encrypting controller using fully homomorphic encryption for security of cyber-physical systems. In *Proceedings of the 6th IFAC workshop on distributed estimation and control in networked systems* (pp. 175–180).

Kim, J., & Shim, H. (2019). Encrypted state estimation in networked control systems. In *Proceedings of the 58th conference on decision and control* (pp. 7190–7195). IEEE.

Kim, J., Shim, H., & Han, K. (2023). Dynamic controller that operates over homomorphically encrypted data for infinite time horizon. *IEEE Transactions on Automatic Control*, *68*, 660–672.

Kim, J., Shim, H., Sandberg, H., & Johansson, K. H. (2021). Method for running dynamic systems over encrypted data for infinite time horizon without bootstrapping and re-encryption. In *Proceedings of the 60th conference on decision and control* (pp. 5614–5619). IEEE.

Kishida, M. (2018). Encrypted average consensus with quantized control law. In *Proceedings of the 57th conference on decision and control* (pp. 5850–5856). IEEE.

Kogiso, K., & Fujita, T. (2015). Cyber-security enhancement of networked control systems using homomorphic encryption. In *Proceedings of the 54th conference on decision and control* (pp. 6836–6843).

Köhler, J., Schwenkel, L., Koch, A., Berberich, J., Pauli, P., & Allgöwer, F. (2021). Robust and optimal predictive control of the COVID-19 outbreak. *Annual Reviews in Control*, *51*, 525–539.

Kripfganz, A., & Schulze, R. (1987). Piecewise affine functions as a difference of two convex functions. *Optimization*, *18*, 23–29.

Lei, X., Liao, X., Huang, T., & Heriniaina, F. (2014). Achieving security, robust cheating resistance, and high-efficiency for outsourcing large matrix multiplication computation to a malicious cloud. *Information Sciences*, *280*, 205–217.

Lin, F., Fardad, M., & Jovanovic, M. R. (2011). Augmented Lagrangian approach to design of structured optimal state feedback gains. *IEEE Transactions on Automatic Control*, *56*, 2923–2929.

Lin, Y., Farokhi, F., Shames, I., & Nešić, D. (2018). Secure control of nonlinear systems using semi-homomorphic encryption. In *Proceedings of the 57th conference on decision and control* (pp. 5002–5007). IEEE.

Lindell, Y. (2017). How to simulate it – A tutorial on the simulation proof technique. In *Tutorials on the foundations of cryptography: Dedicated to Oded Goldreich* (pp. 277–346). Springer.

Linschmann, O., Leonhardt, S., & Hoog Antink, C. (2020). Model-based sensor fusion of multimodal cardiorespiratory signals using an unscented Kalman filter. *Automatisierungstechnik*, *68*, 933–940.

Lyubashevsky, V., Peikert, C., & Regev, O. (2013). A toolkit for ring-LWE cryptography. In *Advances in cryptology–EUROCRYPT 2013* (pp. 35–54). Springer.

Marcolla, C., Sucasas, V., Manzano, M., Bassoli, R., Fitzek, F. H., & Aaraj, N. (2022). Survey on fully homomorphic encryption, theory, and applications. *Proceedings of the IEEE*, *110*, 1572–1609.

Mo, Y., & Sinopoli, B. (2009). Secure control against replay attacks. In *Proceedings of the 47th annual allerton conference* (pp. 911–918).

Mo, Y., & Sinopoli, B. (2012). Integrity attacks on cyber-physical systems. In *Proceedings of the 1st international conference on high confidence networked systems* (pp. 47–54).

Mohassel, P., & Zhang, Y. (2017). SecureML: A system for scalable privacy-preserving machine learning. In *Symposium on security and privacy* (pp. 19–38). IEEE.

Mouchet, C., Troncoso-Pastoriza, J., Bossuat, J. -P., & Hubaux, J. -P. (2021). Multiparty homomorphic encryption from ring-learning-with-errors. In *Proceedings on Privacy Enhancing Technologies*: *Vol. 2021*, (pp. 291–311).

Muller, J. -M. (2006). *Elementary functions*. Springer.

Murguia, C., Farokhi, F., & Shames, I. (2020). Secure and private implementation of dynamic controllers using semi-homomorphic encryption. *IEEE Transactions on Automatic Control*, *65*, 3950–3957.

Naseri, A. M., Lucia, W., & Youssef, A. (2022). A privacy preserving solution for cloud-enabled set-theoretic model predictive control. In *Proceedings of the European control conference* (pp. 894–899). IEEE.

Nozari, E., Tallapragada, P., & Cortés, J. (2016). Differentially private distributed convex optimization via objective perturbation. In *Proceedings of the American control conference* (pp. 2061–2066). IEEE.

Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *Advances in cryptology - Eurocrypt '99, volume 1592* (pp. 223–238). Springer.

Parno, B., Howell, J., Gentry, C., & Raykova, M. (2016). Pinocchio: Nearly practical verifiable computation. *Communications of the ACM*, *59*, 103–112.

Pasqualetti, F., Dörfler, F., & Bullo, F. (2013). Attack detection and identification in cyber-physical systems. *IEEE Transactions on Automatic Control*, *58*, 2715–2729.

Paterson, M. S., & Stockmeyer, L. J. (1973). On the number of nonscalar multiplications necessary to evaluate polynomials. *SIAM Journal on Computing*, *2*, 60–66.

Pollard, J. M. (1971). The fast Fourier transform in a finite field. *Mathematics of Computation*, *25*, 365–374.

Rabin, M. O. (2005). How to exchange secrets with oblivious transfer. Cryptology ePrint Archive.

Rawlings, J. B., Mayne, D. Q., & Diehl, M. (2017). *Model predictive control: Theory, computation, and design, Vol. 2*. Nob Hill Publishing Madison.

Regev, O. (2005). On lattices, learning with errors, random linear codes, and cryptography. In *Proceedings of the 37th symposium on theory of computing* (pp. 84–93). ACM.

Regev, O. (2009). On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM*, *56*, 1–40.

Regev, O. (2010). The learning with errors problem. In *Proceedings of the 25th conference on computational complexity* (pp. 191–204).

Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, *21*, 120–126.

Ruan, M., Gao, H., & Wang, Y. (2019). Secure and privacy-preserving consensus. *IEEE Transactions on Automatic Control*, *64*, 4035–4049.

Scherer, C., & Weiland, S. (2015). *Linear matrix inequalities in control*.

Schlor, S., Hertneck, M., Wildhagen, S., & Allgöwer, F. (2021). Multi-party computation enables secure polynomial control based solely on secret-sharing. In *Proceedings of the 60th conference on decision and control* (pp. 4882–4887). IEEE.

Schlüter, N., Binfet, P., Kim, J., & Schulze Darup, M. (2022). Encrypted distributed state estimation via affine averaging. In *Proceedings of the 61st conference on decision and control* (pp. 7754–7761). IEEE.

Schlüter, N., Binfet, P., & Schulze Darup, M. (2023). Cryptanalysis of random affine transformations for encrypted control. In *Proceedings of the IFAC World Congress* (pp. 12031–12038).

Schlüter, N., Neuhaus, M., & Schulze Darup, M. (2021). Encrypted dynamic control with unlimited operating time via FIR filters. In *Proceedings of the European control conference* (pp. 947–952).

Schlüter, N., & Schulze Darup, M. (2020). Encrypted explicit MPC based on two-party computation and convex controller decomposition. In *Proceedings of the 59th conference on decision and control* (pp. 5469–5476). IEEE.

Schlüter, N., & Schulze Darup, M. (2021). On the stability of linear dynamic controllers with integer coefficients. *IEEE Transactions on Automatic Control*, *67*, 5610–5613.

Schlüter, N., & Schulze Darup, M. (2022). Encrypted extremum seeking for privacy-preserving PID tuning as-a-service. In *Proceedings of the European control conference* (pp. 1288–1293).

Schulze Darup, M. (2020). Encrypted polynomial control based on tailored two-party computation. *International Journal of Robust and Nonlinear Control*, *30*, 4168–4187.

Schulze Darup, M., Alexandru, A. B., Quevedo, D. E., & Pappas, G. J. (2021). Encrypted control for networked systems: An illustrative introduction and current challenges. *IEEE Control Systems Magazine*, *41*, 58–78.

Schulze Darup, M., Redder, A., & Quevedo, D. E. (2018). Encrypted cloud-based MPC for linear systems with input constraints. In *Proceedings of the 6th IFAC conference on nonlinear model predictive control NMPC 2018* (pp. 635–642).

Schulze Darup, M., Redder, A., & Quevedo, D. E. (2019). Encrypted cooperative control based on structured feedback. *IEEE Control Systems Letters*, *3*, 37–42.

Schulze Darup, M., Redder, A., Shames, I., Farokhi, F., & Quevedo, D. (2018). Towards encrypted MPC for linear constrained systems. *IEEE Control Systems Letters*, *2*, 195–200.

Shamir, A. (1979). How to share a secret. *Communications of the ACM*, *22*, 612–613.

Shan, Z., Ren, K., Blanton, M., & Wang, C. (2018). Practical secure computation outsourcing: A survey. *ACM Computing Surveys*, *51*, 1–40.

Shoukry, Y., Gatsis, K., Alanwar, A., Pappas, J. G., Seshia, S. A., Srivastava, M., et al. (2016). Privacy-aware quadratic optimization using partially homomorphic encryption. In *Proceedings of the 55th conference on decision and control* (pp. 5053–5058).

Smith, R. S. (2011). A decoupled feedback structure for covertly appropriating networked control systems. *IFAC Proceedings Volumes*, *44*, 90–95.

Söderström, T., & Stewart, G. W. (1974). On the numerical properties of an iterative method for computing the Moore–Penrose generalized inverse. *SIAM Journal on Numerical Analysis*, *11*, 61–74.

Stobbe, P., Keijzer, T., & Ferrari, R. M. G. (2022). A fully homomorphic encryption scheme for real-time safe control. In *Proceedings of the 61st conference on decision and control* (pp. 2911–2916). IEEE.

Suh, J., & Tanaka, T. (2021a). Encrypted value iteration and temporal difference learning over leveled homomorphic encryption. In *Proceedings of the 2021 American control conference* (pp. 2555–2561). IEEE.

Suh, J., & Tanaka, T. (2021b). SARSA (0) reinforcement learning over fully homomorphic encryption. In *Proceedings of the 2021 SICE international symposium on control systems* (pp. 1–7). IEEE.

Sultangazin, A., & Tabuada, P. (2020). Symmetries and isomorphisms for privacy in control over the cloud. *IEEE Transactions on Automatic Control*, *66*, 538–549.

Sun, X., Yu, F. R., Zhang, P., Sun, Z., Xie, W., & Peng, X. (2021). A survey on zero-knowledge proof in blockchain. *IEEE Network*, *35*, 198–205.

Teixeira, A., Shames, I., Sandberg, H., & Johansson, K. H. (2015). A secure control framework for resource-limited adversaries. *Automatica*, *51*, 135–148.

Tjell, K., Cascudo, I., & Wisniewski, R. (2019). Privacy preserving recursive least squares solutions. In *Proceedings of the 18th European control conference* (pp. 3490–3495).

Tjell, K., Schlüter, N., Binfet, P., & Schulze Darup, M. (2021). Secure learning-based MPC via garbled circuit. In *Proceedings of the 60th conference on decision and control* (pp. 4907–4914). IEEE.

Tjell, K., & Wisniewski, R. (2019). Privacy preservation in distributed optimization via dual decomposition and ADMM. In *Proceedings of the 58th conference on decision and control* (pp. 7203–7208). IEEE.

Tjell, K., & Wisniewski, R. (2021). Privacy in distributed computations based on real number secret sharing. arXiv preprint arXiv:2107.00911.

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

Wang, C., Ren, K., & Wang, J. (2011). Secure and practical outsourcing of linear programming in cloud computing. In *Proceedings of the IEEE Infocom* (pp. 820–828). IEEE.

Wei, K., Li, J., Ding, M., Ma, C., Yang, H. H., Farokhi, F., et al. (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, *15*, 3454–3469.

Xu, Z., & Zhu, Q. (2015). Secure and resilient control design for cloud enabled networked control systems. In *Proceedings of the first ACM workshop on cyber-physical systems-security and/or privacy* (pp. 31–42).

Zhang, K., Li, Z., Wang, Y., Louati, A., & Chen, J. (2022). Privacy-preserving dynamic average consensus via state decomposition: Case study on multi-robot formation control. *Automatica*, *139*, Article 110182.