Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 49-22 (2016) 175–180

Encrypting Controller using Fully Homomorphic Encryption for Security of Cyber-Physical Systems ⋆

Junsoo Kim * **Chanhwa Lee** * **Hyungbo Shim** *
Jung Hee Cheon ** **Andrey Kim** ** **Miran Kim** **
Yongsoo Song **

* *ASRI, Dep. of Electrical and Computer Engineering, Seoul National University, Seoul, Korea.*
** *Dep. of Mathematical Sciences, Seoul National Univ., Seoul, Korea.*

Abstract: In order to enhance security of cyber-physical systems, it is important to protect the signals from sensors to the controller, and from the controller to the actuator, because the attackers often steal and compromise those signals. One immediate solution could be encrypting the signals, but in order to perform computation in the controller, they should be decrypted before computation and encrypted again after computation. For this, the controller keeps the secret key, which in turn increases vulnerability from the attacker. In this paper, we introduce the *fully homomorphic encryption (FHE)*, which is an advanced cryptography that has enabled arithmetic operations directly on the encrypted variables without decryption. However, this also introduces several new issues that have not been studied for conventional controllers. Most of all, an encrypted variable has a finite lifespan, which decreases as an arithmetic operation is performed on it. Our solution is to run multiple controllers, and orchestrate them systematically. Also, in order to slow down the decrease of the lifespan, a tree-based computation of sequential matrix multiplication is introduced. We finally demonstrate the effectiveness of the proposed algorithm with quadruple water tank example.

Keywords: Cyber-physical system, Security, Homomorphic encryption, Controller encryption

1. INTRODUCTION

As physical systems are connected to computers through network communications, real control systems can be a target of cyber attackers. Such an integration of computation (cyber part), physical process (physical part), and communication (link between cyber and physical parts), is called *cyber-physical systems* (CPSs). By its openness and connectivity nature, CPS is vulnerable to malicious attacks. Furthermore, it is much more dangerous than conventional cyber attacks by a hacker since failure or malfunction on the critical infrastructures, such as power plants and transportation systems, caused by cyber-physical attacks lead to a tremendous catastrophe (Slay and Miller, 2007; Langner, 2011). Thus, security of CPS is becoming more and more important and attracts many researchers' attention recently (Sandberg et al., 2015).

In control system community, CPSs are regarded as large-scale networked control systems. Therefore, they focus on the system theoretic properties of physical plants and try to enhance security of CPS by adopting and modifying advanced control techniques. Various control engineering methods are applied to increase security in physical layers of CPS, such as fault detection and isolation (Pasqualetti et al., 2013), robust optimal control (Amin et al., 2009), estimation theory (Lee et al., 2015), network graph theory (Sundaram and Hadjicostis, 2011), and game-theoretic approach (Zhu and Basar, 2015).

However, fundamental limitations on detectability of attacks are investigated by Pasqualetti et al. (2013) and the finding of *zero-dynamics attack*, which is in the category of stealthy attack (Teixeira et al., 2012), has made the situation difficult because, in theory, there is no way to detect the attack by any anomaly detectors such as fault detection methods (Teixeira et al., 2015). On the other hand, in order to maintain stealthiness, the attacker should continuously generate and inject signals exactly corresponding to transmission zeros of the system. This requires exact model knowledge to the attacker, and so the attacker may need to monitor input and output signals to build the model. Similarly, a recent robust zero-dynamics attack by Park et al. (2016) also requires input and output information of the plant. Therefore, the security level of CPSs can be enhanced if the feedback loop is entirely encrypted so that original messages are protected and are not revealed to adversaries.

In this regard, one can employ encryption to protect communication of information between plant and controller. See Fig. 1 (left figure). However, a drawback of conventional encryption is that the received information should be decrypted in order to compute the control input. That is, secret key must be kept inside the controller, which has possible risk to be stolen by attackers. Hence, it is desired if the computation for control input is performed without

⋆ The work of J. Kim, C. Lee, and H. Shim was supported by ICT R&D program of MSIP/IITP Grant number 14-824-09-013, Resilient Cyber-Physical Systems Research. The work of J. H. Cheon, A. Kim, M. Kim, and Y. Song was supported by IT R&D program of MSIP/KEIT [No. 0450-21060006] and Samsung Electronics Co., Ltd. (No. 0421-20150074). Corresponding author: Hyungbo Shim.

2405-8963 © 2016, IFAC (International Federation of Automatic Control) Hosting by Elsevier Ltd. All rights reserved.
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2016.10.392

encrypting the data, so that the controller does not have to maintain the secret key of the encryption.

Homomorphic encryption (HE) is a cryptographic scheme that allows homomorphic operations (e.g., homomorphic addition and multiplication) on encrypted data without decryption. Since Gentry (2009) discovered the first plausible construction of *fully homomorphic encryption* (FHE) scheme, many other HE schemes have been suggested following Gentry's blueprint (Cheon and Stehlé, 2015). This in turn allows possibility that the controller does not have to maintain the secret key. The right figure in Fig. 1 illustrates the control configuration using FHE. We will call this scheme as 'encrypting controller.'

In data aggregation for smart grids, HE is used to protect the privacy of users (Li et al., 2010). However, to the best of authors' knowledge, the first attempt to employ HE in controller has been made by Kogiso and Fujita (2015). They used ElGamal (1984) encryption, but it does not allow addition between encrypted signals. In order to overcome this difficulty, the controller transmits many pieces of decrypted information to the actuator block, which are then encrypted and added to compose the control input. Unfortunately, in order to update the internal state of the controller, the outcome of this addition is also necessary in the controller. Thus, this outcome is passed to and encrypted again in the sensor block and is transmitted back to the controller. This scheme unnecessarily increases complexity of control system and requires more network throughput (or, channel capacity) of communication.

In this paper, we employ FHE for computation of encrypted signals. Use of FHE in the control system is new, and so, there may be potential difficulties. One of apparent difficulties is so-called 'bootstrapping' the encrypted variable. Unlike the plaintext variable (the term indicating un-encrypted information), the encrypted variable (which is often called ciphertext) has a lifespan which decreases whenever operation such as multiplication or addition is performed. We will briefly review this phenomenon in Section 2. During the bootstrapping of encrypted variable is performed, the controller cannot operate. To overcome this difficulty, we propose running multiple controllers and a 'catch-up' method that allows resetting the controller state (after bootstrapping) without decrypting any variables in the controller (Section 4). In addition, decrease of lifespan is proportional to the number of matrix multiplications. In order to slow down the decrease rate of lifespan as the multiplication is repeated, we develop a 'tree-based multiplication algorithm' (Section 5). This algorithm increases the lifespan of encrypted variables from the order of h to 2^h by using additional h memory and computation.

More potential issues may be

- Size of encrypted variable: When a variable is encrypted, its size usually increases. As this size increases, more network throughput is required. In this paper, a symmetric key HE is used for encryption since it is simpler and faster than by a public key HE and moreover, the size of ciphertexts can be compressed using a pseudo random number generator (PRNG) by substituting their random part as a seed. We also assume the actuator and the sensor are designed by a trusted party and they share the same symmetric key. If it is not the case, the secret keys in the actuator and the sensor should be different, and it is necessary to use a public key HE between them.
- Speed of arithmetic operation: Speed of arithmetic operation on the encrypted variable is slow than on the plaintext. While this problem needs more attention from the cryptography community, the controller design should also take into account this point. In this paper, controller parameters are not encrypted in order to speed up the operations. Multiplication between plaintext and encrypted variable is much faster than between both encrypted variables. As we consider linear controllers, only products of the plaintext gains and the encrypted (controller) states/inputs are necessary.

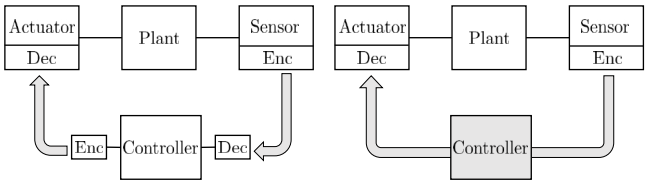

Fig. 1. Configurations with conventional encryption and with FHE

As a showcase, we will illustrate in Section 6 the use of FHE for the water-tank system, that has been used as a testbed of cyber-physical security (Teixeira et al., 2012).

2. FULLY HOMOMORPHIC ENCRYPTION

Notation. All logarithms are base 2 unless otherwise indicated. The usual dot product of two vectors is denoted by $\langle \cdot, \cdot \rangle$. For a real number r, $\lfloor r \rceil$ denotes the nearest integer to r. For a positive integer q, we use $\mathbb{Z} \cap [-q/2, q/2)$ as a representative of $\mathbb{Z}_q$. We use $x \leftarrow \mathcal{D}$ to denote the uniform sampling x according to distribution $\mathcal{D}$. For a set S, $U(S)$ denotes the uniform distribution on S. For a positive number σ, we denote $\mathcal{D}_\sigma$ the discrete Gaussian distribution of parameter σ. Throughout the paper, we let λ denote the security parameter: all known valid attacks against the cryptographic scheme under scope should take $\Omega(2^\lambda)$ bit operations. [1]

2.1 Learning With Errors (LWE)

The LWE problem was introduced by Regev (2005) as a generalization of learning parity with noise. Suppose that positive integers n and $q \geq 2$ are given. For $s \in \mathbb{Z}_q^n$ and a distribution χ over $\mathbb{Z}$, we define $A_{q,\chi}^{\mathsf{LWE}}(s)$ as the distribution obtained by sampling $a \leftarrow U(\mathbb{Z}_q^n)$ and $e \leftarrow \chi$, and returning $c = (b, a) \in \mathbb{Z}_q^{n+1}$ where $b = \langle a, \boldsymbol{s} \rangle + e$.

Let $\mathcal{D}$ be a distribution on $\mathbb{Z}_q^n$. The learning with errors problem, denoted by $\mathsf{LWE}_{n,q,\chi}(\mathcal{D})$, is to distinguish arbitrarily many independent samples chosen according to $A_{q,\chi}^{\mathsf{LWE}}(\boldsymbol{s})$ for a fixed $\boldsymbol{s} \leftarrow \mathcal{D}$, from $U(\mathbb{Z}_q^{n+1})$.

The LWE problem is self-reducible, that is, $\mathsf{LWE}_{n,q,\chi}(\mathcal{D})$ can be reduced to $\mathsf{LWE}_{n,q,\chi}(U(\mathbb{Z}_q^n))$ for any distribution $\mathcal{D}$. It was shown that the hardness of $\mathsf{LWE}_{n,q,\chi}(U(\mathbb{Z}_q^n))$ can be established by reductions to approximate short vector problems in worst-case lattices (Regev, 2005; Peikert,

[1] One writes $f(n) = \Omega(g(n))$ if and only if there exists $M > 0$ and $n_0 \in \mathbb{Z}$ such that $|f(n)| \geq M\,|g(n)|$ for all $n \geq n_0$.

2009). Moreover, one can reduce $\mathsf{LWE}_{n,q,\mathcal{D}_\sigma}(U(\mathbb{Z}_q^n))$ efficiently to $\mathsf{LWE}_{n,q,\mathcal{D}_\sigma}(\mathcal{D}_\sigma)$ when D_σ is a discrete Gaussian distribution with an appropriate parameter $\sigma = \alpha q$ (Applebaum et al., 2009). For the hardness of LWE problem to ensure a time/advantage ratio of at least $\Omega(2^\lambda)$, the size of samples $n \log q$ should be at least $\Omega(\frac{\lambda}{\log \lambda} \log^2 \alpha)$ to thwart the lattice reduction attacks (Micciancio and Regev, 2009). By the estimation of Lindner and Peikert (2011), the size of ciphertexts is at least $\frac{\lambda+110}{7.2} \cdot \log^2 \alpha$. To get 80-bit security level, it is enough to set $n \log q \geq 27 \cdot \log^2 \alpha$.

2.2 A LWE-based Homomorphic Encryption Scheme

Most of existing HE schemes have fixed plaintext spaces and support modular operations. Recently, Cheon et al. (2016) suggested a homomorphic encryption scheme allowing floating-point operations between ciphertexts. The following is a description of floating-point homomorphic encryption scheme based on LWE problem.

- $\mathsf{Setup}(1^\lambda)$. Take a modulus $\mathfrak{p}$ and base modulus $q_0 = q_0(\lambda)$. Let $q_l = q_0 \cdot \mathfrak{p}^l$ for $l = 1, \ldots, L$. Choose the parameter $n = n(\lambda, q_L)$ and an error parameter $\alpha = \alpha(\lambda, q_L)$ appropriately for $\mathsf{LWE}_{n,q_L,D_\sigma}(D_\sigma)$ problem that achieves at least 2^λ security for $\sigma = \alpha q_L$. Output the parameters $params = (\mathfrak{p}, q_L, n, \sigma)$.
- $\mathsf{SecretKeyGen}(params)$. Sample and set the secret key $sk = s \leftarrow \mathcal{D}_\sigma^n$.
- $\mathsf{Enc}(m, sk)$. Encryption of the message $m \in \mathbb{Z}_{q_0}^\ell$. Sample $a \leftarrow U(\mathbb{Z}_{q_L}^{\ell \times n})$ and $e \leftarrow \mathcal{D}_\sigma^\ell$. Output $c = (b, a) \in \mathbb{Z}_{q_L}^{\ell \times (n+1)}$ of level L for $b = -a \cdot s + m + e \bmod q_L$ where $\cdot$ is an ordinary matrix multiplication.
- $\mathsf{Add}(c_1, c_2)$ For two ciphertexts c_1 of level l_1 and c_2 of level l_2, output $c_1 + c_2 \bmod q_{\min\{l_1,l_2\}}$ of level $\min\{l_1, l_2\}$.
- $\mathsf{mMult}(A, c)$. For a real matrix $A \in \mathbb{R}^{\ell \times \ell}$ and a ciphertext $c \in \mathbb{Z}_{q_l}^{\ell \times (n+1)}$ of level l, output $\lfloor (\lfloor \mathfrak{p} A \rceil \cdot c)/\mathfrak{p} \rceil \bmod q_{l-1}$ of level $l-1$.
- $\mathsf{Dec}(c, sk)$. For a ciphertext $c = (b, a) \in \mathbb{Z}_{q_l}^{\ell \times (n+1)}$ at level l, output $m' = b + a \cdot s \bmod q_l$.

This scheme is IND-CPA secure under the decisional LWE assumption of parameter (n, q_L, σ). The output $m' = m + e$ of its decryption circuit is slightly different from the original message m. However, the output can be considered to be an approximate value of the original message in floating-point computation if e is small enough. Note that $\mathsf{Add}(c_1, c_2)$ is an encryption of the summation of corresponding messages $m_1 + m_2$ and $\mathsf{mMult}(A, c_1)$ is an encryption of an approximate value of $A \cdot m_1$. With a slight abuse of notation, we will write $c_1 + c_2$ as a shorthand for $\mathsf{Add}(c_1, c_2)$ and Ac for $\mathsf{mMult}(A, c)$.

When a plain text being encrypted, it starts from level L and it decreases by one level for each multiplication. For instance, the operation $A \cdot A \cdot c = \mathsf{mMult}(A, (\mathsf{mMult}(A, c)))$ is differ from $A^2 \cdot c = \mathsf{mMult}(A^2, c)$ in the sense that the former consumes one more ciphertext level.

2.3 Finite Lifespan of Encrypted Variable and Bootstrapping

As mentioned in the previous subsection, homomorphic multiplication of a real matrix to a ciphertext decreases its level. Hence, a leveled HE scheme itself only supports homomorphic evaluation of functions with the bounded depth. *Bootstrapping* of ciphertexts, first suggested by Gentry (2009), evaluates the decryption circuit of HE scheme homomorphically on a given ciphertext to obtain an encryption of the same message with reduced noise. Recently, Ducas and Micciancio (2015) suggested an efficient method to homomorphically compute the NAND operation and bootstrap the resulting ciphertext efficiently. Their implementation of bootstrapping on a personal computer runs in about 0.69 sec when given public ciphertexts have the error parameter $\log \alpha^{-1} = 32$.

In this paper, we adapt their idea to leveled HE scheme to refresh a ciphertext, so a ciphertext can be transformed into a new ciphertext of the same message with a larger level. This transformation enables us to carry out homomorphic operations infinitely. The asymptotic complexity of bootstrapping is $\mathrm{poly}(L \log \mathfrak{p})$, and our rudimentary implementation takes several seconds for practical parameters in Section 6.

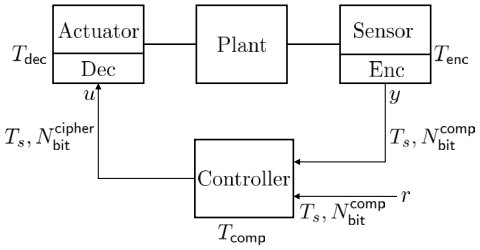

Fig. 2. Configuration of control system with encrypting controller

3. CONFIGURATION OF CONTROL SYSTEM WITH ENCRYPTED CONTROLLER

A discrete-time linear time-invariant controller is considered whose state-space representation is

$$\begin{aligned} x(t+1) &= Ax(t) + B(r(t) - y(t)) \\ u(t) &= Cx(t) + D(r(t) - y(t)) \end{aligned} \tag{1}$$

where $x(t) \in \mathbb{R}^\ell$ is the controller state, $y(t) \in \mathbb{R}^p$ is the plant output (or the controller input), and $u(t) \in \mathbb{R}^m$ is the plant input (or the controller output) with $r(t) \in \mathbb{R}^p$ being the reference command. See Fig. 2. The matrices $A \in \mathbb{R}^{\ell \times \ell}$, $B \in \mathbb{R}^{\ell \times p}$, $C \in \mathbb{R}^{m \times \ell}$, and $D \in \mathbb{R}^{m \times p}$ are controller parameters, or gains, of appropriate dimensions. Suppose that all signals are bounded by some constant $M > 0$. The signals are scaled by $q_0/(2M)$ and rounded before encryption so that each component of $x(t)$, $y(t)$, $r(t)$, and $u(t)$ is represented as an element of the message space $\mathbb{Z}_{q_0} = \mathbb{Z} \cap [-q_0/2, q_0/2)$. We also assume that the gain matrices are stored as in plaintext. Thus, the controller handles encrypted signals $\bar{x}(t) \in \mathbb{Z}_q^{\ell \times (n+1)}$, $\bar{y}(t) \in \mathbb{Z}_q^{p \times (n+1)}$, $\bar{u}(t) \in \mathbb{Z}_q^{m \times (n+1)}$, and $\bar{r}(t) \in \mathbb{Z}_q^{p \times (n+1)}$. The encrypted controller inputs are generated by $\bar{r}(t) = \mathsf{Enc}(\lfloor \frac{q_0}{2M} \cdot r(t) \rceil, sk)$ and $\bar{y}(t) = \mathsf{Enc}(\lfloor \frac{q_0}{2M} \cdot y(t) \rceil, sk)$, and the decrypted plant input is recovered by $\frac{2M}{q_0} \cdot \mathsf{Dec}(\bar{u}(t), sk)$ after decryption.

Hence, the controller dynamics (1) actually run in the ciphertext space as [2]

$$\begin{aligned} \bar{x}(t+1) &\leftarrow A\bar{x}(t) + B(\bar{r}(t) - \bar{y}(t)) \\ \bar{u}(t) &\leftarrow C\bar{x}(t) + D(\bar{r}(t) - \bar{y}(t)). \end{aligned} \tag{2}$$

[2] It can also be expressed as

When a control system with encrypting controller is designed, one has to consider design specifications as:

- $\log \mathfrak{p}$: Number of bits of $\mathfrak{p}$. Especially, it determines the HE error parameter α in the following relation $L \log \mathfrak{p} \approx \log \alpha^{-1}$ where L is the maximal level of ciphertext defined in Section 2. It is also related to the quantization of gain matrices in the controller. During the operation of $\mathsf{mMult}(A, \bar{x})$, the matrix A is scaled and rounded by $\lfloor \mathfrak{p} A \rceil$, which is eventually equivalent to quantization of A with the quantization interval $1/\mathfrak{p}$.
- $N_{\mathsf{bit}}^{\mathsf{cipher}}$: Number of bits that is required to represent one (uncompressed ciphertext) element of the signal $\bar{x}(t)$, $\bar{u}(t)$, $\bar{y}(t)$, and $\bar{r}(t)$. Since it holds that $2^{N_{\mathsf{bit}}^{\mathsf{cipher}}} = q^{n+1}$ where q and n are the HE parameters, we have
$$N_{\mathsf{bit}}^{\mathsf{cipher}} = (n+1)\log q = \Omega\left(\frac{\lambda}{\log \lambda}(L \log \mathfrak{p})^2\right)$$
as indicated in Section 2.
- $N_{\mathsf{bit}}^{\mathsf{comp}}$: Number of bits that is required to represent one (compressed ciphertext) element of the signal $\bar{y}(t)$ and $\bar{r}(t)$. Note that the encrypted signals $\bar{y}(t)$ and $\bar{r}(t)$ can easily be compressed since they are newly generated fresh ciphertexts at each time instant, while $\bar{x}(t)$ and $\bar{u}(t)$ can not. It is approximately obtained from
$$N_{\mathsf{bit}}^{\mathsf{comp}} = \Omega\left(L \log \mathfrak{p}\right).$$
- T_s: Sampling time in seconds. We assume that the whole control system shares the same sampling time.
- I_{capa}: Total network throughput or channel capacity of the communication in bps (bits per second). This represents the rate at which information can be reliably transmitted over a communication channel and can be calculated as
$$I_{\mathsf{capa}} = \frac{1}{T_s}\left(N_{\mathsf{bit}}^{\mathsf{cipher}} m + 2 N_{\mathsf{bit}}^{\mathsf{comp}} p\right).$$
- T_{enc}, T_{dec}: Time in seconds that is consumed for encryption (decryption, resp.) in the sensor (actuator, resp.) block.
- $T_{\mathsf{mult}}^{A,x}$, T_{mult}: Time in seconds that is consumed for all element-wise multiplications occurred between the matrix A and the ciphertext $\bar{x}$. It can be computed from $\mathfrak{p}$, $N_{\mathsf{bit}}^{\mathsf{cipher}}$, and the number of elements in A, i.e., $T_{\mathsf{mult}}^{A,x} \propto N_{\mathsf{bit}}^{\mathsf{cipher}} \ell^2 \log \mathfrak{p}$. For other matrices and ciphertexts, it is similarly defined. In addition, T_{mult} denotes the total time that is consumed to complete all multiplications in (2) and can be calculated as
$$T_{\mathsf{mult}} \propto N_{\mathsf{bit}}^{\mathsf{cipher}} \log \mathfrak{p}(\ell^2 + \ell p + m\ell + mp).$$
- T_{add}: Time in seconds that is consumed to complete all addition operations in (2). This quantity is relatively small compared with T_{mult} so that it is negligible.
- T_{comp}: Time in seconds that is consumed to complete all operations in (2), i.e., $T_{\mathsf{comp}} = T_{\mathsf{mult}} + T_{\mathsf{add}}$. If multiple controller scheme introduced in Section 4 is applied, T_{comp} is proportionally increased to the number of controllers.

If the total elapsed time from measuring the (plaintext) output to generating (plaintext) control input, is less than one sampling time, i.e., $T_s > T_{\mathsf{enc}} + T_{\mathsf{comp}} + T_{\mathsf{dec}}$, then the securing process by HE does not induce any additional time delay from the discrete-time control theory's standpoints.

$$\bar{x}(t+1) \leftarrow \mathsf{Add}(\mathsf{mMult}(A, \bar{x}(t)), \mathsf{mMult}(B, \mathsf{Add}(\bar{r}(t), -\bar{y}(t))))$$
$$\bar{u}(t) \leftarrow \mathsf{Add}(\mathsf{mMult}(C, \bar{x}(t)), \mathsf{mMult}(D, \mathsf{Add}(\bar{r}(t), -\bar{y}(t))))$$

by utilizing functions defined in Section 2.

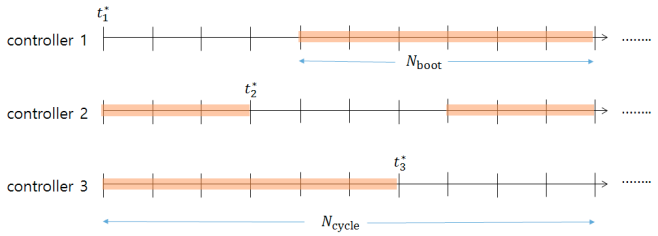

Fig. 3. An example time chart for three controllers ($N_c = 3$) with different t_i^*, in which thick bars indicate the period for bootstrapping.

4. RUNNING MULTIPLE CONTROLLERS: A REMEDY TO BOOTSTRAPPING

Since the encrypted controller state $\bar{x}$ has a finite lifespan (this doesn't apply to other encrypted variables such as $\bar{u}$ or $\bar{y}$ because their computation is not recursive as for $\bar{x}$, as seen in (2)), the bootstrapping process should be performed on $\bar{x}$ when its lifespan is expired, in order to revive it and give a new lifespan. A drawback is that the control input can not be generated because x is not updated during the bootstrapping process (which usually takes more time than T_s). In this section, as a remedy to this problem, we propose running multiple controllers so that, at every time instance, at least one controller is working to generate the control input. For convenience, let

- N_{boot}: Number of samples (measured by T_s) that is required for performing bootstrapping the state $\bar{x}$, which is $\mathsf{ceiling}(($time for bootstrapping$)/T_s)$. Note that $N_{\mathsf{boot}} = \mathrm{poly}(L \log \mathfrak{p})$ as mentioned in Section 2.
- N_{lifespan}: Number of samples (measured by T_s) that takes for a fresh $\bar{x}$ to consume its lifespan by computing (2). With the tree-based multiplication algorithm developed in Section 5, $N_{\mathsf{lifespan}} \approx 2^L$, while the naive algorithm in this section gives $N_{\mathsf{lifespan}} \approx L$.
- $N_{\mathsf{cycle}} := N_{\mathsf{boot}} + N_{\mathsf{lifespan}}$.

Then, the idea begins by an observation from (2) that

$$\bar{x}(t + N_{\mathsf{boot}}) \to A^{N_{\mathsf{boot}}} \bar{x}(t) + \sum_{j=0}^{N_{\mathsf{boot}}-1} A^{N_{\mathsf{boot}}-1-j} B(\bar{r}(t+j) - \bar{y}(t+j)) \quad (3)$$

which is easily derived by recursively solving $\bar{x}(t)$ from (2). Equation (3) implies that, even though $\bar{x}(t)$ is not updated during the bootstrapping process, just one update by (3) with the revived $\bar{x}(t)$ after the process, recovers the necessary information at that time.

Moreover, by running multiple controllers whose bootstrapping occurs at different time windows, one can select the state x from a working controller. Indeed, choose sufficiently large number N_c of controllers and the set of $\{t_i^* : 0 \le t_i^* < N_{\mathsf{cycle}}, 1 \le i \le N_c\}$ like in Fig. 3, where t_i^* is the time instant (modulo N_{cycle}) for the bootstrapping of the i-th controller to be completed, such that one can find at least one controller not in bootstrapping status at each time t.

Suppose that there are N_c controllers running in parallel, and let $\bar{x}^i(t)$ denote the encrypted state of the i-th con-

troller. (If there is enough time for each controller (1) to update sequentially within one processor, and all computation finishes within T_s, then there is no need to rely on parallel computing.) All controllers start at $t = 0$ with zero initial condition, *i.e.,* $\bar{x}^i(0) = 0_{\ell\times(n+1)}$. Note that $\mathsf{Dec}(0_{\ell\times(n+1)}, sk) = 0_{\ell\times 1}$. Then, the proposed strategy is the following.

Generate the control input $\bar{u}$:

At each sampling time t,
for $i = 1$ **to** N_c
  if $(t - t_i^*) \bmod N_{\mathsf{cycle}} < N_{\mathsf{lifespan}}$
    $\bar{x}^* \leftarrow \bar{x}^i$
  endif
endfor
$\bar{u} \leftarrow C\bar{x}^* + D(\bar{r} - \bar{y})$

And, for next sample, perform the following for all N_c controllers.

Operation of the i-th controller:

initialization: Let $m_i := N_{\mathsf{boot}}$ if $t_i^* = 0$, $m_i := \min\{t_i^*, N_{\mathsf{boot}}\}$ otherwise. Store $W_{0k} := A^{N_{\mathsf{boot}}-k}B$ $(k = 1, \ldots, N_{\mathsf{boot}} - 1)$, $Z_0 := A^{N_{\mathsf{boot}}}$, $W_{ik} := A^{m_i-k}B$ $(k = 1, \cdots, m_i - 1)$, and $Z_i := A^{m_i}$.
Set $\bar{x}^i \leftarrow 0_{\ell\times(n+1)}$, $\bar{w}^i \leftarrow 0_{\ell\times(n+1)}$, $j^i \leftarrow N_{\mathsf{cycle}} - t_i^*$.

At each sampling time t,
if $j^i < N_{\mathsf{lifespan}}$
  $\bar{x}^i \leftarrow A\bar{x}^i + B(\bar{r} - \bar{y})$
  $j^i \leftarrow j^i + 1$
elseif $j^i < N_{\mathsf{cycle}}$
  if $j^i = N_{\mathsf{lifespan}}$, perform bootstrapping of $\bar{x}^i$
  $\bar{w}^i \leftarrow \bar{w}^i + W_{i(N_{\mathsf{cycle}}-j^i)}(\bar{r} - \bar{y})$
  $j^i \leftarrow j^i + 1$
else
  $\bar{x}^i \leftarrow Z_i\bar{x}^i + \bar{w}^i + B(\bar{r} - \bar{y})$
  $j^i \leftarrow 0$, $\bar{w}^i \leftarrow 0_{\ell\times(n+1)}$,
  $Z_i \leftarrow Z_0$, $W_{ik} \leftarrow W_{0k}$ $(k = 1, \ldots, N_{\mathsf{boot}} - 1)$
endif

5. KEEPING MULTIPLICATION TREE: A REMEDY TO SHORT LIFESPAN UNDER MULTIPLICATION

The algorithm for operation of a controller in the previous section computes the samples iteratively. It determines the lifespan N_{lifespan} proportional to the maximal level of ciphertext L because the matrix A is multiplied i times to $\bar{x}(t)$ to compute $\bar{x}(t + i)$. If that is the case, the bootstrapping requires $N_{\mathsf{boot}} = \mathrm{poly}(L \log \mathfrak{p})$ complexity, which is much larger than the lifespan of controllers. In this section, the *multiplication tree* structure is introduced to increase the lifespan of encrypted variables up to 2^{L-1} while maintaining the number of matrices multiplied to the input value at L times.

For an $(\ell \times \ell)$ matrix A and ℓ-dimensional vectors $z_0, \ldots, z_{i-1}$, let $\mathcal{P}_i(z_0, \ldots, z_{i-1}) := \sum_{j=0}^{i-1} A^{i-j-1}z_j$. It is easy to check the equality $\mathcal{P}_{i+j}(z_0, \ldots, z_{i+j-1}) = A^j \cdot \mathcal{P}_i(z_0, \ldots, z_{i-1}) + \mathcal{P}_j(z_i, \ldots, z_{i+j-1})$. If one takes $z_j \leftarrow B(r(t+j) - y(t+j))$ for $0 \le j < i$, then $x(t+i) = A^i x(t) + \mathcal{P}_i(z_0, \ldots, z_{i-1})$ is obtained for all j as in (3). Hence, $x(t)$ can be updated to $x(t + i)$ directly using the 'catch-up' vector $\mathcal{P}_i(z_0, \ldots, z_{i-1})$.

The $\textsc{mTree}_h$ is an algorithm which computes the encryptions of $\mathcal{P}_1(z_0), \ldots, \mathcal{P}_{2^h}(z_0, \ldots, z_{2^h-1})$ iteratively with given encryptions $\bar{z}_0, \ldots, \bar{z}_{2^h-1}$. It is required to store at most h ciphertexts during algorithm, and the level of ciphertext is decreased by at most h compared to the input ciphertexts. This algorithm can be applied to

1. compute the controller state $\bar{x}(t + i) = A^i\bar{x}(t) + \bar{\mathcal{P}}_i(z_0, \ldots, z_{i-1})$ for $0 < i \le 2^h$ during lifespan of $\bar{x}(t)$.
2. generate an encryption of catch-up vector to update the refreshed ciphertext $\bar{x}(t)$ to $\bar{x}(t + 2^h)$.

procedure $\textsc{mTree}_0(\bar{z}_0)$\{
  $\bar{p}_0 \leftarrow \bar{z}_0$
  return $(\bar{p}_0)$
\}
procedure $\textsc{mTree}_{k+1}(\bar{z}_0, \ldots, \bar{z}_{2^{k+1}-1})$\{
  run $(\bar{p}_0, \ldots, \bar{p}_{2^k-1}) \leftarrow \textsc{mTree}_k(\bar{z}_0, \ldots, \bar{z}_{2^k-1})$ and return $(\bar{p}_0, \ldots, \bar{p}_{2^k-1})$ iteratively
  keep $\bar{p}_{2^k-1}$
  run $(\bar{p}'_{2^k}, \ldots, \bar{p}'_{2^{k+1}-1}) \leftarrow \textsc{mTree}_k(\bar{z}_{2^k}, \ldots, \bar{z}_{2^{k+1}-1})$ and return $(\bar{p}_{2^k}, \ldots, \bar{p}_{2^{k+1}-1}) = (A\bar{p}_{2^k-1} + \bar{p}'_{2^k}, \ldots, A^{2^k}\bar{p}_{2^k-1} + \bar{p}'_{2^{k+1}-1})$ iteratively
\}
run $\textsc{mTree}_h(\bar{z}_0, \ldots, \bar{z}_{2^h-1})$

The correctness of this algorithm can be shown inductively as follows. It is clear that $\bar{p}_0 = \bar{z}_0$ is an encryption of $\mathcal{P}_1(z_0) = z_0$ in the same level. If $\bar{p}_i$ and $\bar{p}'_{2^k+i}$ are encryptions of $\mathcal{P}_{i+1}(z_0, \ldots, z_i)$ and $\mathcal{P}_{i+1}(z_{2^k}, \ldots, z_{2^k+i})$ respectively for all $0 \le i < 2^k$, then $\bar{p}_{2^k+i} = A^{i+1}\bar{p}_{2^k-1} + \bar{p}'_{2^k+i}$ is an encryption of

$$A^{i+1}\mathcal{P}_{2^k}(z_0, \ldots, z_{2^k-1}) + \mathcal{P}_{i+1}(z_{2^k}, \ldots, z_{2^k+i}) = \mathcal{P}_{2^k+i+1}(z_0, \ldots, z_{2^k+i}),$$

as desired. The algorithm $\textsc{mTree}_{k+1}(\cdot)$ requires additional memory to keep one ciphertext $\bar{p}_{2^k-1}$ compared to $\textsc{mTree}_k(\cdot)$, hence one needs to store at most h-number of ciphertexts during $\textsc{mTree}_h(\bar{z}_0, \ldots, \bar{z}_{2^h-1})$. Moreover, output ciphertexts of $\textsc{mTree}_{k+1}$ have the level decreased by at most 1 from the output ciphertexts of $\textsc{mTree}_k$. So output ciphertexts of $\textsc{mTree}_h(\cdot)$ consume at most h levels, and the level of the controller state $\bar{x}(t + i)$ is larger than $L - h$ for $0 < i \le 2^h$. It needs one more level to compute $\bar{u}(t + i)$ using the state $\bar{x}(t + i)$, thus it is enough to set the maximal level $L = h + 1$.

6. SIMULATION STUDY

In this section, simulation results of encrypting controller are illustrated with a quadruple water tank system introduced in (Johansson, 2000) whose dynamics is given by

$$\begin{aligned}
\frac{dh_1}{dt} &= -\frac{a_1}{A_1}\sqrt{2gh_1} + \frac{a_3}{A_1}\sqrt{2gh_3} + \frac{\gamma_1 k_1}{A_1}v_1 \\
\frac{dh_2}{dt} &= -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{a_4}{A_2}\sqrt{2gh_4} + \frac{\gamma_2 k_2}{A_2}v_2 \\
\frac{dh_3}{dt} &= -\frac{a_3}{A_3}\sqrt{2gh_3} + \frac{(1-\gamma_2)k_2}{A_3}v_2 \\
\frac{dh_4}{dt} &= -\frac{a_4}{A_4}\sqrt{2gh_4} + \frac{(1-\gamma_1)k_1}{A_4}v_1 \\
y_1 &= k_c h_1, \quad y_2 = k_c h_2
\end{aligned} \tag{4}$$

where v_1 and v_2 are the control inputs (voltage to the pumps), y_1 and y_2 are the measurement outputs (voltages from water level measurement), and h_i is the state variable

(water level). Moreover, all necessary parameter values are given in Table 1. A decentralized PI controller is designed for a linearized model of (4) around the operating points $h_1^0 = 12.4$ cm, $h_2^0 = 12.7$ cm, $h_3^0 = 1.8$ cm, $h_4^0 = 1.4$ cm, $v_1^0 = 3.00$ V, $v_2^0 = 3.00$ V (Johansson, 2000). By simple calculations, the controller takes the form of (1) with gains

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ C = \begin{bmatrix} 0.05 & 0 \\ 0 & 0.034 \end{bmatrix},\ D = \begin{bmatrix} 3.025 & 0 \\ 0 & 2.717 \end{bmatrix}.$$

The computed control input is then decrypted in the actuator and generates continuous signal by the zero order holder.

Table 1. Parameter description

A_1, A_3	$28[cm^2]$	cross-section of tank
A_2, A_4	$32[cm^2]$	cross-section of tank
a_1, a_3	$0.071[cm^2]$	cross-section of outlet hole
a_2, a_4	$0.057[cm^2]$	cross-section of outlet hole
k_c	$0.50[V/cm]$	output gain
g	$981[cm/s^2]$	acceleration of gravity
(k_1, k_2)	$(3.33,\ 3.35)[cm^3/Vs]$	input gain
(γ_1, γ_2)	$(0.70,\ 0.60)$	input ratio

All controller parameters are set to 8-bit fixed point numbers (i.e., $\mathfrak{p} = 2^8$), and all signals quantized to 9-bit fixed point numbers (i.e., $q_0 = 2^{10}, M = 2$) with sampling time $T_s = 0.5$ sec. The level of tree-based algorithm is $h = 9$ and thus, the HE parameter is obtained as $n = 2375$. In addition, output y and reference signal r are transmitted in compressed ciphertexts that is $N_{\mathsf{bit}}^{\mathsf{comp}} = 15$ Bytes for each data and the number of bits for uncompressed ciphertext is $N_{\mathsf{bit}}^{\mathsf{cipher}} = 28.5$ kB. The controllers have been repeatedly activated for 256 sec and bootstrapped for 35 sec, so only two encrypting controllers, which have identically same dynamics and initial conditions, are needed for the implementation ($N_c = 2$). It can also be checked that $T_s > T_{\mathsf{enc}} + T_{\mathsf{comp}} + T_{\mathsf{dec}}$ which implies that we have enough time for HE process in one sampling time T_s so that it does not induce any additional time delay in the control loop.

With all the parameters given above, the encrypting controller scheme successfully operates as depicted in Fig. 4, that is, the outputs track the reference signal well with small errors. It also shows that two controllers $c1$ and $c2$ are synchronized and are smoothly switched over from $c1$ to $c2$ at $t = 256$ sec without any transient peak.

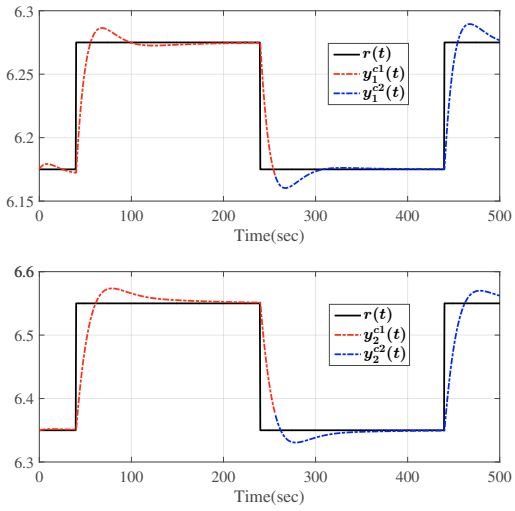

Fig. 4. Reference and output signals near operating point

REFERENCES

S. Amin, A. A. Cárdenas, A and S. S. Sastry. Safe and secure networked control systems under denial-of-service attacks. In *Hybrid Systems: Computation and Control*, pp. 31–45, 2009.

B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Public-key cryptosystems from the worst-case shortest vector problem: Extended abstract. In *Proc. of CRYPTO'09*, pp. 595–618, 2009.

J. H. Cheon, A. Kim, M. Kim and Y. Song. Floating-point homomorphic encryption. In *IACR Cryptology ePrint Archive*, 2016:421, 2016.

J. H. Cheon and D. Stehlé. Fully homomorphic encryption over the integers revisited. In *Proc. of EUROCRYPT'15*, pp. 513–536, 2015.

M. Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. In *Proc. of EUROCRYPT'10*, pp. 24–43, 2010.

L. Ducas and D. Micciancio. FHEW: Bootstrapping homomorphic encryption in less than a second. In *Proc. of EUROCRYPT'15*, pp. 617–640, 2015.

T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proc. of CRYPTO'84*, pp. 10–18, 1984.

C. Gentry. *A fully homomorphic encryption scheme*. Ph.D. dissertation, Stanford University, 2009.

K. H. Johansson. The quadruple-tank process: A multivariable laboratory process with an adjustable zero. *IEEE trans. on Control Systems Technology*, vol. 8, no. 3, pp. 456–465, 2000.

K. Kogiso and T. Fujita. Cyber-security enhancement of networked control systems using homomorphic encryption. In *Proc. of IEEE 54th Conf. on Decision and Control*, pp. 6836–6843, 2015.

R. Langner. Stuxnet: Dissecting a cyberwarfare weapon. *IEEE Security & Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

C. Lee, H. Shim, and Y. Eun. Secure and robust state estimation under sensor attacks, measurement noise and process disturbances: Observer-based combinatorial approach. In *Proc. of 14th European Control Conf.*, pp. 1872–1877, 2015.

F. Li, B. Luo, and P. Liu. Secure information aggregation for smart grids using homomorphic encryption. In *Proc. of 1st IEEE Int. Conf. on SmartGridComm*, pp. 327–332, 2010.

R. Lindner and C. Peikert. Better key sizes (and attacks) for LWE-based encryption In *Proc. of CT-RSA'11*, pp. 319–339, 2011.

D. Micciancio and O. Regev. Lattice-based Cryptography. *Post-Quantum Cryptography*, pp. 147–191, 2009.

G. Park, H. Shim, C. Lee, Y. Eun, K. H. Johansson. When adversary encounters uncertain cyber-physical systems: Robust zero-dynamics attack with disclosure resources. To be presented at *IEEE Conf. on Decision and Control*, Las Vegas, 2016.

F. Pasqualetti, F. Dorfler, and F. Bullo. Attack detection and identification in cyber-physical systems. *IEEE Trans. on Automatic Control*, vol. 58, no. 11, pp. 2715–2729, 2013.

C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem: Extended abstract. In *Proc. of STOC'09*, pp. 333–342, 2009.

O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *Proc. of STOC'05*, pp. 84–93, 2005.

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

J. Slay and M. Miller. *Lessons learned from the maroochy water breach*, Springer, 2007.

S. Sundaram and C. N. Hadjicostis. Distributed function calculation via linear iterative strategies in the presence of malicious agents. *IEEE Trans. on Automatic Control*, vol. 56, no. 7, pp. 1498–1508, 2011.

A. Teixeira, I. Shames, H. Sandberg, and K. H. Johansson. Revealing stealthy attacks in control systems. In *Proc. 50th Annu. Allerton Conf. Communication, Control, Computing*, pp. 1806-–1813, 2012.

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

Q. Zhu and T. Basar. Game-theoretic methods for robustness, security, and resilience of cyberphysical control systems: Games-in-games principle for optimal cross-layer resilient control systems. *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 46–65, 2015.