INL/CON-24-78338-Revision-0

End-to-End Encryption for Cyber-Physical Systems Using Fully Homomorphic Encryption

April 2024

Robert s Lois II

Changing the World's Energy Future

Idaho National Laboratory

INL/CON-24-78338-Revision-0

End-to-End Encryption for Cyber-Physical Systems Using Fully Homomorphic Encryption

Robert s Lois II

April 2024

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov

Prepared for the
U.S. Department of Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

End-to-End Encryption for Cyber-Physical Systems Using Fully Homomorphic Encryption

Robert Lois
robert.lois@pitt.edu
MFANS 2024

PhD Candidate – University of Pittsburgh
INL Graduate Fellow – National and Homeland Security

Idaho National Laboratory

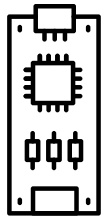
The goal of this research is to thwart cyber adversaries to meaningful ends by integrating fully homomorphic encryption schemes into control systems.

If this research is successful, then we will be to do the following:

Integrate fully homomorphic encryption into network/cloud-based control systems.

MFANS2024

Analyze real-time characteristics to maintain proper operation of encrypted control systems.



Realize design on embedded/cloud-based hardware that is controlling a physical (or virtual) asset.

The goal of this research is to thwart cyber adversaries to meaningful ends by integrating fully homomorphic encryption schemes into control systems.

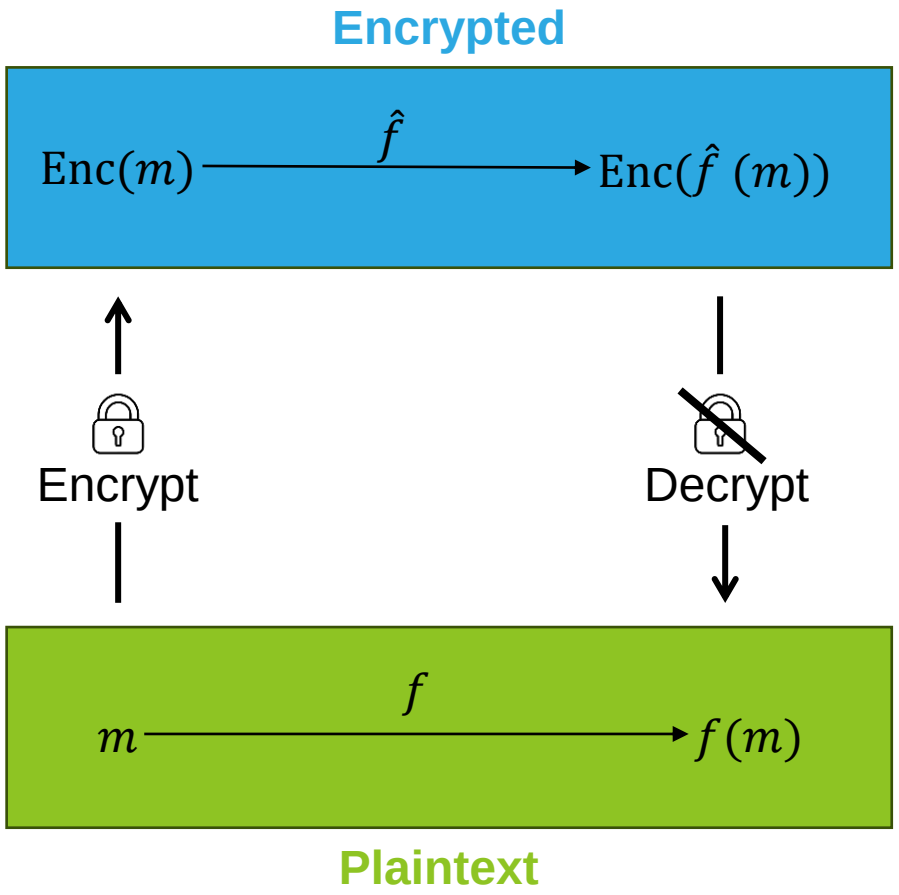
If this research is successful, then we will be to do the following:

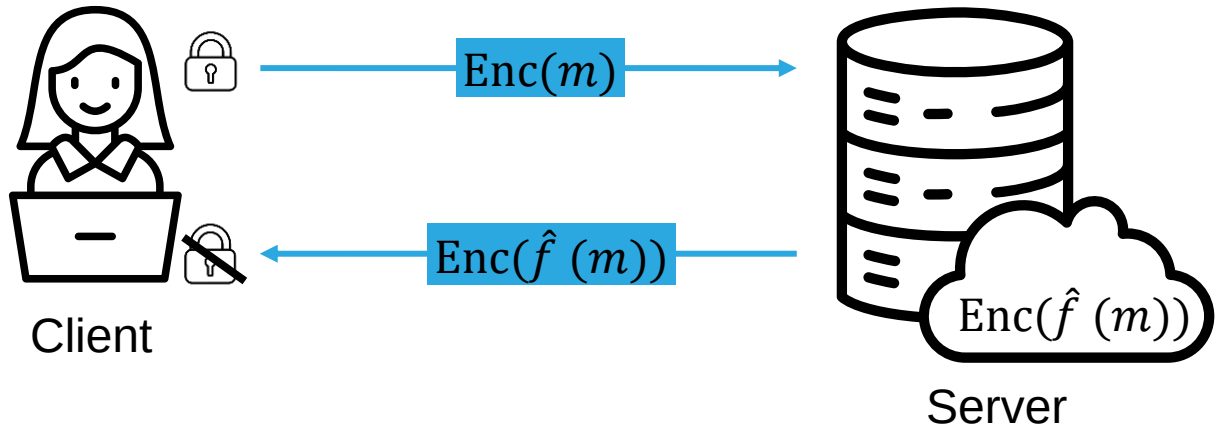
Integrate fully homomorphic encryption into network/cloud-based control systems.

MFANS2024

Musing: How can formal methods help integrate these two systems?

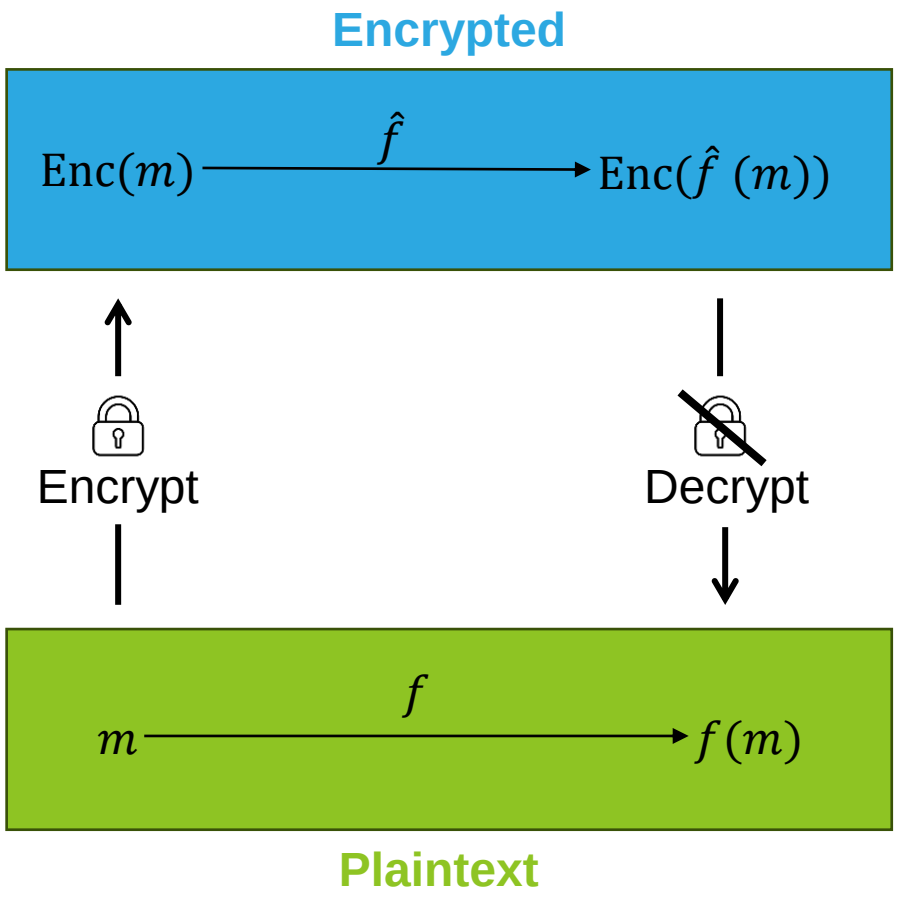
Fully homomorphic encryption secures data while preserving its utility, enabling privacy-preserving applications for sensitive data and functions.

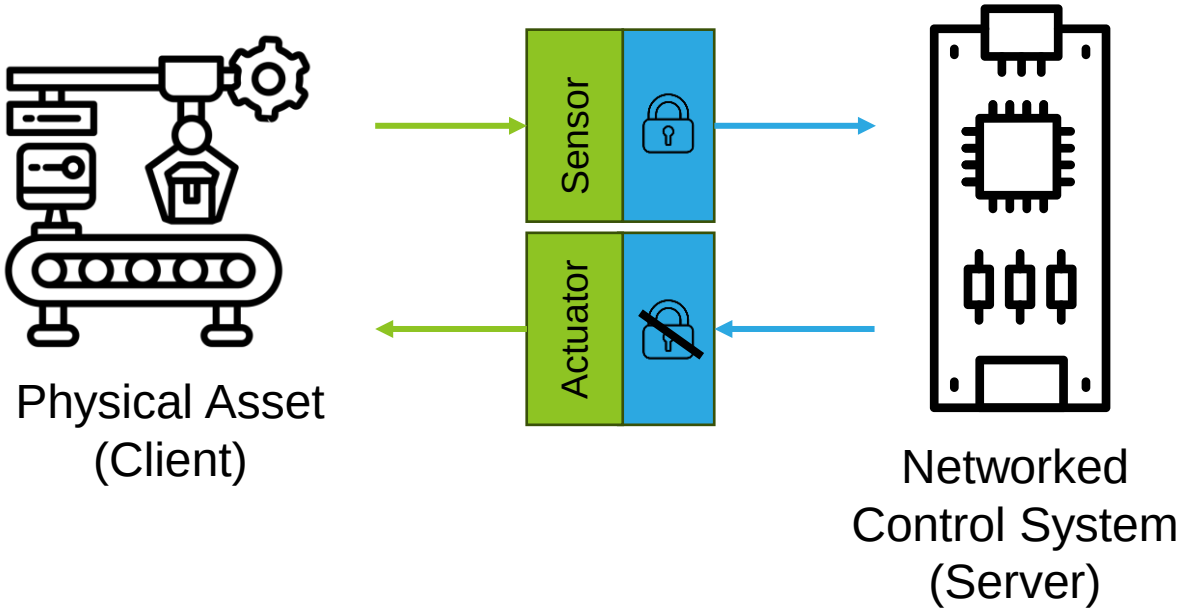




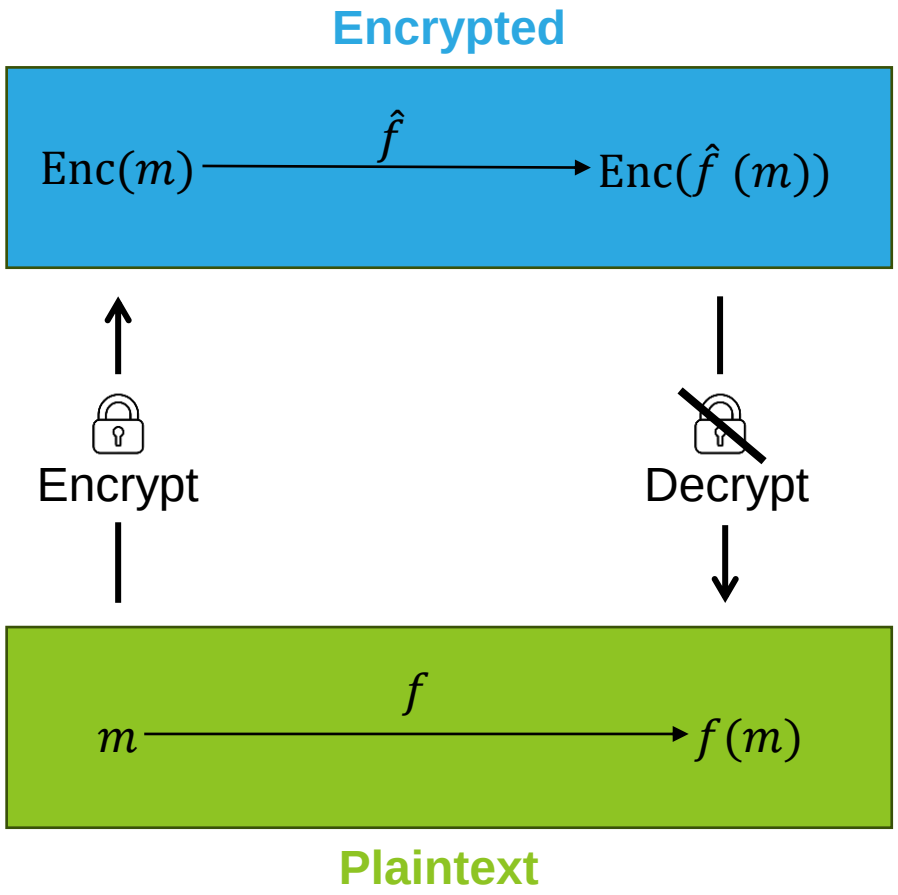
SoK: Cryptographic Neural Network Computation
https://sokcryptonn.github.io/

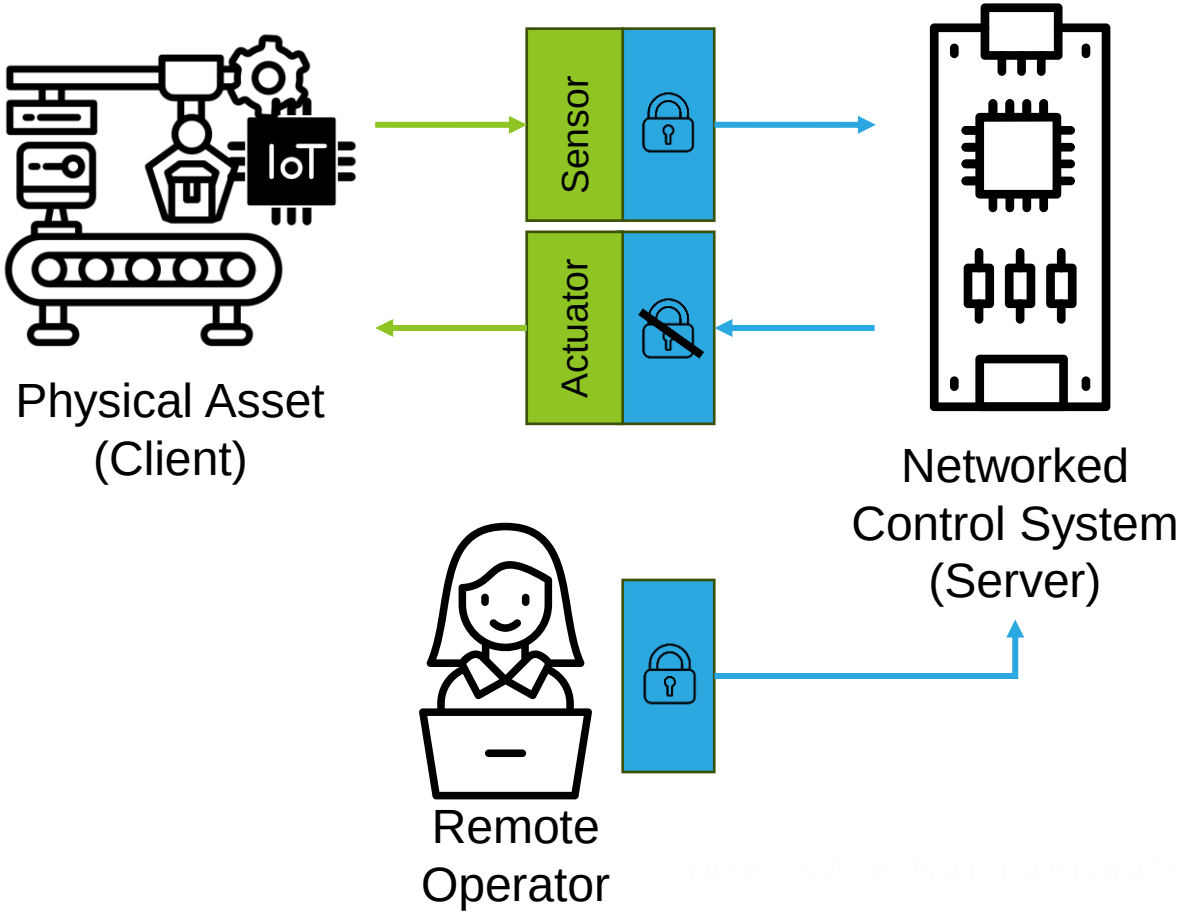
Using fully homomorphic encryption, we can perform any operations on encrypted signals without the need to decrypt it.



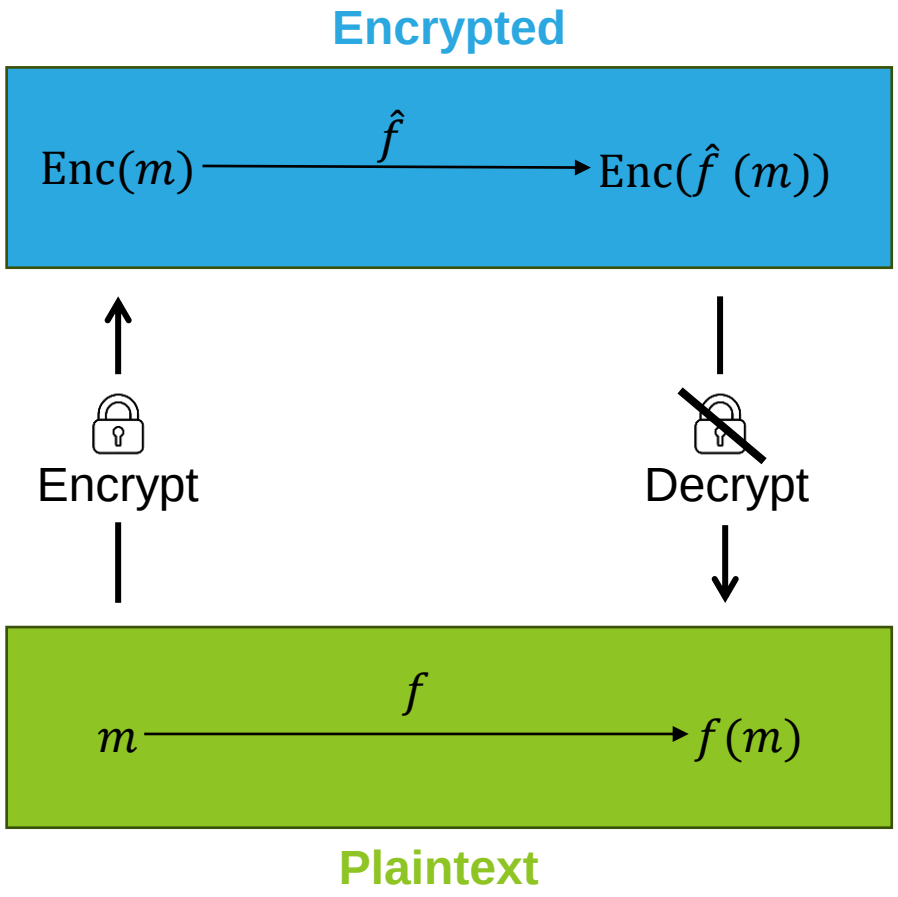


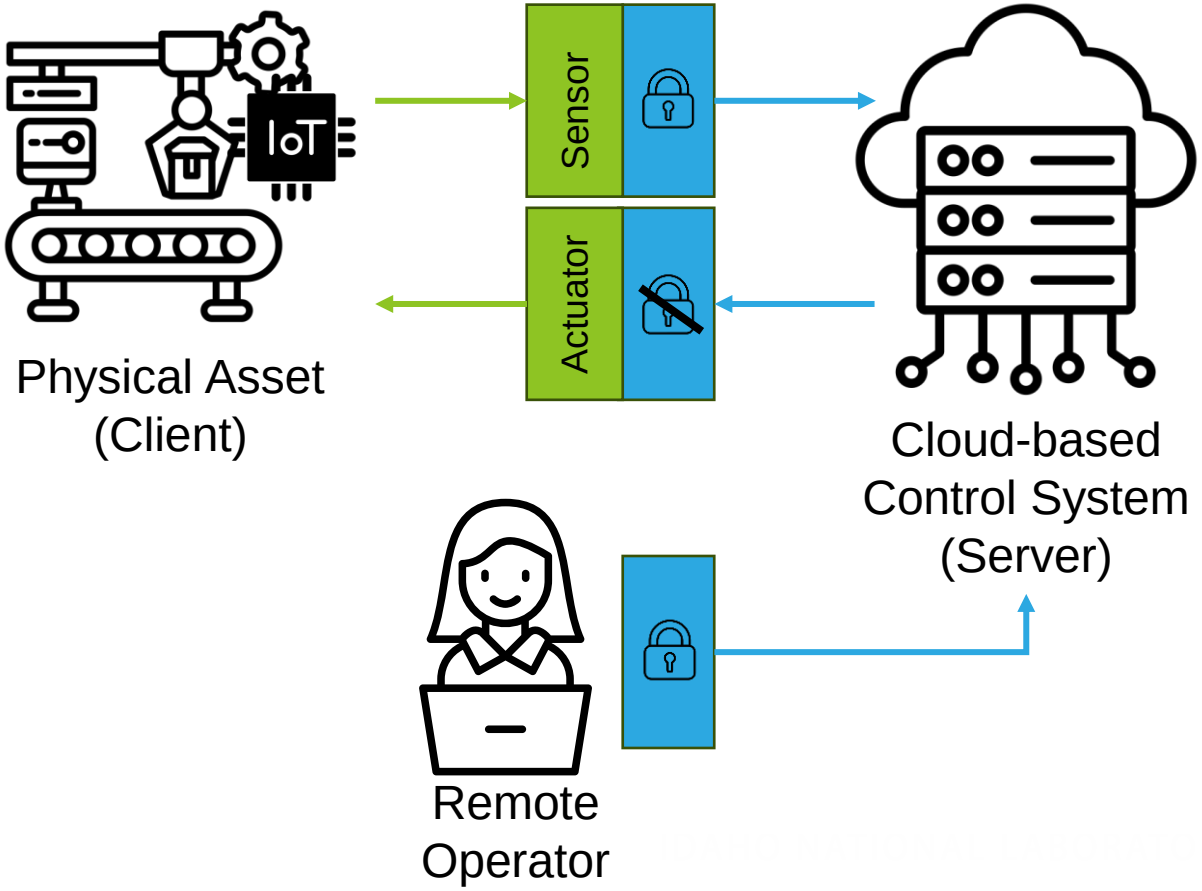
Using fully homomorphic encryption, we can perform any operations on encrypted signals without the need to decrypt it.



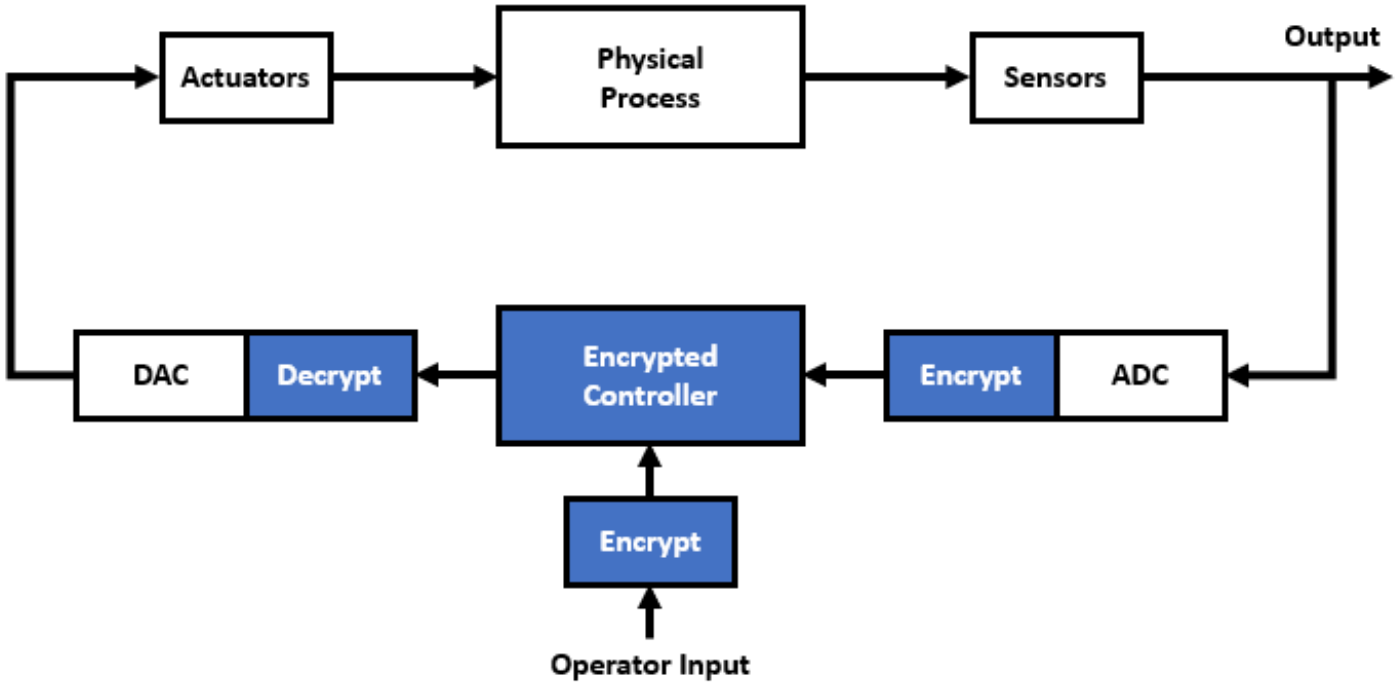


Using fully homomorphic encryption, we can perform any operations on encrypted signals without the need to decrypt it.





In order to integrate FHE into control systems, signals and controller matrices are required to be integers.



Problem Statement

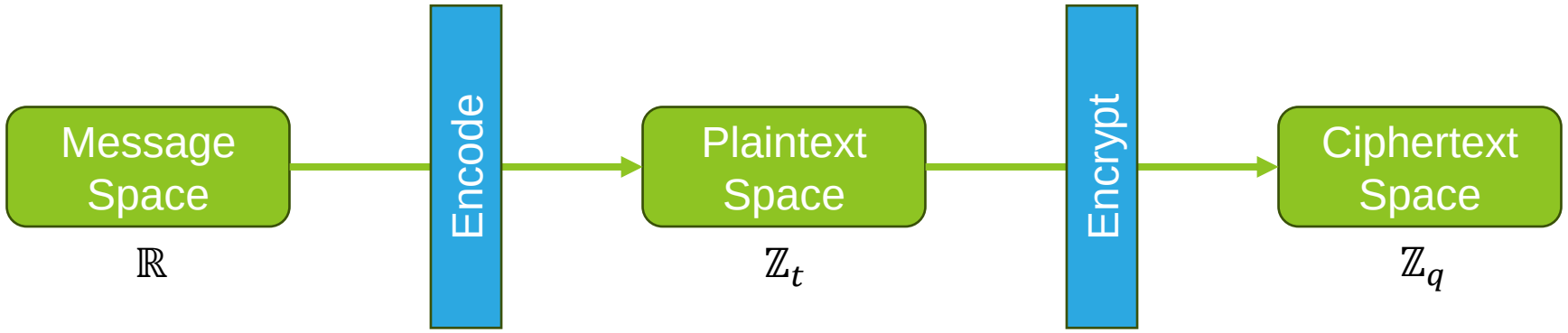
Fully encrypt controller dynamics and signals from sensor, through the controller, to the actuator.

Plant:
$$x^p_{k+1} = Ax^p_k + Bu_k$$
$$y_k = Cx^p_k + Du_k$$

Controller:
$$x_{k+1} = A_d x_k + B_d e_k$$
$$u_k = C_d x_k + D_d e_k$$

$\in \mathbb{R}$

All FHE schemes encode the message space to an algebraic structure prior to encryption.



Research Question: Can we exploit the encoding function in LWE and the gadget decomposition tool to achieve a function $f: \mathbb{R} \to \mathbb{Z}$?

We propose a modified encoding function and gadget decomposition tool to achieve a fully encrypted control system

Encrypting Controller Input

$$\hat{x}_k = \mathrm{LWE}(x_k, sk) = [v + \Delta y_k \; A]_q$$

$$v = [-A \cdot sk + e]_q$$

Encrypting System Matrices

$$\bar{A} = \mathrm{trunc}(A, \log_b(\delta))$$

$$\hat{A} = \mathrm{GSW}(\bar{A}, sk) = \left[\bar{A} \cdot G^* + \mathrm{LWE}\left(0^{\log(q)(N+1)\times 1}\right)\right]_q$$

$$G^* = \mathbb{I}^{N+1} \otimes R^{*F}$$

$$R^* = [\delta, \delta b, \delta b^2, \ldots, \delta b^{F-1}] \text{ for } F = (N+1)\lfloor \log_b q \rfloor$$

External Product

$$c_{mult} = G^{*-1}(\hat{x}_k)\hat{A}$$

Decrypting Controller Output

$$\mathrm{Dec}(c, sk) = \frac{1}{\delta}\left\lceil \frac{c \cdot s}{\Delta} \right\rfloor_q$$

Fully Encrypted Control System

$$x_{k+1}^p = Ax_k^p + Bu_k$$

$$\hat{y}_k = \mathrm{LWE}(Cx_k^p, sk)$$

$$\hat{x}_{k+1} = G^{*-1}(\hat{x}_k) \cdot \hat{A} + G^{*-1}(\hat{r}_k) \cdot \hat{B} - G^{*-1}(\hat{y}_k) \cdot \hat{B}$$

$$u_k = \mathrm{Dec}(\hat{C}\hat{x}_k, sk)$$

Modern FHE schemes are based on the Learning With Errors problem, which is post-quantum secure.

Learning WITHOUT Errors

Suppose we have a secret vector: $s = (s_1, s_2, \ldots, s_n) \in \mathbb{Z}^n$. We have a set of linear equations from sampled points from a lattice that yields $As = d$:

$$\begin{aligned} a_{1,1}s_1 + a_{1,2}s_2 + \cdots + a_{1,n}s_n &= d_1 \\ a_{2,1}s_1 + a_{2,2}s_2 + \cdots + a_{2,n}s_n &= d_2 \\ &\vdots \\ a_{n,1}s_1 + a_{n,2}s_2 + \cdots + a_{n,n}s_n &= d_n \end{aligned}$$

Problem Statement: Can we **learn** the **secret vector?**

Answer: Yes! **Gaussian elimination** will easily provide a solution in **polynomial time.**

Learning the secret vector by injecting noise (or errors) makes the problem nearly impossible.

Learning With Errors

Suppose we have a secret vector: $s = (s_1, s_2, \ldots, s_n) \in \mathbb{Z}^n$ and sampled small error values $e \in \mathbb{Z}^n$. We have a set of linear equations from sampled lattice points:

$$
\begin{aligned}
a_{1,1}s_1 + a_{1,2}s_2 + \cdots + a_{1,n}s_n + e_1 &= d_1 \\
a_{2,1}s_1 + a_{2,2}s_2 + \cdots + a_{2,n}s_n + e_2 &= d_2 \\
&\vdots \\
a_{n,1}s_1 + a_{n,2}s_2 + \cdots + a_{n,n}s_n + e_n &= d_n
\end{aligned}
$$

Problem Statement: Can we **learn** the **secret vector** from this set of noisy equations**?**

Answer: There is no polynomial time algorithm that approximates lattice problems to within polynomial factors (Regev09) – LWE is an average-case hard.

We will integrate an LWE encryption scheme that is suitable for network/cloud-based control systems.

Suppose we wanted to encrypt a sensor measurement denoted y_k.

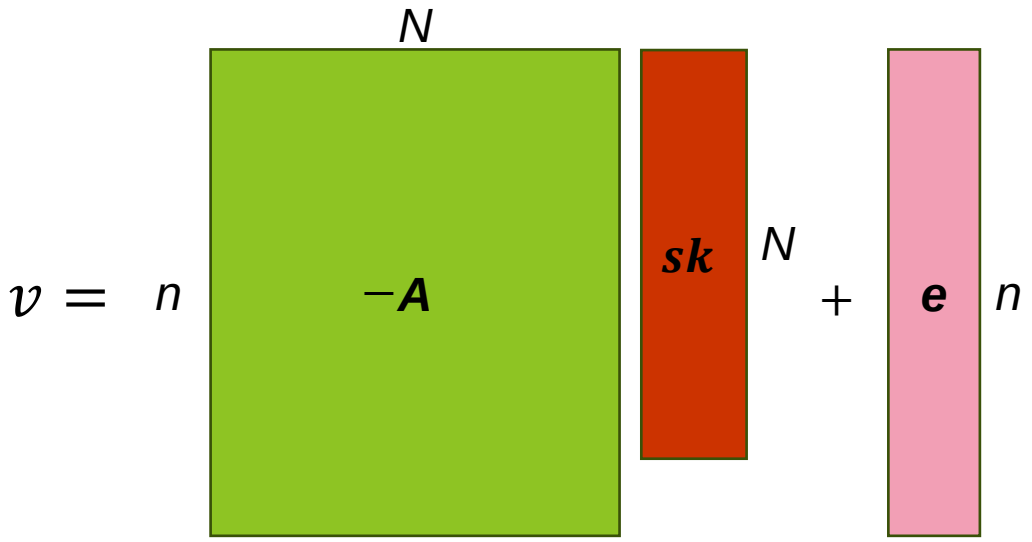
Secret Key: $sk \leftarrow \mathbb{Z}_q^N \qquad s = \begin{bmatrix} 1 \\ sk \end{bmatrix} \in \mathbb{Z}_q^{N+1}$

Public Key: $A \leftarrow \mathbb{Z}_q^{nxN} \qquad v = [-A \cdot sk + e]_q$

$e \sim N_d^n(0, \alpha) \qquad pk = [v, A] \in \mathbb{Z}_q^{n\times(N+1)}$

Encryption: $c = \mathrm{LWE}(y_k) = \left[v + \left\lfloor \frac{q}{t} \right\rfloor y_k, A\right]_q \in \mathbb{Z}_q^{n\times(N+1)}$

Decryption: $\mathrm{Dec}(c) = \left\lceil \frac{cs}{\lfloor q/t \rfloor} \right\rfloor_q = y_k$



Unpacking Decryption

$$\left\lceil \frac{cs}{\lfloor q/t \rfloor} \right\rfloor_q = \left\lceil \frac{\left\lceil -A\cdot sk + e + \left\lfloor \frac{q}{t} \right\rfloor y_k, A \right\rceil \cdot \begin{bmatrix} 1 \\ sk \end{bmatrix}}{\lfloor q/t \rfloor} \right\rfloor_q = \left\lceil \frac{-A\cdot sk + e + \left\lfloor \frac{q}{t} \right\rfloor y_k + A\cdot sk}{\lfloor q/t \rfloor} \right\rfloor_q$$

$$= \lceil y_k + e \rfloor_q = y_k$$

We will integrate an LWE encryption scheme that is suitable for network/cloud-based control systems.

Suppose we wanted to encrypt a sensor measurement denoted y_k.

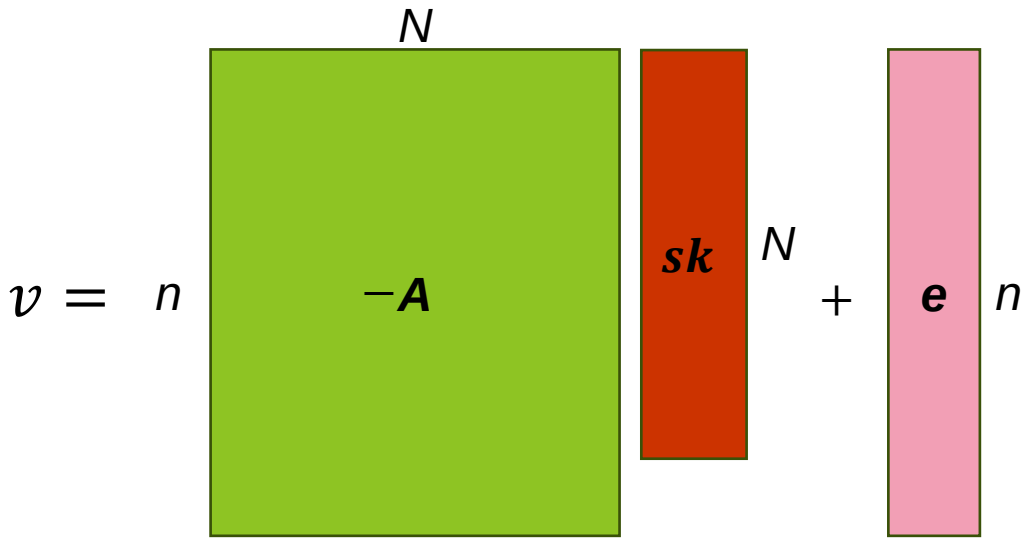
Secret Key: $sk \leftarrow \mathbb{Z}_q^N \qquad s = \begin{bmatrix} 1 \\ sk \end{bmatrix} \in \mathbb{Z}_q^{N+1}$

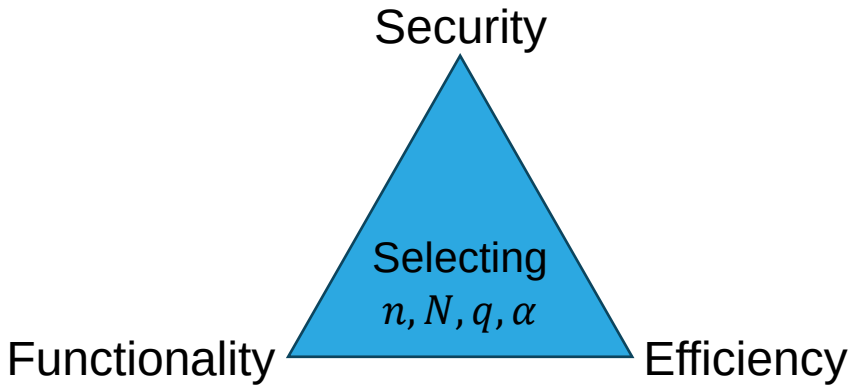
Public Key: $A \leftarrow \mathbb{Z}_q^{nxN} \qquad v = [-A \cdot sk + e]_q$

$e \sim N_d^n(0, \alpha) \qquad pk = [v, A] \in \mathbb{Z}_q^{n \times (N+1)}$

Encryption: $c = \mathrm{LWE}(y_k) = \left[v + \left\lfloor \frac{q}{t} \right\rfloor y_k, A\right]_q \in \mathbb{Z}_q^{n \times (N+1)}$

Decryption: $\mathrm{Dec}(c) = \left\lceil \frac{cs}{\lfloor q/t \rfloor} \right\rfloor_q = y_k$





We will exploit the homomorphic operations that are derived from LWE – this includes addition and multiplication.

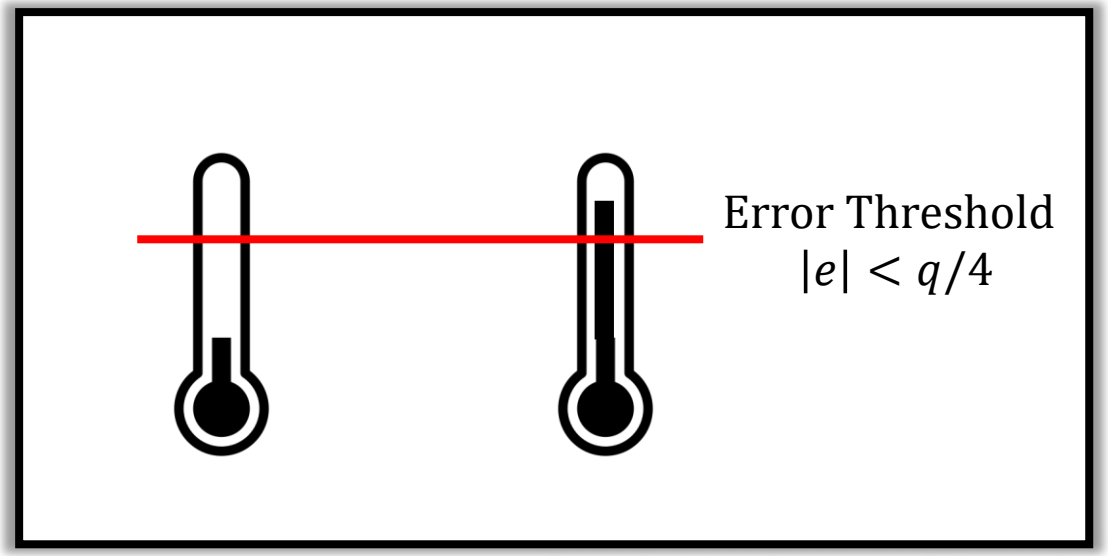
Homomorphic Addition

$$c_1 + c_2 = [(-A_1 sk + \lfloor q/t \rfloor m_1 + e_1), A_1] + [(-A_2 sk + \lfloor q/t \rfloor m_2 + e_2), A_2]_q$$

$$= [(-(A_1 + A_2) sk + \lfloor q/t \rfloor (m_1 + m_2) + (e_1 + e_2)), A_1 + A_2]_q$$

Note: injected noise accumulates with homomorphic operations!

If noise accumulates passed a *threshold*, then the probability of an incorrect decryption increases.



Homomorphic multiplication is performed through an external product between the Gentry-Sahai-Waters (GSW) and LWE ciphertext.

Homomorphic Multiplication

Gentry 2013: Can we construct an LWE-based FHE scheme with a *natural* multiplication procedure i.e. $c_1 * c_2$?

New encryption method: $\mathbb{C}_2 = \mathrm{GSW}(m_2) = m_2 \cdot G + \mathrm{LWE}\left(0^{\log(q)(N+1)\times 1}\right) \epsilon\ \mathbb{Z}_q^{\log(q)(N+1)\times(N+1)}$

- $G = \left[10^0, 10^1, \ldots, 10^{\log(q-1)}\right]^T \otimes \mathbb{I}^{N+1}$ → size is $\log(q)(N+1) \times (N+1)$
- $\mathbb{O} = \mathrm{LWE}\left(0^{\log(q)(N+1)\times 1}\right) \epsilon\ \mathbb{Z}_q^{\log(q)(N+1)\times(N+1)}$
  - Each row is an encryption of 0 but are all unique due to the randomness of $A \leftarrow \mathbb{Z}_q^{nxN}$ and $e \sim N_d^n(0, \alpha)$.
- Decomposition Function $G^{-1}: \mathbb{Z}_q^{n\times(N+1)} \rightarrow \mathbb{Z}_q^{n\times\log(q)(N+1)}$
  - $G^{-1}(\zeta) = [\zeta_0, \zeta_1, \ldots, \zeta_{\log(q-1)}]$
  - Any vector $\zeta\ \in \mathbb{Z}_q^{1\times(N+1)}$ is represented by a radix of 10.
  - $\zeta = \sum_{i=0}^{\log(q-1)} \zeta_i \cdot 10^i$ → $\zeta = G^{-1}(\zeta)G$

$c_1 = \mathrm{LWE}(m_1)\ \epsilon\ \mathbb{Z}_q^{n\times(N+1)}$ → same encryption procedure as before.

$$c_1\mathbb{C}_2 = G^{-1}(c_1)\mathbb{C}_2\ \in \mathbb{Z}_q^{n\times(N+1)}$$

We can perform an external product between real numbers that have been shifted using the modified gadget tool.

Recall from Previous Slide

$$G^* = \mathbb{I}^{N+1} \otimes R^{*F}$$

$$R^* = [\delta, \delta b, \delta b^2, \dots, \delta b^{F-1}]$$
for $F = (N+1)\lfloor \log_b q \rfloor$

$$\bar{A} = \text{trunc}(A, \log_b(\delta))$$

$$\mathbb{C}_2 = \text{GSW}(\bar{A}, sk) = \left[\bar{A} \cdot G^* + \text{LWE}\left(0^{\log(q)(N+1)\times 1}\right)\right]_q$$

Homomorphic Multiplication

$$c_1\mathbb{C}_2 = G^{-1}(c_1)\mathbb{C}_2 \in \mathbb{Z}_q^{n\times(N+1)}$$

$$\text{Dec}(c_1\mathbb{C}_2) = \text{Dec}(G^{-1}(c_1)\mathbb{C}_2 * s)$$

$$= G^{-1}(m_1)\bar{A} \cdot G$$

SANITY CHECK

$$G^{-1}(c_1) \in \mathbb{Z}_q^{n\times\log(q)(N+1)}, \quad \mathbb{C}_2 \in \mathbb{Z}_q^{\log(q)(N+1)\times(N+1)}$$
$$\left(n \times \log(q)\,(N+1)\right) \cdot \left(\log(q)\,(N+1) \times (N+1)\right)$$

Results in a $n \times (N+1)$ matrix.

We can perform an external product between real numbers that have been shifted using the modified gadget tool.

Recall from Previous Slide

$$G^* = \mathbb{I}^{N+1} \otimes R^{*F}$$

$$R^* = [\delta, \delta b, \delta b^2, \dots, \delta b^{F-1}]$$
for $F = (N+1)\lfloor \log_b q \rfloor$

$$\bar{A} = \text{trunc}(A, \log_b(\delta))$$

$$\mathbb{C}_2 = \text{GSW}(\bar{A}, sk) = \left[\bar{A} \cdot G^* + \text{LWE}\left(0^{\log(q)(N+1)\times 1}\right)\right]_q$$

Homomorphic Multiplication

$$c_1\mathbb{C}_2 = G^{-1}(c_1)\mathbb{C}_2 \in \mathbb{Z}_q^{n\times(N+1)}$$

$$\text{Dec}(c_1\mathbb{C}_2) = \text{Dec}(G^{-1}(c_1)\mathbb{C}_2 * s)$$

$$= G^{-1}(m_1)\bar{A} \cdot G$$

Modified Decryption Function

$$\text{Dec}(c, sk) = \frac{1}{\delta}\left\lceil \frac{c \cdot s}{\Delta} \right\rfloor_q$$

By developing a method to design and realize LWE-integrated control systems, we can achieve end-to-end encryption to thwart cyber adversaries to meaningful ends.

Robert S. Lois
robert.lois@pitt.edu

Questions?