An Introduction to Homomorphic Encryption

for Computational Linguists

Symmetric Encryption (like AES)

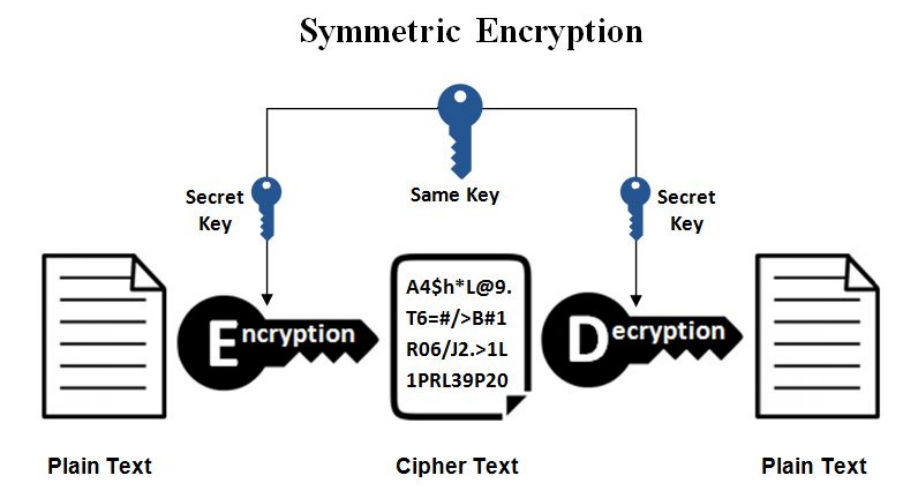


Image source: https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences

Asymmetric Encryption (like RSA)

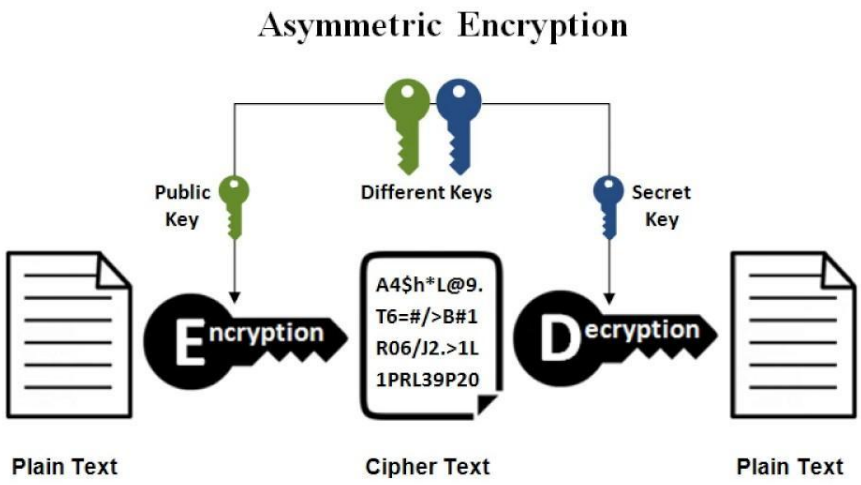


Image source: https://www.ssl2buy.com/wiki/symmetric-vs-asymmetric-encryption-what-are-differences

The Looming Risk

Cryptographic Schemes are based on hard problems

RSA is based on the hard problem of large prime factorization

Can be broken by Shor's algorithm using quantum computers

Ring Learning With Errors

Ring Learning with Errors (Brakerski et al., 2014):

$$R = \mathbb{Z}[x]/(x^d+1)$$

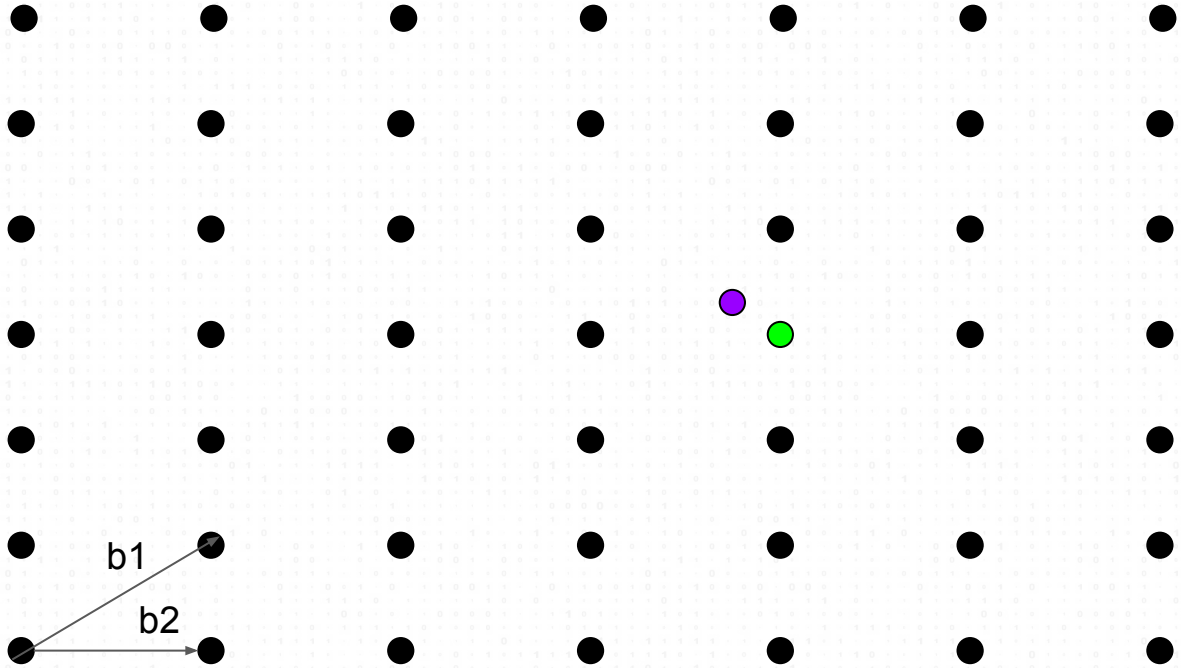
$d = d(\lambda)$ is a power of 2, $q = q(\lambda)$ is an integer ≥ 2, $\chi = \chi(\lambda)$ is a distribution over R

The $\mathrm{RLWE}_{d,q,\chi}$ problem is to distinguish between the following two distributions:

1. (a_i, b_i) sampled uniformly from R_q^2
2. Draw $s \leftarrow R_q$ uniformly and then sample $(a_i, b_i) \in R_q^2$ by sampling $a_i \leftarrow \mathbb{R}_q$ uniformly, $e \leftarrow \chi$, and setting $b_i = a_i \cdot s + e_i$.

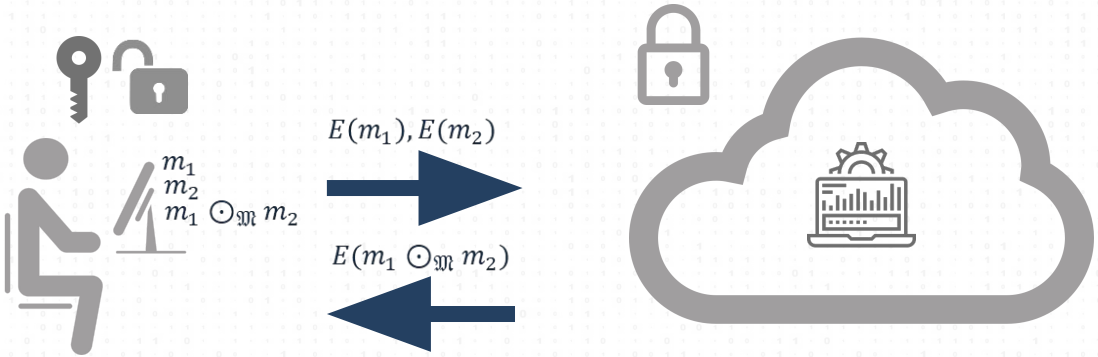
The $\mathrm{RLWE}_{d,q,\chi}$ assumption: this problem is computationally

Lattice-Based Cryptography



Homomorphic Encryption

$$\forall m_1, m_2 \in \mathfrak{M}, E(m_1 \odot_{\mathfrak{M}} m_2) \leftarrow E(m_1) \odot_{\mathfrak{M}} E(m_2)$$



Data owner

Service provider

Benefits

- Send expensive computations to the cloud
- Limit communication costs
- Not have to depend on network availability
- Have input and output data privacy
- Have mathematically guaranteed post-quantum security

Component-Wise Operations

Addition	Multiplication
$0x^4 + 4x^3 + 6x^2 + 2x + 5$ $+ 1x^4 + 6x^3 + 3x^2 + 5x + 2$ $1x^4 + 10x^3 + 9x^2 + 7x + 7$	$0x^4 + 4x^3 + 6x^2 + 2x + 5$ $* \quad 1x^4 + 6x^3 + 3x^2 + 5x + 2$ $1x^4 + 24x^3 + 18x^2 + 10x + 10$

Polynomial Operations

Addition	Multiplication
$0x^4 + 4x^3 + 6x^2 + 2x + 5$ $+ 1x^4 + 6x^3 + 3x^2 + 5x + 2$ $1x^4 + 10x^3 + 9x^2 + 7x + 7$	$0x^4 + 4x^3 + 6x^2 + 2x + 5$ $* \quad 1x^4 + 6x^3 + 3x^2 + 5x + 2$ $4x^7+30x^6+50x^5+55x^4+74x^3+37x^2+29x+10$

What You Can Do

- Addition: easy & cheap
- Multiplication: easy & expensive
- Plaintext & Ciphertext
- Ciphertext & Ciphertext

What You Can't Do

- Non-polynomial operations
- Most schemes: floating point operations

Dealing with Non-Polynomial Operations

- f(x) = x^2 used as an activation function instead of ReLU (Gilad-Bachrach et al., 2016)
- Polynomial approximation of sigmoid function using Chebyshev Polynomials or other such approximations (examples in multiple papers by Ehsan Hesamifard)
- Approximation using lookup table (Thaine, Gorbunov, and Penn, 2019)

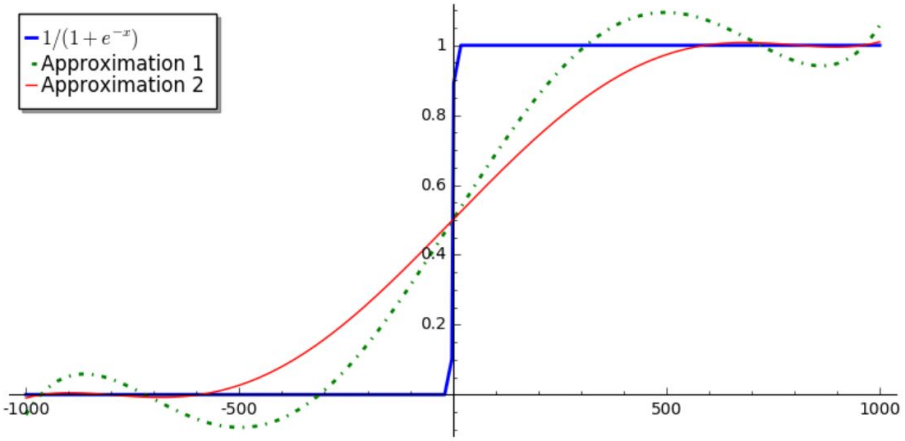
Example Approximation

(Hesamifard et. al 2016/2017)

Table 1: Polynomial approximation of sigmoid function on Interval $[-10^3, 10^3]$

$$p_1(x) = (2.069e-15) * x^5 + \cdots + 0.001 * x + 0.499$$

$$p_2(x) = (6.653e-16) * x^5 + \cdots + 0.001 * x + 0.500$$



Floating Point Operations

Option 1
(BGV & BFV schemes):

Decide precision n, multiple by 10^n and round or truncate

Option 2
(CKKS scheme):

Floating point operations possible

Some NLP-Specific Applications of Homomorphic Encryption

Secure spoken language processing proposed:

- Secure spoken keyword recognition (using HMMs) [Pathak et al., 2011]
- Secure speaker identification (using GMMs) [Pathak et al., 2013]
- Secure speaker authentication (using GMMs) [Pathak et al., 2013]
- Secure Fourier transform and speech feature extraction [Thaine and Penn, 2019]

Secure text processing proposed:

- Secure TF-IDF (Patent)
- Secure Language Recognition (Patent)
- Secure Keyword Search (Patent)
- Privacy-Preserving Character Language Modelling [Thaine and Penn, 2019]

Homomorphic Encryption Standardization

An Open Industry / Government / Academic Consortium to Advance Secure Computation

Homomorphic Encryption

Homomorphic Encryption provides the ability to compute on data while the data is encrypted. This ground-breaking technology has enabled industry and government to provide never-before enabled capabilities for outsourced computation securely.

HomomorphicEncryption.org is an open consortium of industry, government and academia to standardize homomorphic encryption.

Please join our mailing list and participate in our standardization efforts.

Community Growth:

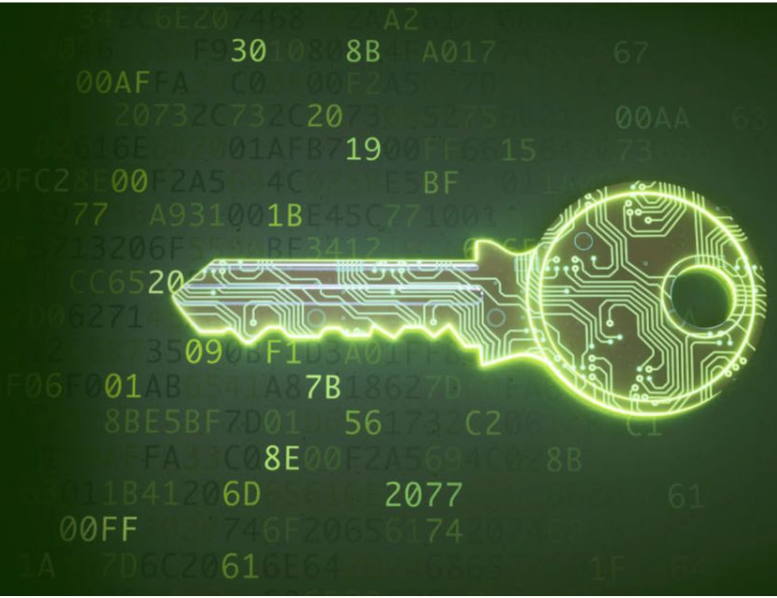
OpenFHE is an open-source project that provides
efficient extensible implementations of the leading
post-quantum Fully Homomorphic Encryption (FHE) schemes.

We always welcome new members and contributors to the OpenFHE community.

To get started, we maintain a project announcements mailing list here:
https://groups.google.com/a/openfhe.org/g/announcements

Microsoft SEAL

Build end-to-end encrypted data storage and computation services



Microsoft SEAL—powered by open-source homomorphic encryption technology—provides a set of encryption libraries that allow computations to be performed directly on encrypted data. This enables software engineers to build end-to-end encrypted data storage and computation services where the customer never needs to share their key with the service.

Microsoft SEAL is open source (MIT license). Start using it today!

Task:

Which Computational Linguistics tasks would you use Homomorphic Encryption for & how?

Thank You!

patricia@private-ai.com

@PrivateNLP

www.private-ai.com

PRIVATEAI