Securing MLaaS with Homomorphic Encryption

Finding an intersection between Machine Learning and Cryptography

Lorenzo Rovida

l.rovida1@campus.unimib.it

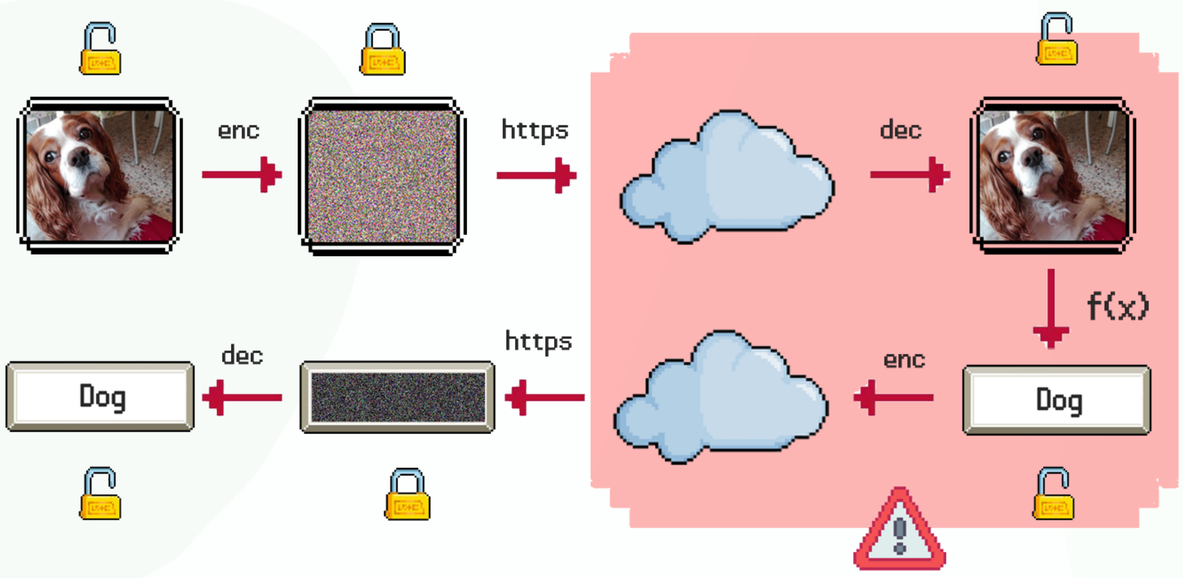
Supervisor: Prof. Alberto Leporati

Department of Informatics, Systems and Communication (DISCo)

Ph.D. Program in Computer Science

The problem

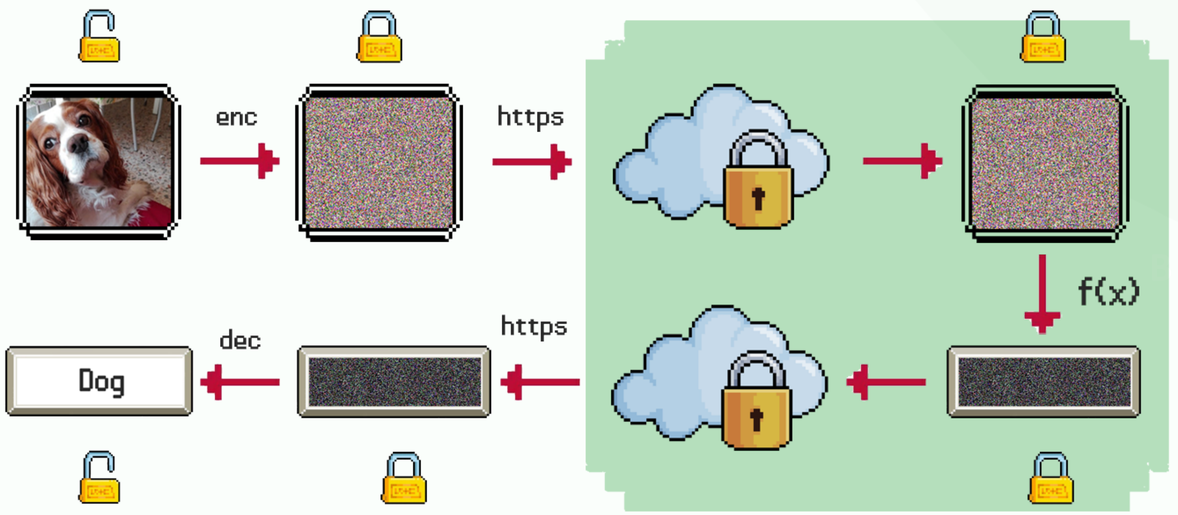
Today, when using cloud services, data is encrypted in-transit, but it needs to be decrypted by the service provider in order to be used. This issue raises a significant concern regarding data privacy.



The service provider can access the data in its unencrypted form! Unluckily, this is a structural characteristic of classical cryptographic schemes, therefore it is difficult to address this issue. This is where Homomorphic Encryption (HE) comes into play!

Homomorphic Encryption (HE)

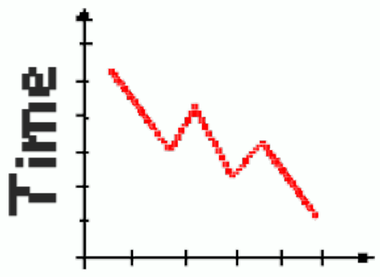
A HE scheme allows for computations to be performed on encrypted data, meaning that a service could be provided while maintaining data in an encrypted form.



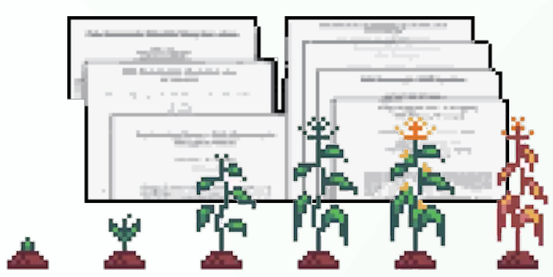
In particular, we are able to perform primitive operations (`add` and `mul`) on encrypted data. It has been shown [1] that Turing Machines can be evaluated on encrypted data, thus obtaining a Turing-complete model of computation.

From Past to Present

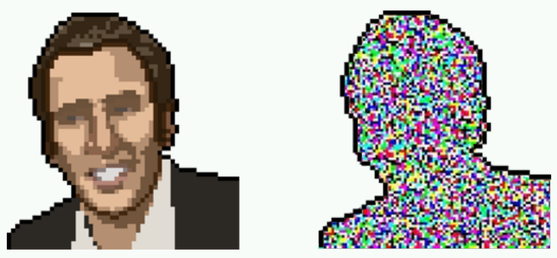
In 2009 Craig Gentry proposed the first FHE scheme [2] built on a lattice-based and quantum-resistant problem: Learning with Errors (LWE). The scheme was able to perform additions and multiplications on encrypted bits.



A single operation on a bit used to take more than 10^8 seconds to complete. However, ten years later, research has progressed and better hardware has improved this time to just 10^{-4} seconds, enabling interesting applications of HE.

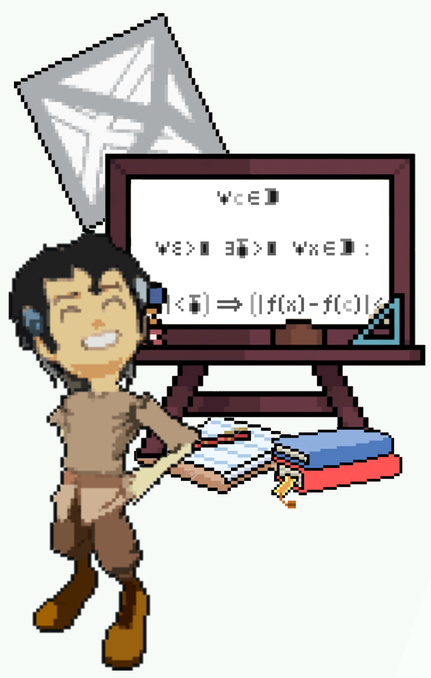


These results renewed a research field stuck since 1978, when Rivest et. al. [4] introduced the concept of HE. This enabled the birth of four generations of HE schemes that work on binary, integer and real values.



Today, we are able to use HE to train Neural Networks that operate on encrypted data, allowing for tasks such as CNNs inference [3] and clustering algorithms [5] to be performed on data that the server is not able to see in clear.

Discussion



Nevertheless, not everything that glitters is gold. HE comes with significant overhead, thus performing complex computations can be resource-intensive. Plus, there are some non-trivial challenges, such as:

- Comparing values: since values are encrypted, how do we evaluate something like $a > b$?
- Non-linear functions: in a context where only additions and multiplications are supported, how do we evaluate a function like e^x ?

References

[1] Goldwasser, S., Kalai, Y. T., Popa, R. A., Vaikuntanathan, V. & Zeldovich, N. (2013). How to Run Turing Machines on Encrypted Data.. In R. Canetti & J. A. Garay (eds.), *CRYPTO (2)* (p./pp. 536-553), : Springer. ISBN: 978-3-642-40083-4

[2] Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. In M. Mitzenmacher (ed.), *STOC* (p./pp. 169-178), : ACM. ISBN: 978-1-60558-506-2

[3] Kim, D., & Guyot, C. (2023). Optimized Privacy-Preserving CNN Inference With Fully Homomorphic Encryption. IEEE Transactions on Information Forensics and Security, 18, 2175-2187.

[4] Rivest, R. L., Adleman, L. & Dertouzos, M. L. (1978). On Data Banks and Privacy Homomorphisms. *Foundations of Secure Computation, Academia Press*, , 169-179.

[5] Rovida, L. (2023). Fast but approximate homomorphic k-means based on masking technique. International Journal of Information Security. 1-15. doi: 10.1007/s10207-023-00708-9.