Saving the (post-quantum) world with neural networks

Mihail-Iulian Pleșa

Who am I?

- Passionate about research and dissemination
- Interests: security aspects and applications of A.I.
- Security Researcher at Orange Services

What You Will Know (about post-quantum crypto)

1. What it is?

2. Why do we need it?

3. What do neural networks have to do with it?

Contents

1. RSA and DH
2. Shor’s algorithm
3. NIST post-quantum initiative
4. Key encapsulation and signatures
5. Tree Parity Machines
6. Demo
7. Conclusions

DH algorithm

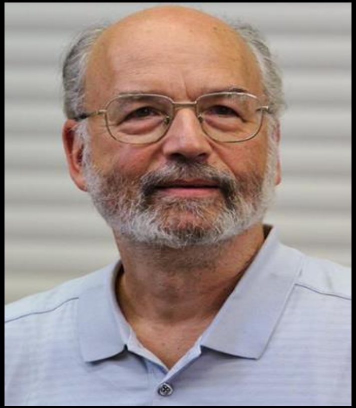
D

H

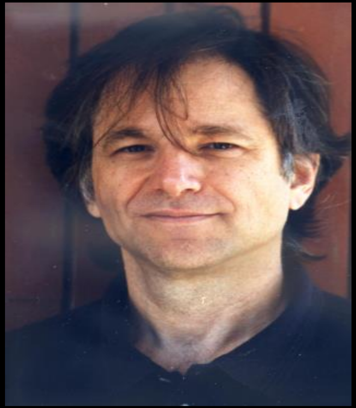
RSA algorithm

R

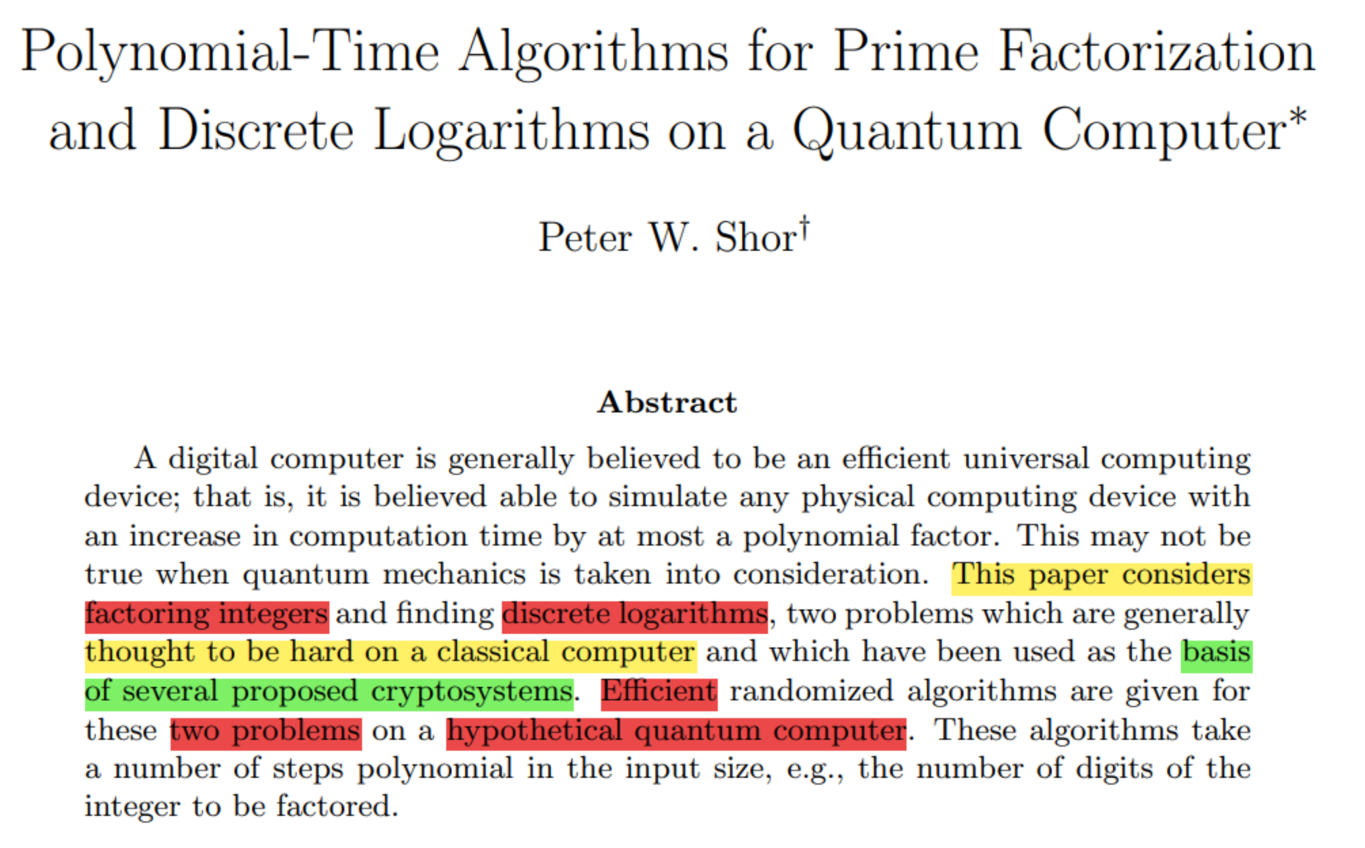
S



A



Shor’s algorithm



Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer*

Peter W. Shor†

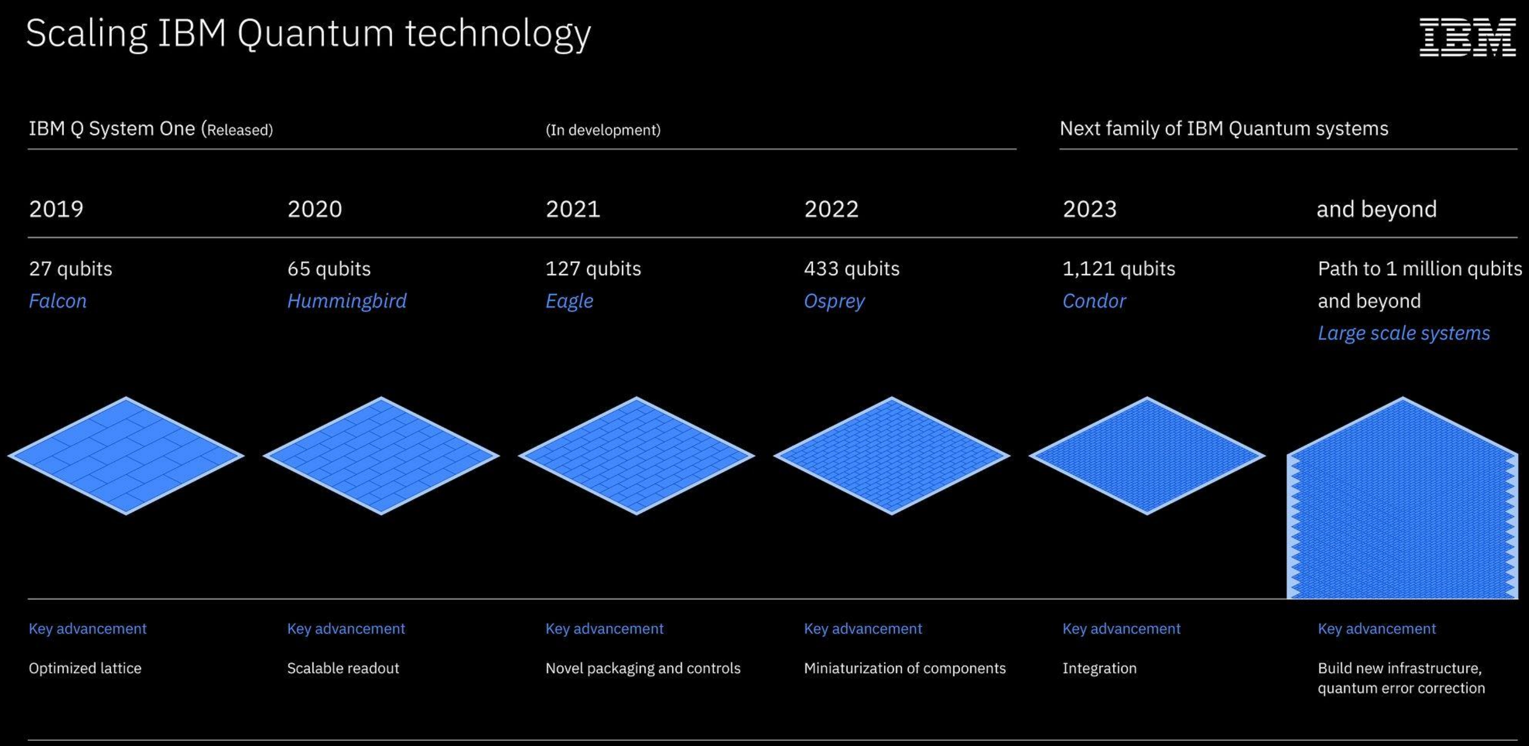
Abstract

A digital computer is generally believed to be an efficient universal computing device; that is, it is believed able to simulate any physical computing device with an increase in computation time by at most a polynomial factor. This may not be true when quantum mechanics is taken into consideration. This paper considers factoring integers and finding discrete logarithms, two problems which are generally thought to be hard on a classical computer and which have been used as the basis of several proposed cryptosystems. Efficient randomized algorithms are given for these two problems on a hypothetical quantum computer. These algorithms take a number of steps polynomial in the input size, e.g., the number of digits of the integer to be factored.

RSA-2048?

We just need 20M qubits

IBM Q



Scaling IBM Quantum technology

IBM Q System One (Released) (In development) Next family of IBM Quantum systems

2019	2020	2021	2022	2023	and beyond
27 qubits *Falcon*	65 qubits *Hummingbird*	127 qubits *Eagle*	433 qubits *Osprey*	1,121 qubits *Condor*	Path to 1 million qubits and beyond *Large scale systems*
Key advancement	Key advancement	Key advancement	Key advancement	Key advancement	Key advancement
Optimized lattice	Scalable readout	Novel packaging and controls	Miniaturization of components	Integration	Build new infrastructure, quantum error correction

What to do?

Quantum key agreement

- Based on quantum effects
- Specific hardware
- High costs
- Guarantee to work

Post-quantum cryptography

- Based on hard math problems
- Usual hardware
- Low costs
- Many believe it will work

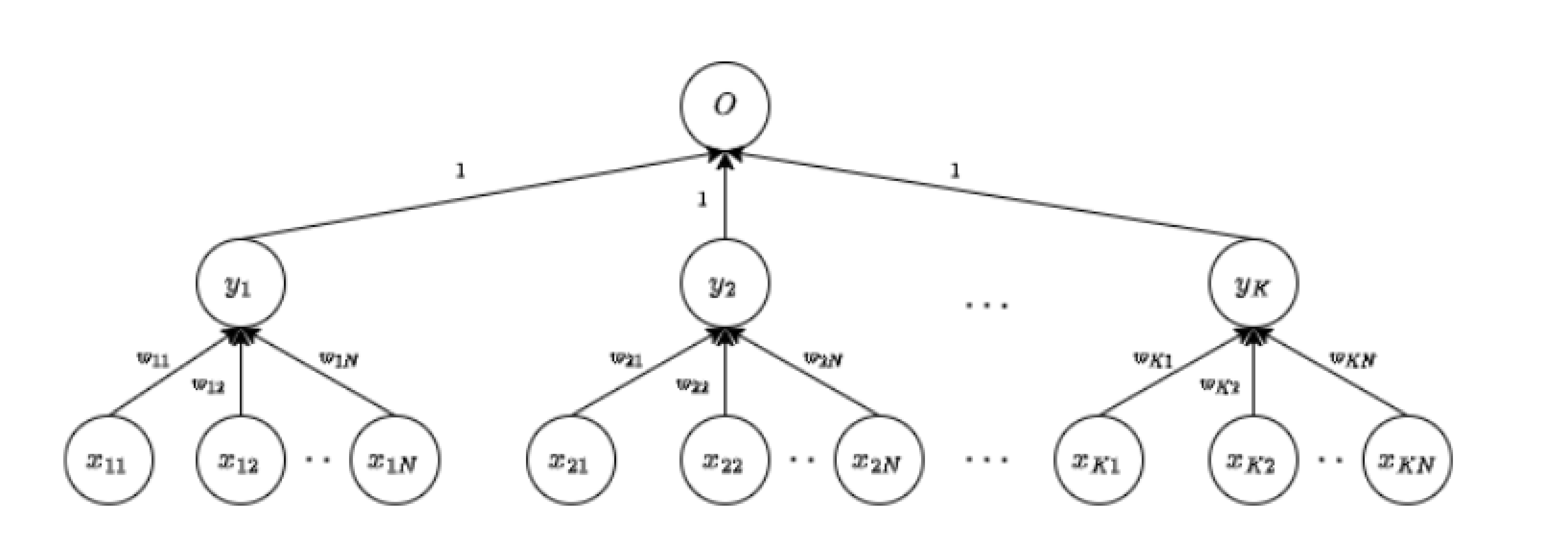
Finaists

2016	Call for Proposals
2017	Round 1
2019	Round 2
2020	Round 3
2022	Round 4

NIST

- Encryption:
  - CRYSTALS-Kyber (lattice)
- Signatures:
  - CRYSTALS-Dilithium (lattice)
  - FALCON (lattice)
  - SPHINCS+ (hash)

TPM


O
1
1
1
y_1
y_2
$\cdots$
y_K
w_{11}
w_{12}
w_{1N}
w_{21}
w_{22}
w_{2N}
w_{K1}
w_{K2}
w_{KN}
x_{11}
x_{12}
$\cdot\cdot$
x_{1N}
x_{21}
x_{22}
$\cdot\cdot$
x_{2N}
$\cdots$
x_{K1}
x_{K2}
$\cdot\cdot$
x_{KN}

TPM

- K groups of N input neurons
- Each group is connected to a single hidden neuron
- All hidden neurons are connected to the output neuron
- The weights are integers from 0 to L

Notations

- X^i - the values of the input neurons from the i^{th} group
- y^i - the value of i^{th} hidden neuron
- O - the value of the output neuron
- $\sigma(x)$ - the sign of x

TPM

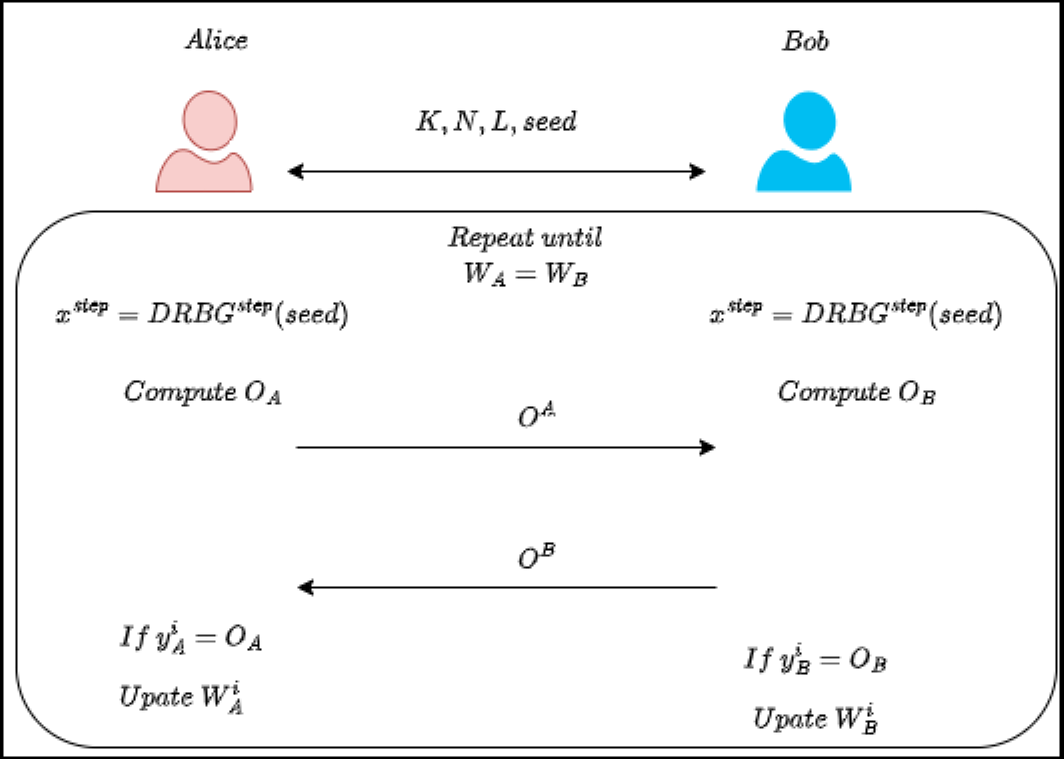
Compute the hidden neurons

- Multiply the inputs with the weights for each group
- $y^i = w_i \cdot x_i$

Compute the output

- Get the sign of each hidden neuron and multiply them
- $O = \prod_{i=1}^{K} \sigma(y_i)$

Protocol


$Alice$
Bob
$K, N, L, seed$
$Repeat\ until$
$W_A = W_B$
$x^{step} = DRBG^{step}(seed)$
$x^{step} = DRBG^{step}(seed)$
$Compute\ O_A$
$Compute\ O_B$
O^A
O^B
$If\ y_A^i = O_A$
$Upate\ W_A^i$
$If\ y_B^i = O_B$
$Upate\ W_B^i$

Intuition

Let’s take a (random) walk

Differences with current SOTA

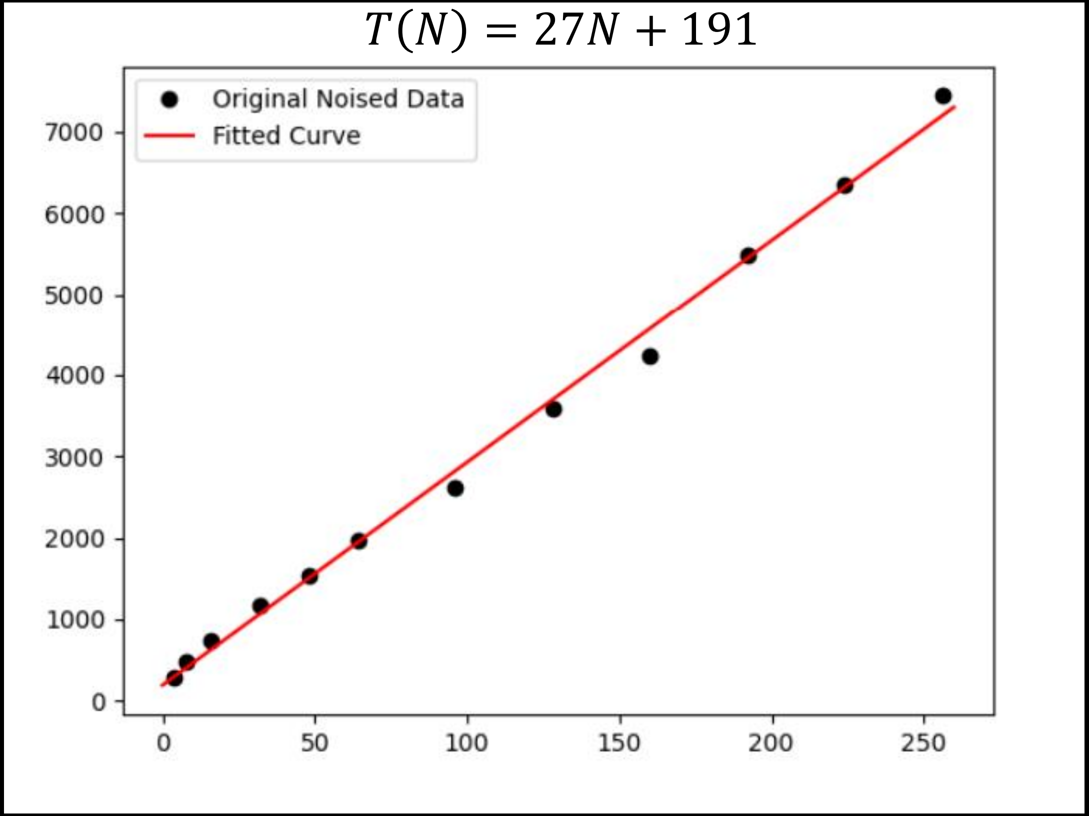
Previous approaches

- Non-cryptographic security definition i.e. secure means the attacker cannot recover 90% of the key
- Inputs are binary
- Weights can be negative
- Ignore the 0 sign
- Include the extrema values (L) in the key

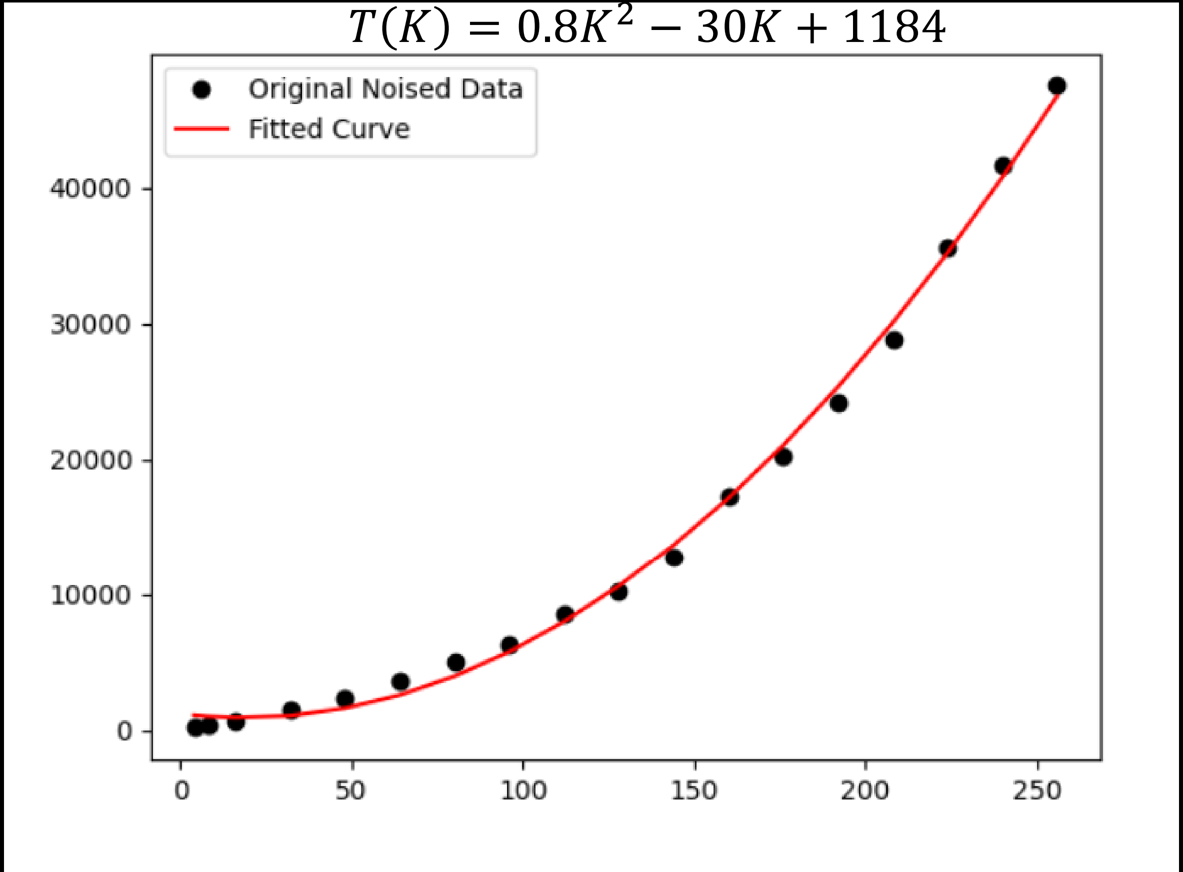
Our approch

- More cryptographic security definition i.e. secure means the attacker cannot recover a non-negligible percentage of the key
- Inputs are integers
- Weights are positive
- Take the 0 sign into consideration
- Exclude the extrema values (L) from the key

Running time w.r.t. N


$$T(N) = 27N + 191$$

Running time w.r.t. K



$$T(K) = 0.8K^2 - 30K + 1184$$

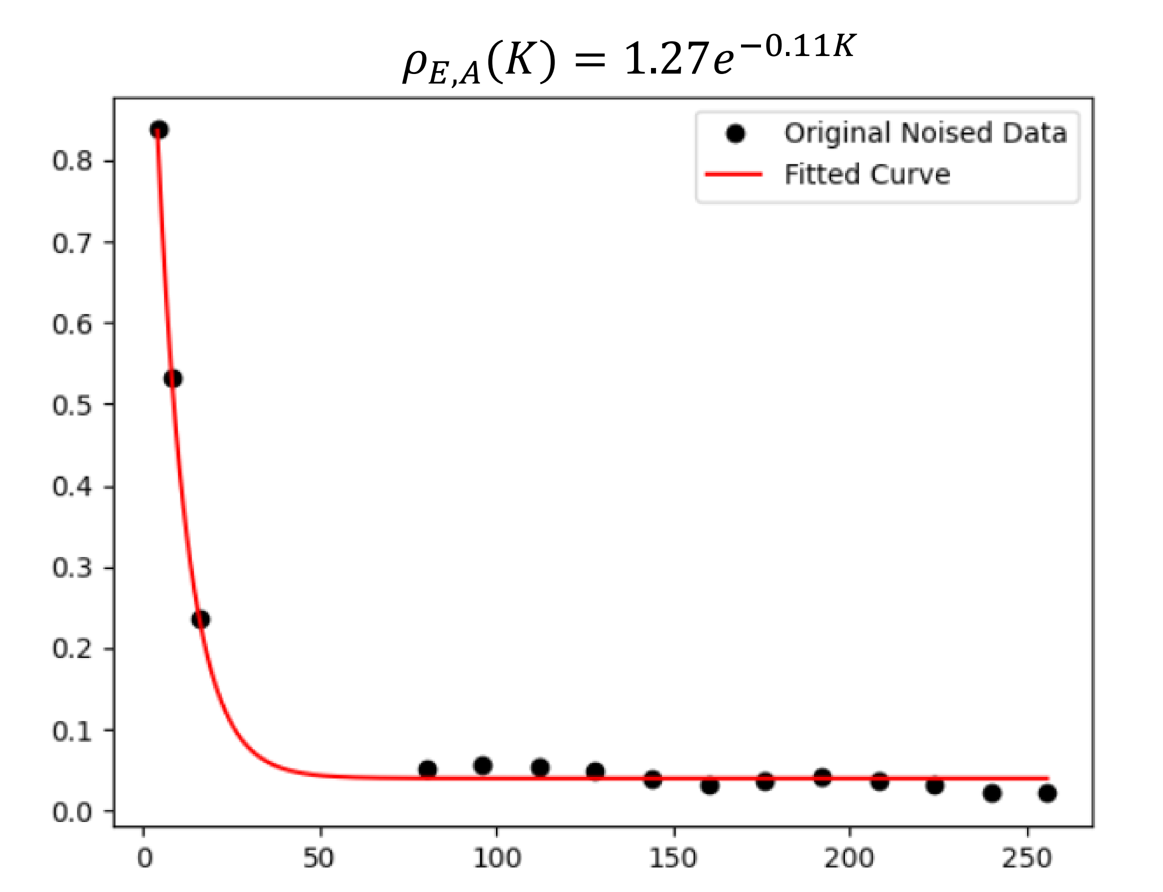
Attacks

- Naive attack
- Geometric attack
- Majority attack
- Genetic attack

Naive attack

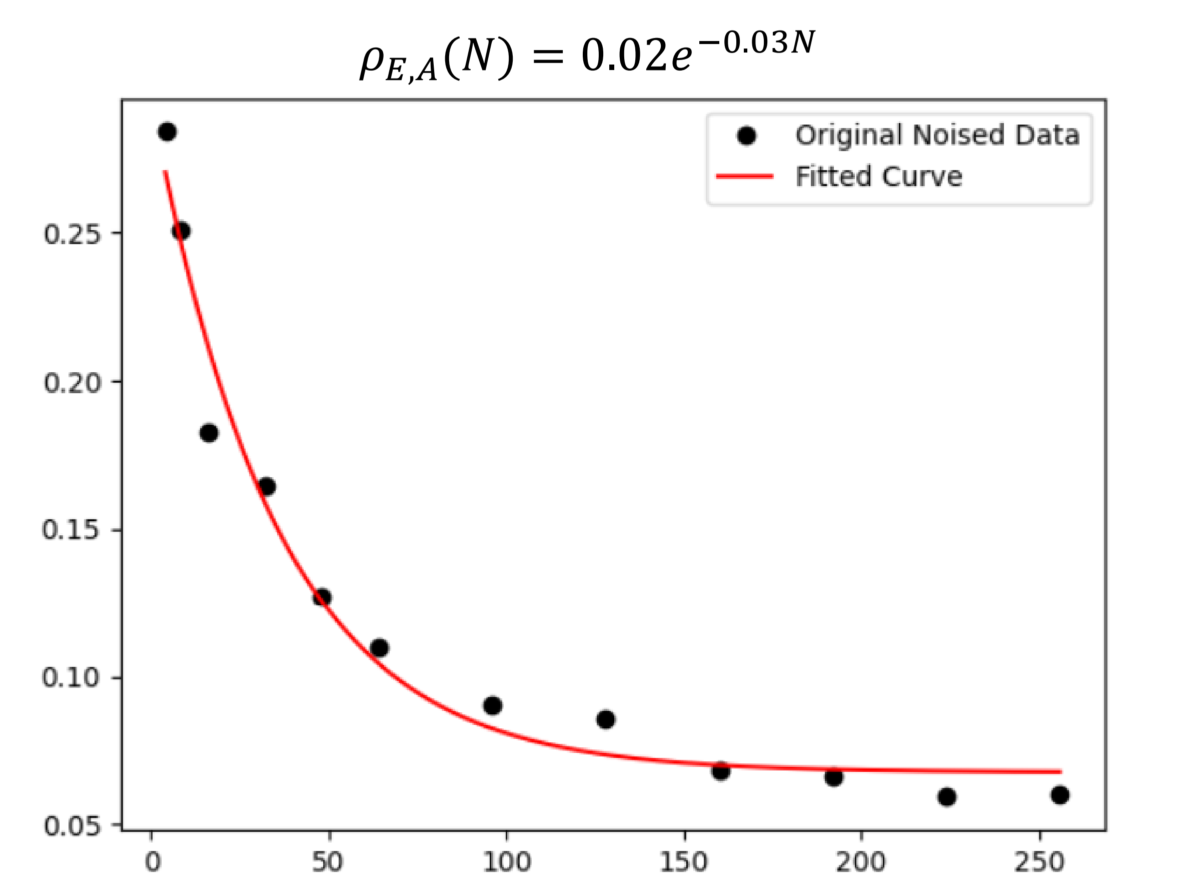
- Attacker intercepts the inputs and the outputs
- It updates its weights if it is synchronized with both Alice and Bob

Nave attack w.r.t. K



$$\rho_{E,A}(K) = 1.27e^{-0.11K}$$

Nave attack w.r.t. N



$$\rho_{E,A}(N) = 0.02e^{-0.03N}$$

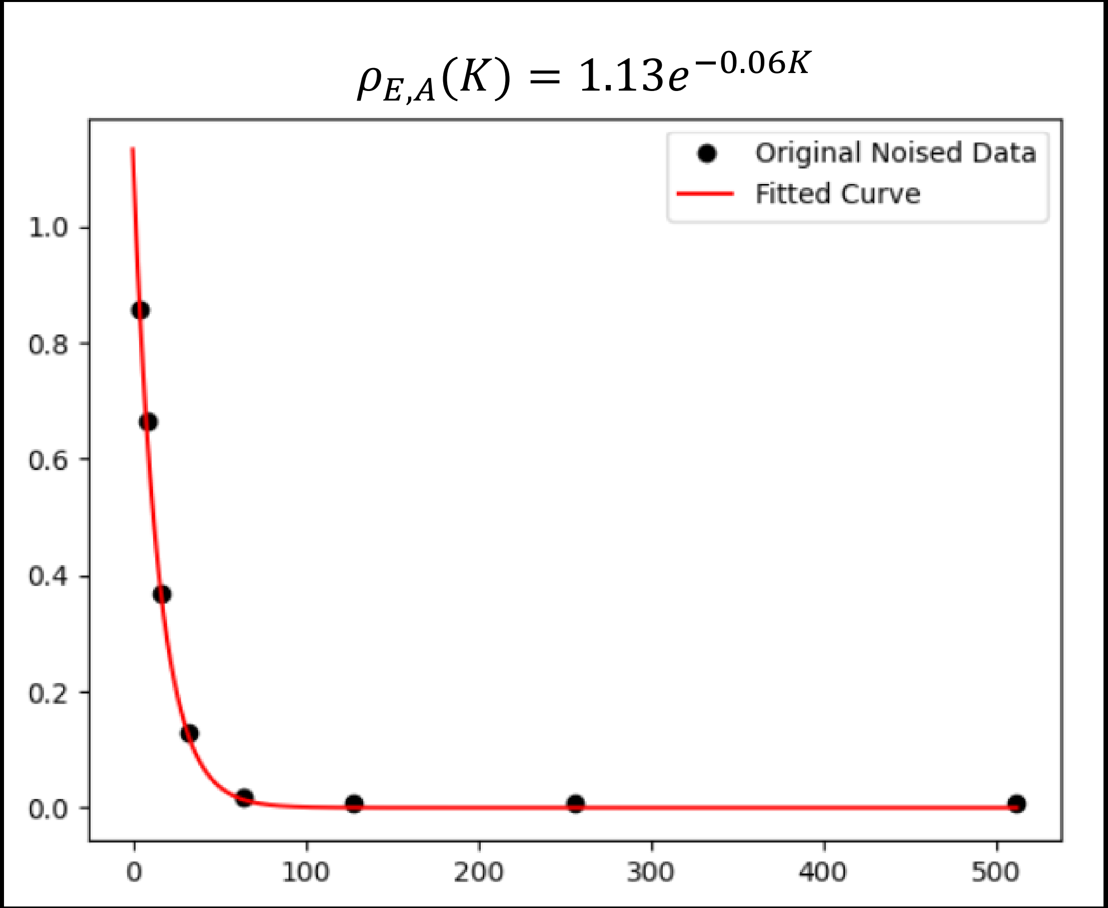
Geometric attack

- Attacker intercepts the inputs and the outputs
- If the attacker is not synchronized with both Alice and Bob, it corrects the corresponding hidden neurons and then updates the weights
- Idea: reverse the sign of the hidden neuron whose input is closest to the weights

Majority attack

- The attacker instantiates M parallel geometric attacks
- It updates all the hidden neurons with the most frequent configuration

Nave attack w.r.t. K



$$\rho_{E,A}(K) = 1.13e^{-0.06K}$$

Genetic attack

- Similar to majority attack
- Use a genetic algorithm to update the weights
- Exponential in K

Demo

Summary

- What about authentication?
- It may be a solution
- Not based on a “hard” math problems
- Simple to implement
- Need to be formalized

Thank you

orange™

References

- Shor, P.W., 1999. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM review, 41(2), pp.303-332.Li, N., Lyu, M., Su, D. and Yang, W., 2017.
- Gidney, C. and Ekerå, M., 2021. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. Quantum, 5, p.433.
- https://www.ibm.com/quantum/blog/ibm-quantum-roadmap
- https://csrc.nist.gov/Projects/Post-Quantum-Cryptography/Post-Quantum-Cryptography-Standardization
- https://www.nist.gov/news-events/news/2022/07/nist-announces-first-four-quantum-resistant-cryptographic-algorithms
- Rosen-Zvi, M., Klein, E., Kanter, I. and Kinzel, W., 2002. Mutual learning in a tree parity machine and its application to cryptography. Physical Review E, 66(6), p.066135.
- Klimov, A., Mityagin, A. and Shamir, A., 2002. Analysis of neural cryptography. In Advances in Cryptology—ASIACRYPT 2002: 8th International Conference on the Theory and Application of Cryptology and Information Security Queenstown, New Zealand, December 1–5, 2002 Proceedings 8 (pp. 288-298). Springer Berlin Heidelberg.
- Ruttor, A., Kinzel, W., Naeh, R. and Kanter, I., 2006. Genetic attack on neural cryptography. Physical Review E, 73(3), p.036121.