Learning with Errors report

1 Introduction

This report details the strategies and the reasoning behind the algorithm choices used to attack a Learning with Errors (LWE) cryptosystem. The main objective is to explain how these attacks work clearly and thoroughly, backed up by solid theoretical and practical justifications. Each attack was customised to the cryptosystem, with precise modifications made to error vectors by Alice and Bob. Each attack implementation is tailored to the cryptosystem with nuances based on adjustments made to error vectors by Alice and Bob

The Cryptosystem:

Key Generation

1. Choose $\mathbf{A}$ uniformly at random from $\mathbb{Z}_q^{m \times n}$.
2. Choose $\mathbf{s}$ uniformly at random from $\mathbb{Z}_q^n$.
3. Sample $\mathbf{e}$ from χ^m.
4. Calculate $\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$
5. Return $(\mathbf{A}, \mathbf{b})$ as the public key.
6. Return $\mathbf{s}$ as the private key.

Encryption of a plaintext bit pt

1. Choose $\mathbf{r}$ uniformly at random from $\mathbb{Z}_2^m$.
2. Calculate $\mathbf{a}'^\top = \mathbf{r}^\top \cdot \mathbf{A}$
3. Calculate $b' = \mathbf{r}^\top \cdot \mathbf{b} + \text{pt} \cdot \frac{q}{2}$
4. Return $(\mathbf{a}', b')$ as the ciphertext.

Decryption

1. $v = \mathbf{a}'^\top \ \mathbf{s}$
2. $m' = b' - v$
3. If m' is closer to 0 than $\frac{q}{2} \pmod q$ then pt = 0; else pt = 1.

Where χ is based on either the Gaussian or Binomial distribution with a mean of 0 and a small standard deviation.

2 Attack Implementations

2.1 crack1 ‘Learning without Errors’.

Alice and Bob have changed their cryptosystem so that the error distribution is always zero. This modification simplifies the equation b = A · s + e to b = A · s. Since the error vector(e) is zero, it doesn't need to be considered in this new scheme. The public key, ciphertext, and modulus q are known values in this adapted method. The aim is to use this information to attack the cryptosystem and decrypt the plaintext accurately.

To carry out an effective attack, it is essential to determine the secret vector (s) that can decrypt the ciphertext. With zero error, the equation can be simplified to b = A · s, where A and b are known from the public key. However, the challenge is solving for s since directly inverting A to compute $A^{-1} \cdot b = s$ is not feasible because A is a non-square matrix.

To overcome this problem, NumPy is used in Python to create an augmented matrix [A|b] and applied Gaussian elimination [4] to transform it into an upper triangular format. This process simplifies the matrix while preserving its solution space. Once the matrix was in an upper triangular format, implemented a backward substitution to solve for the s. This step systematically unravels the s starting from the last row, exploiting the triangular structure to obtain each entry of s. The decryption process is then straightforward with the s obtained. The plaintext can be revealed by solving the original LWE-based decryption equation using the uncovered s and the given ciphertext, also making sure all calculations is between 0 and q-1.

The time complexity of the attack is $O(mn^2)$. For each column (up to n), operations on all m rows may be required. Each operation involves O(n) computations due to row manipulations.

When the error term is absent from the LWE problem, the problem becomes a standard system of linear equations, represented as b = A·s. This system is deterministic, which implies that the solution s is fixed for a given A and b. If the system is consistent and A has full rank, the solution can be found with absolute certainty.

Theorem:

Let A be a m x n matrix with m ³ n over a field F and vector b Î F^m. The linear system b = A·s has a has a unique solution s in F^m if and only if matrix A has a full Rank.

Proof:

The Rouché–Capelli theorem [1] states that for a system of linear equations, if the rank of the coefficient matrix is equal to the rank of the augmented matrix, then the system has a solution. Additionally, if the coefficient matrix has full rank, then the solution to the system is unique. In other words, there are no free variables in the system.

Consider Matrix A size of m x n with m ³ n and vector b Î F^m here A is stipulated to have full rank, implying that rank(A) = n. The augmented matrix [A|b] is constructed by appending b to A. By the definition of matrix rank and the properties of linear independence, the rank of the augmented matrix [A|b] cannot exceed the number of its columns, and thus rank([A|b]) = rank(A) = n, if b lies within the column space of A. The consistency of the linear system b = A·s dictates that b must indeed be a linear combination of the columns of A, confirming that rank([A|b]) = n.

Therefore, applying Gaussian elimination to the equation b = A·s always yields the accurate secret vector, thanks to these linear algebraic principles.

Gaussian elimination is a reliable method used for solving linear systems. It is a direct method that guarantees an exact solution within a finite number of steps. This is particularly important in cryptographic applications, where accuracy in the decryption process is crucial for the correct recovery of plaintext. Compared to iterative methods, which are susceptible to accumulating computational errors through successive iterations, Gaussian elimination ensures the integrity of the solution. Moreover, it has better computational efficiency than LU decomposition, which requires additional matrices (L and U) to execute, thereby increasing the calculation complexity. Therefore, Gaussian elimination is the better choice in cryptographic scenarios as it ensures both exactness and computational efficiency.

2.2 crack2 'Learning with Few Errors'

Alice and Bob have changed their cryptosystem so that the error distribution is 1, −1 with very low probability or otherwise is 0. Again, the public key, ciphertext, and modulus q are known values in this adapted method. The aim is to use this information to attack the cryptosystem and decrypt the plaintext accurately.

The attack strategy's design aims to simplify the equation b = A · s + e by using the principles applied in the crack1 implementation. The goal is to make it b = A·s by identifying the correct error vector e such that b – e = A·s. This approach is based on linear algebraic principles established in crack1, which suggest that a unique solution s can be found for the modified equation, provided the appropriate e is discovered. To compute s reliably, the method of Gaussian elimination is used. Identifying the correct e involves an exhaustive combinatorial search over potential low-distribution error vectors. For example: 1 error 2 error……etc

100 -100 1-10

010 0-10 10-1

001 00-1 -110…… etc

Once every possible error vectors are generated and tested systematically in the equation b – e = A · s. This generates a corresponding s for every e. To identify the correct s, the recalculated b' is compared with the original b. As s is unique in the equation b = A·s, only one e will yield a matching result. Once the correct s is found, the decryption process becomes simple. The plaintext is recovered by applying the original LWE-based decryption formula using the s found and the given ciphertext. It is ensured that all operations are conducted within the range of 0 to q−1.

The design choices of the crack2 function are carefully crafted to take advantage of the unique properties of the modified LWE cryptosystem, like low-distribution error vectors. Its accuracy is based on the same linear principles as crack1 and the exhaustive listing of possible error vectors.

This cryptosystem has an error vector with a small magnitude. This method directly targets these lowmagnitude errors by exhaustively generating and testing them, making it more efficient than other methods that do not leverage this characteristic. However, it is essential to note that while this method has advantages, it may not be universally superior. Its effectiveness and efficiency are highly dependent on the specific parameters of the cryptosystem. For instance, when m is large, it becomes computationally expensive due to the time complexity being $O(Emn^2)$ where E is the number of generated vectors.

2.3 crack3 'Learning with Errors'

Alice and Bob switched to a more appropriate value for χ and used the standard specification of LWE. Once again, the public key, ciphertext, and modulus q are known values. The attack aims to use this information to decrypt the plaintext accurately. Initial approach was to use lattice reduction to get an estimation for e, however a switch to a brute force method was later decided which will be outlines further in this section.

The initial strategy involved using lattice reduction to estimate the error vector. This technique was necessary because brute force methods struggle with complex error distributions in the LWE problem. The main aim of this strategy was to format the matrix A and vector b into a lattice structure [2]. Using the "scipy" library, QR decomposition [3] was applied to approximate e and simplify the equation from $b - e = A \cdot s$ to $b = A \cdot s$. This step set the stage for Gaussian elimination. The goal was to find the secret vector, enabling the standard LWE decryption methodology. However, this approach encountered difficulties finding suitable approximations for e., leading to inaccuracies in the decryption results.

Upon further analysis, a crucial vulnerability was discovered within the encryption algorithm. The error distribution was limited to 0, 1, and -1 values. This discovery made it possible for a brute-force attack, which is particularly viable for smaller dimensions of n. The attack involves generating all potential combinations of the secret key (s) within the range [0, q-1]. By iterating through each combination and computing a value b using the equation $b' = A \cdot s'$, it became possible to match the computed b' with the actual b within a specified tolerance. Finding a matching value of b' would indicate the correct value of s' for decryption. However, this brute-force method is computationally expensive, especially for larger values of n or q, as its time complexity grows exponentially with the size of n.

References

[1] https://www.sangakoo.com/en/unit/rouche-capelli-theorem

[2] Nguyen, Phong Q. (2009). "Hermite's Constant and Lattice Algorithms". The LLL Algorithm. Information Security and Cryptography. Berlin, Heidelberg: Springer Berlin Heidelberg. pp. 19–69. doi:10.1007/978-3-642-02295-1_2. ISBN 978-3-642-02294-4. ISSN 1619-7100.

[3] https://en.wikipedia.org/wiki/QR_decomposition

[4] https://www.oreilly.com/library/view/computer-securityand/9780471947837/sec3.9.html#:~:text=Gaussian%20elimination%20is%20a%20process,x%20in%20Equation%20(3.20).