EPQ Cryptosystem

Ella Rose

Independent Researcher, Enfield, NH

python_pride@protonmail.com

Abstract. This paper defines the EPQ cryptosystem, which provides a trapdoor style cryptographic primitive built around lattices and noisy linear euqations. Two new techniques are introduced, short modular inverses and noisy moduli. A short modular inverse is a value a^{-1} such that $a^{-1} = inverse(a, q)$ and $log(a^{-1}) < log(a)$ by a significant margin. A noisy modulus is a value $q + r$ for a modulus q and some relatively small r, which facilitates approximate modular reduction.

Keywords: Public-Key Cryptography

1 Introduction

Public-Key cryptography enables secured communications over an non-secure communications channel, which is a vital capability in the age of the internet. Classical cryptosystems such as Diffie-Hellman and RSA have been in use for decades, but are vulnerable to attack by quantum computers.

Lattice based cryptosystems offer a promising alternative to classical cryptosystems such as Diffie-Hellman and RSA. In addition to being resistant to attack by quantum computers, lattice based constructions tend to be significantly more efficient than their classical counterparts.

Lattice-based cryptosystems are built around noisy linear equations. Many are conceptually complex and non-trivial to implement.

This paper defines a performant, conceptually simple, easy to implement public-key cryptosystem. The cryptosystem operates in a minimal dimension with large errors, as opposed to the high dimension matrix with small errors that many other cryptosystems utilize.

The trapdoor works by multiplying a ciphertext $as + e \bmod q$ with a secret short modular inverse a^{-1}, resulting in a value $s + a^{-1}e$ that is smaller than the modulus q. Solving for some bits of s is then trivial.

In order to prevent an adversary from recovering the short inverse a^{-1} from the pair a and q, the value a is generated using a modulus $q + k$ that is approximately similar to q. Working with an approximately similar modulus introduces noise into the result; The parameter sizes are selected such that the resulting noise is no larger then the size of the error term e.

The cryptosystem utilizes BigNums instead of polynomials or matrices of machine sized words. Reliable and reputable BigNum libraries have been widely

available for quite some time. The ability to utilize older, heavily vetted libraries is an advantage when deploying the algorithm the real world.

The key generation phase is the least efficient and most complicated part, requiring an algorithm for modular inversion. This is not to say that the key generation phase is inefficient or complex: The extended euclidean algorithm can simply and efficiently generate modular inverses even for very large numbers.

The public key operation and private key operation are very fast: Each requires only a single modular multiplication. Optimized variants do not require an addition operation. The error term e can be the result of zeroing the least significant bits of $as \bmod q$, which may be implemented via simple binary right shifting. This has the added effect of compressing the ciphertexts.

Public keys, private keys, and ciphertexts are all very small. For a 256-bit security level, public keys are 1536 bits, a private key is 768 bits, and a compressed ciphertext is about 550 bits.

Additionally, the cryptosystem is provably secure as a 1d LWE instance with a single sample - An adversary who can break the crypotsystem can solve an instance of 1d LWE with only a single sample.

1.1 Organization

Section 2 defines the notation used in this paper. Section 3 explains short modular inverses and noisy moduli. Section 4 details the EPQ cryptosystem. Section 5 discusses other algorithms that can be instantiated using short inverses and/or noisy moduli. Section 6 concludes the paper.

2 Notation

$a \leftarrow b$ indicates the assignment of the value of b to the variable a
ab denotes integer multiplication of a times b
$a + b$ denotes integer addition of a plus b
$a \oplus b$ denotes the exclusive-or operation on a, b
$x \bmod q$ indicates the modular reduction of x modulo q
$inverse(a, q)$ indicates the generation of the multiplicative inverse of $a \bmod q$
$random(n)$ indicates the generation of a random n-bit integer
$x << n$ indicates the binary left shift of x by n places
$x >> n$ indicates the binary right shift of x by n places
2^n indicates the raising of 2 to the power of n

3 Short Inverses and Noisy Moduli

A modular multiplicative inverse is a number a^{-1} such that $aa^{-1} \bmod q = 1$. A short inverse is an inverse a^{-1} such that $log(a^{-1}) < log(a)$ by a significant margin. Combined with appropriate parameter sizes, this size discrepancy allows one to shrink a value $as+e \bmod q$ to a value $s+a^{-1}e$. Since $s+a^{-1}e < q$, recovery of at least some of s is then trivial.

A noisy modulus is a number $q + k$ that is approximately similar to q, where k is a relatively small noise term. For a value x chosen at random and $x > q$, $x \bmod q \approx x \bmod q+k$. This introduces noise into the result, and the magnitude of the noise is determined by $k(x/q)$, where x is the value being reduced modulo $q + k$. As long as the sizes of x, q, k are selected carefully, the additional noise generated by the approximate reduction will be no larger then the noise term e and will be of no consequence. It should be noted that there exist edge cases where $x \bmod q \not\approx x \bmod q + k$. Such cases are not relevant to the correctness of the cryptosystem.

It is easy to choose a small value a^{-1} and calculate a via the extended euclidean algorithm. However, a random large a is exceedingly unlikely to have a short inverse. For a given large a, finding a modulus $q+k$ such that $inverse(a, q+k)$ is short is a hard problem. An algorithm that can be used to solve this problem can be used to solve a 1d instance of the LWE problem with only a single sample. Proof of this is presented in the following section.

4 EPQ Cryptosystem

The EPQ Cryptosystem provides a trapdoor style primitive, built from short inverses and noisy moduli. The short inverse a^{-1} and the noise term k function as the private key, while $inverse(a^{-1}, q + k)$ functions as the public key. The short inverse incorporates a power of 2 factor that enables simple recovery of the low bits of s.

The public key operation is performed modulo q, while the private key operation is performed modulo $q + k$.

This paper defines the cryptosystem in terms of a public key operation and private key operation. It does not dictate whether this is used to form a key encapsulation mechanism or a public key encryption algorithm. Standard transformations and techniques may be employed to use the cryptosystem for either purpose.

4.1 Public Parameters

There is one public parameter, the modulus q. The same q may be embedded in each implementation of the cryptosystem.

4.2 Key Generation

Private Key Generation
$k \leftarrow random(k_{size})$
$a^{-1} \leftarrow random(a^{-1}_{size}) << n$

Public Key Generation
$a \leftarrow inverse(a^{-1}, q + k)$

4.3 Public Key Operation

$s \leftarrow random(s_{size})$
$e \leftarrow random(e_{size})$
$c \leftarrow as + e \bmod q$

4.4 Private Key Operation

$s_{lsb} \leftarrow a^{-1}c \bmod q + k \bmod 2^n$

4.5 Optimization and Compression

There are a few optimizations that may be applied to the cryptosystem. The first is to utilize zeroization of the least significant bits instead of the addition of random errors. This foregoes the need to generate a large random e value, which can be relatively time consuming. This may be implemented via simple binary right shift, which has the added bonus of compressing the ciphertext. Decompression is implemented via binary left shift, and is applied during the private key operation.

The need for an integer addition circuit can be completely removed from the cryptosystem by replacing the addition of the noise term during private key generation with the exclusive-or operation. Elimination of the addition operation reduces implementation complexity, particularly in hardware.

Optimized Public Key Generation
$a \leftarrow inverse(a^{-1}, q \oplus k)$

Optimized Public Key Operation
$c \leftarrow (as \bmod q) >> e_{shift}$

Optimized Private Key Operation
$s_{lsb} \leftarrow a^{-1}(c << e_{shift}) \bmod q \bmod 2^n$

4.6 Parameter Size Selection

Correct parameter sizes are critical for both correctness and security. For correctness, it is required that $s + a^{-1}e < q$.

Padding is required to obtain a non-negligible probability of decryption failure. The padding is applied to the size of the the error term/shift amount, and results in a slightly smaller error. x bits of padding provide a probability of failure equivalent to 2^{-x}.

For a security level of n bits for the public key operation, it is required that $log(s) + log(e) >= log(q) + n$.

The following routine can be used to generate parameter sizes for a target security level n and padding amount:

```
algorithm Generate Parameter Sizes
     q_size := n * 6;
     a^-1_size := n;
     k_size := n;
     s_size := n * 3;
     e_shift := n * 4 - padding;
 output q_size, a^-1_size, k_size, s_size, e_shift
end.
```

4.7 Security Analysis

An adversary who can break the EPQ cryptosystem can solve instances of 1-d LWE with large errors and a single sample.

The LWE problem in one dimension is: given $a, as + e \bmod q$, recover s.

There are two strategies for doing so. One is to recover a valid private key from the public key, then use the private key operation to solve the problem. The other is to invert the public key operation directly.

Clearly, an adversary that can directly invert the public key operation has solved the LWE problem in one dimension.

An algorithm that can be used to recover a private key from a public key can be used to solve a random instance of the LWE problem in one dimension, by using the private key to perform the private key operation. Equivalent keys may exist, but finding one implies the ability to solve the 1-d LWE problem.

5 Homomorphic Encrytion and Backdoored Key Agreement

Other cryptosystems can be instantiated using short inverses and/or noisy moduli. Two are listed here the sake of completeness. The details of these cryptosystems are outside of the scope of this paper, and should be considered open problems and work for the future.

5.1 Homomorphic Encryption

The cipher from Fully Homomorphic Encryption Over The Integers[1] can be instantiated using a short inverse, yielding a similar scheme that should be smaller and more secure. The cipher itself is only somewhat homomorphic, the techniques presented in the aforementioned paper are what facilitate fully homomorphic encryption.

5.2 Backdoored Key Agreement

The trapdoor can be used to craft a practical, undetectable, nobody-but-us (NOBUS) backdoor for a noisy Diffie-Hellman key agreement scheme[2]. This point alone may warrant publication of this paper, as noisy Diffie-Hellman key agreement forms the basis of some other lattice based cryptosystems.

A basic sketch of a noisy key agreement is the following: There is a public parameter a and public modulus q. A private key is an integer s. A public key is a ciphertext $as + e \bmod q$. Key agreement comes from the fact that $as_1s_2 + s_2e_1 \bmod q \approx as_1s_2 + s_1e_2 \bmod q$.

The public parameter a could be a public key for a short inverse based trapdoor. Such a public key is uniformly random mod q, and is undetecable assuming the hardness of 1d LWE with large errors.

The short inverse trapdoor can be used to recover the private key s from the noisy Diffie-Hellman public key. Knowledge of s allows the adversary to legitimately perform the key agreement, providing an undetectable key escrow mechanism.

Fortunately, there is a simple mitigation for this backdoor: Each party can contribute to one half of the public parameter a, rather than one party (or software library) hardcoding a single public parameter a. Simple multiplication of two randomly selected values should thwart the backdoor and key escrow mechanism, even if one of the parties is dishonest. However, both parties must supply their half and agree on a before the protocol can begin, implying an additional round trip, degrading the performance of the key agreement scheme.

Forutnately, no real world noisy Diffie-Hellman schemes are known to operate in 1 dimension, and it is not immediately clear that this backdoor translates to higher dimension variants of the cryptosystem. It is also not immediately clear that there can be no higher dimensional variant of the backdoor.

6 Conclusion

This paper presents two novel techniques that enable a conceptually simple, performant public key cryptosystem. Key generation is fast, key sizes and ciphertext sizes are small, and the public key operation and private key operation are very efficient.

Additionally, these techniques can be used to create homomorphic encryption, as well as an undetectable backdoor and key escrow mechanism for an approximate key agreeement scheme.

References

1. van Dijk M., Gentry C., Halevi S., Vaikuntanathan V. (2010) Fully Homomorphic Encryption over the Integers. In: Gilbert H. (eds) Advances in Cryptology EUROCRYPT 2010. EUROCRYPT 2010. Lecture Notes in Computer Science, vol 6110. Springer, Berlin, Heidelberg

2. Erdem Alkim, Lo Ducas, Thomas Pppelmann, and Peter Schwabe: Newhope without reconciliation. In: https://eprint.iacr.org/2016/1157.pdf