MACHINE LEARNING FOR CRYPTOGRAPHY AND INFORMATION SECURITY

Marius Iulian Mihailescu

Faculty of Engineering and Computer Science, SPIRU HARET University
46G Fabricii Street, sector 6, Bucharest, Romania

***Abstract**: As we have observed in the last two years, machine learning techniques started to gain a significant importance and the applications based on are increasing every day. Related to information and network security, machine learning is not new and several advances have been made. The objective of the paper is to give an open discussion on the applications of machine learning in cryptography.*

***Keywords**: cryptography; information security; cybersecurity; machine learning; artificial intelligence;*

1. Introduction

Starting with 1992, Ronald Rivest, one of the founders of RSA, discussed about machine learning and cryptography in ASIACRYPPT conference from 1991. The similarities and differences presented in his talk, managed to open new directions and challenges in the research field of cryptography, by pointing out the impact that machine learning and cryptography will have on the future of applications and different environments.

Starting the current incursion in field of machine learning and cryptanalysis we will observe that both fields have so many things in common. One of the most important things that machine learning and cryptography have in common is represented by the fact that both share the same target. The cryptanalyst plays an important role, he's objective being to find the right key which later is used for decryption. The machine learning goal is to identify the proper solution using a large space of solutions that can be used.

As we mentioned above, the applications of machine learning techniques started to gain significant terrain and the attention of researchers is focused on the following directions:

- Cryptanalysis and its applications: launching attacks designed on machine learning, and providing a correct cryptanalysis for encryption algorithms.
- Cryptography and its applications: designing and implementing cryptosystems that are using machine learning techniques, and providing classification models for the traffic that is being encrypted.

Related to information and network security, there is a wide range of applications on which the attention should be focused due to the high risk at which the data are exposed, such as:

- Detection of botnets;
- Detection of Network Anomalies;
- Homorphic Encryption;
- Searchable Encryption;
- Classification of malicious code.

2. Cryptography for Machine Learning: Learning with Errors (LWE) Primtive vs. Ring-Learning with Errors (R-LWE)

LWE represents one of the most interesting mechanism for encrypting a message. The encryption is performed bit by bit, which can represents an advantage but also a disadvantage due to the hardware resources. One of the main goals of LWE is to be used as a dedicated elementary and fundamental guideline for designing new cryptographic algorithms and protocols. A such example is NewHope [8], which is a post-quantum key algorithm. The goal of NewHope was to offer a protection against different attacks that are issued using quantum computers and as well as a strong and powerful foundation for the challenging homomorphic encryption primitive. R-LWE represents a large specter for LWE problem and it is based on polynomial rings over finite fields. The goal is to use the presumption difficulty for providing solutions to R-LWE problem as well as if we are using quantum computers. R-LWE is a very powerful technique and represents a challenging foundation for future public-key cryptography protocols.

The most important advantage of R-LWE cryptography compared with LWE is based on the key length (e.g., public key, private key). As we will see later, in R-LWE the cryptographic keys are generated using the square root of the keys from LWE. To be more precisely, as an example, if we are using 128 bits for a security level, R-LWE cryptography mechanism will use public keys which will have 7000 bits length.

According to the research directions, there are three main categories of R-LWE cryptographic algorithms, such as:

- R-LWE Key Exchange (R-LWE-KE). The idea and research direction has been proposed in 2011 by Jintai Ding at University of Cincinnati. The idea proposed by Ding that is found behind LWE and R-LWE is based on key exchange mechanism. The basic idea [9] has been built using the associativity property which characterizes the matrix multiplications. The errors that are get as part of the computation process in these situations are used to serve and improve the security. In 20212 the paper has been accepted for publications and the patent called in the same year. Based on Ding's idea, Chris Peikert introduced in 2014 a key transport scheme [10].
- R-LWE Signature (R-LWE-S). In 2011, Lyubashevsky improved the identification protocol that has been introduced by Feige, Fiat and Shamir [11]. The improvement has been based on the process of converting the protocol into digital signature scheme [12]. In 2012, we are facing with another improved version of the signature protocol, that has been extended by Gunesyu, Lyubashevsky and Popplemann [13].
- R-LWE Homomorphic Encryption (R-LWE-HE). Homomorphic encryption allows computations to be performed over subtle data. Starting from the definition of homomorphic encryption, there were some improvements done on R-LWE and homomorphic encryption. The main task was to provide a certain level of security based on the key dependent messages. In 2011, Brakersky and Vaikuntanathan designed and proposed an encryption scheme using R-LWE, which is fully homomorphic, and achieving the certain level of security using the key dependent messages [16].

The followings will present a short mathematical background, pointing out the most important concepts that are necessary for a researcher but as well as for a professional that wants to provide an implementation within his software applications or complex systems.

3. Practical Implementation

LWE represents one of the most abstract method which can be used within a quantum environment for cryptography. Once we switch to the practical side of LWE and to implement simple mechanisms that deals with LWE, it is important to have in mind that we need to create a *secret key (sk)* and a random value (*randomValue*). The following step is quite straightforward, it will be required to declare a set of values (*setOfV alues[]*) and to be able to compute $p[] = t[] - sk + e$. As we will see late, *p[]* and the set of values

(*setOfV alues[]*) will form the public key.

The first implementation that we will discuss is represented by the encryption method (known as LWE) provided and defined by Oded Regev in [4]. The main idea of the encryption method is quite simple, each bit from the message (see the first line from the application in Figure 1) is encrypted using the scheme of O. Regev from [4].

In Listing 1 we have provided the full implementation source code of the LWE encryption method. We have provided a similar implementation of the method in [19] and [20], representing an extended version for some of the practical aspects based on the theoretical mechanism behind LWE and R-LWE.

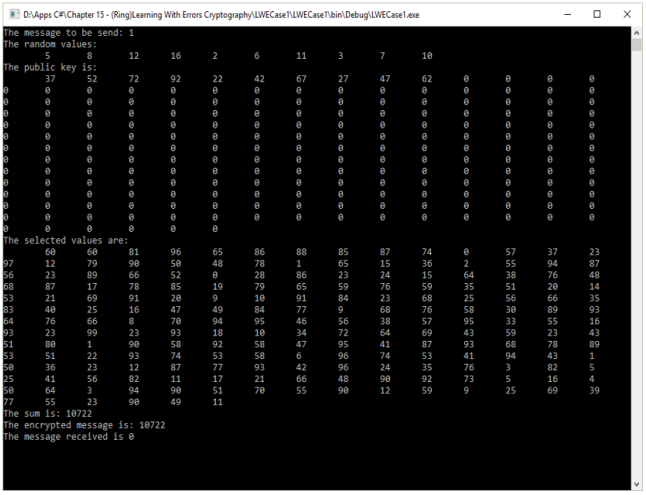

Figure 1. Running an LWE implementation in a Machine Learning context

Listing 1.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Example1_LWEImplementation
{
    class Program
    {
        public static int[] encryption_public_key = new int[200];
        public static int[] array_of_values = new int[]
                { 8, 5, 10, 12, 4, 7, 13, 5, 9, 13 };
        public static int s = 7;
        public static int e = 14;
        public static int message = 1;
        public static int val = 0;
        public static int addition = 0;
        public static int remainder = 0;

        static void Main(string[] args)
        {
            Random random_value = new Random();
            int[] output = new int[200];
            int h = 0;

            for (int b=0; b<array_of_values.Length; b++)
            {
                encryption_public_key[h] =
                    array_of_values[b] * s + e;
                h++;
            }

        for(int d=0; d< encryption_public_key.Length; d++)
            {
                output[d] = random_value.Next(
```

```
                    encryption_public_key[d],
                    encryption_public_key.Length / 2);
            }

            for(int e=0; e<output.Length; e++)
            {
                addition += output[e];
            }

         Console.WriteLine("The message to be send: {0}",
                message);

            Console.WriteLine("The random values:");
            PrintValues(array_of_values);

            Console.WriteLine("The public key is: ");
            PrintValues(encryption_public_key);

           Console.WriteLine("The selected values are:");
            PrintValues(output);

            //** compute the addition
            if (message == 1)
                addition += 1;

    Console.WriteLine("The addition is: {0}", addition);

    Console.WriteLine("The encrypted message is: {0}",
                  addition);

            //** compute the remainder
            remainder = addition % s;

            if(remainder % 2 == 0)
              Console.WriteLine("Message received is 0");
            else
              Console.WriteLine("Message received is 1");

            Console.ReadKey();
        }

        //** dealing with arrays
        public static void PrintValues(Object[] arrayOfValues)
        {
            foreach (Object i in arrayOfValues)
            {
                Console.Write("\t{0}", i);
            }
            Console.WriteLine();
        }

        //** dealing with arrays
        public static void PrintValues(int[] arrayOfValues)
        {
            foreach (int i in arrayOfValues)
            {
                Console.Write("\t{0}", i);
            }
            Console.WriteLine();
        }
    }
}
        }
    }
}
```

4. Conclusions

In this work we have discussed to give a practical implementation for Ring-Learning with Errors Cryptography using C# 8.0 programming language. The purpose of our work is to give a practical support for professionals and researchers at the same time, combining mathematical notions and mechanisms from theoretical cryptography with applied cryptography, in a modern and customized fashion.

The current contribution wish to become a starting-approach and to serve as a pool of challenges for professionals and not only, where significant contributions for this cryptography primitive can be brought in a modern way, by combining theory with practice.

At the end of our research, the main goals were achieved properly and the reader is able to find interesting things from which he will gain the following experiences:

- A solid and at the same time short mathematical foundations regarding the main concepts and definitions on which R-LWE relies.
- Identifying and experimenting the challenges of R-LWE mathematical concepts and their applicability in real life scenarios.

As for future research directions, we have proposed the following topics:

- Providing a cryptanalysis scheme for testing the security;
- Launching chosen-plaintext attack and chosen-ciphertext attack;
- Assuring the portability to quantum computers and quantum cryptography environments.

References

[1]. O. Regev, "The Learning with Errors Problem (Invited Survey)," 2010 IEEE 25th Annual Conference on Computational Complexity, Cambridge, MA, USA, 2010, pp. 191-204, DOI: https://doi.org/10.1109/CCC.2010.26.

[2]. Oded Regev. 2009. On lattices, learning with errors, random linear codes, and cryptography. J. ACM 56, 6, Article 34 (September 2009), 40 pages. DOI: https://doi.org/10.1145/1568318.1568324.

[3]. Lindner R., Peikert C. (2011) Better Key Sizes (and Attacks) for LWE-Based Encryption. In: Kiayias A. (eds) Topics in Cryptology – CT-RSA 2011. CT-RSA 2011. Lecture Notes in Computer Science, vol 6558. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-19074-2_21.

[4]. O. Regev, "The Learning with Errors Problem (Invited Survey)," 2010 IEEE 25th Annual Conference on Computational Complexity, Cambridge, MA, USA, 2010, pp. 191-204. DOI: https://doi.org/10.1109/CCC.2010.26.

[5]. Micciancio D., Regev O. (2009) Lattice-based Cryptography. In: Bernstein D.J., Buchmann J., Dahmen E. (eds) Post-Quantum Cryptography. Springer, Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-540-88702-7_5 .

[6]. Peikert C. (2009) Some Recent Progress in Lattice-Based Cryptography. In: Reingold O. (eds) Theory of Cryptography. TCC 2009. Lecture Notes in Computer Science, vol 5444. Springer, Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-642-00457-5_5.

[7]. Micciancio D. (2009) Cryptographic Functions from Worst-Case Complexity Assumptions. In: Nguyen P., Vallée B. (eds) The LLL Algorithm. Information Security and Cryptography. Springer, Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-642-02295-1_13.

[8]. NewHope – Post-quantum Key Encapsulation. Available online: https://newhopecrypto.org/.

[9]. Ding, Jintai; Xie, Xiang; Lin, Xiaodong (2012-01-01). "A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem". Available online: https://eprint.iacr.org/2012/688.

[10]. Peikert C. (2014) Lattice Cryptography for the Internet. In: Mosca M. (eds) Post-Quantum Cryptography. PQCrypto 2014. Lecture Notes in Computer Science, vol 8772. Springer, Cham. DOI: https://doi.org/10.1007/978-3-319-11659-4_12.

[11]. Desmedt Y. (2011) Fiat–Shamir Identification Protocol and the Feige–Fiat–Shamir Signature Scheme. In: van Tilborg H.C.A., Jajodia S. (eds) Encyclopedia of Cryptography and Security. Springer, Boston, MA. DOI: https://doi.org/10.1007/978-1-4419-5906-5_319.

[12]. Lyubashevsky V. (2012) Lattice Signatures without Trapdoors. \\ In: Pointcheval D., Johansson T. (eds) Advances in Cryptology – EUROCRYPT 2012. EUROCRYPT 2012. Lecture Notes in Computer Science, vol 7237. Springer, Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-642-29011-4_43.

[13]. Guneysu, Tim; Lyubashevsky, Vadim; Pöppelmann, Thomas (2012). Prouff, Emmanuel; Schaumont, Patrick (eds.). Practical Lattice-Based Cryptography: A Signature Scheme for Embedded Systems. Lecture Notes in Computer Science. Springer Berlin Heidelberg. pp. 530–547. DOI: https://doi.org/10.1007/978-3-642-33027-8_31. ISBN 978-3-642-33026-1.

[14]. Brakerski, Zvika; Vaikuntanathan, Vinod (2011). Rogaway, Phillip (ed.). Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages. Lecture Notes in Computer Science. Springer Berlin Heidelberg. pp. 505–524. DOI: https://doi.org/10.1007/978-3-642-22792-9_29. ISBN 978-3-642-22791-2.

[15]. Stefania Loredana, Nita, and Mihailescu Marius Iulian. Proposing a Secure eLearning System Based on Biometric and Homomorphic Encryption. ADLRO, 2018. DOI: https://doi.org/10.12753/2066-026X-18-221.

[16]. Nita, Stefania, et al. “Security and Cryptographic Challenges for Authentication Based on Biometrics Data.” Cryptography, vol. 2, no. 4, Dec. 2018, p. 39. DOI: https://doi.org/10.3390/cryptography2040039.

[17]. Atanasiu, Adrian. Securitatea Informaţ iei – Volumul 1: Criptografie. InfoData, 2007. ISBN: 978-973-1803-16-6.

[18]. Menezes, A. J., et al. Handbook of Applied Cryptography. CRC Press, 1997.

[19]. Mihailescu, Marius Iulian, and Stefania Loredana Nita. Pro Cryptography and Cryptanalysis: Creating Advanced Ciphers with C# and .NET. Apress, 2021. DOI: https://doi.org/10.1007/978-1-4842-6367-9.

[20]. Mihailescu, Marius Iulian, and Stefania Loredana Nita. Pro Cryptography and Cryptanalysis with C++20: Creating and Programming Advanced Ciphers. Apress, 2021. DOI: https://doi.org/10.1007/978-1-4842-6586-4.

[21]. Paar, Christof, and Jan Pelzl. Understanding Cryptography: A Textbook for Students and Practitioners. Springer Berlin Heidelberg, 2010. DOI: \ https://doi.org/10.1007/978-3-642-04101-3.

[22]. Schneier, Bruce. Applied Cryptography: Protocols, Ciphers, and Source Code in C. 20th anniversary edition, Wiley, 2015.

[23]. NIST Post-Quantum Cryptography. Available online: https://csrc.nist.gov/projects/post-quantum-cryptography.

[24]. Barrett, Jonathan, et al. "Memory Attacks on Device-Independent Quantum Cryptography." Physical Review Letters, vol. 110, no. 1, Jan. 2013, p. 010503. DOI: https://10.1103/PhysRevLett.110.010503.

[25]. Fei, YY., Meng, XD., Gao, M. et al. Quantum man-in-the-middle attack on the calibration process of quantum key distribution. Sci Rep 8, 4283 (2018). DOI: https://doi.org/10.1038/s41598-018-22700-3.

[26]. Attacking post-quantum cryptography. Available online: https://pure.tue.nl/ws/files/140306338/20191217_Groot_Bruinderink.pdf

[27]. Trushechkin, A. S., et al. "Security of the Decoy State Method for Quantum Key Distribution." Physics-Uspekhi, vol. 64, no. 1, Jan. 2021, pp. 88–102. DOI: https://doi.org/10.3367/UFNe.2020.11.038882.

[28]. Wang, Fang-Xiang, et al. "Perceiving Quantum Hacking for Quantum Key Distribution Using Temporal Ghost Imaging." Physical Review Applied, vol. 15, no. 3, Mar. 2021, p. 034051. DOI: https://doi.org/10.1103/PhysRevApplied.15.034051.

[29]. Lei Li and Zhi Li. 2020. A verifiable multiparty quantum key agreement based on bivariate polynomial. Inf. Sci. 521, C (Jun 2020), 343–349. DOI: https://doi.org/10.1016/j.ins.2020.02.057.

[30]. Chris Peikert. 2009. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Proceedings of the forty-first annual ACM symposium on Theory of computing (STOC '09). Association for Computing Machinery, New York, NY, USA, 333–342. DOI: https://doi.org/10.1145/1536414.1536461.