NDnano
Center for Nano Science and Technology

NDnano Summer Undergraduate Research 2019 Project Summary

1. Student name & home university:
Cian Levy
Trinity College Dublin

2. ND faculty name & department:
Department of Computer Science and Engineering

3. Summer project title:
Profiling Homomorphic Encryption in Secure Data Mining

4. Briefly describe new skills you acquired during your summer research:
Much of my work this summer involved topics in mathematics and computer science I was previously unfamiliar with. In order to understand the operations involved in homomorphic encryption schemes I learned about polynomial rings in abstract algebra and the Chinese remainder theorem in number theory. Through reading cryptography papers, I learned how to work independently and perform my own research to gain a better understanding of difficult concepts. During my two weekly meetings I learned how to be concise and clear when discussing technical issues. From the weekly presentations by other members in the cryptography research group a gained a broader understanding of other topics in cryptography and learned how to access the strengths and weakness of computer science publications. I acquired many new technical skills during the development of our profiling tool, including setting up a development environment on linux, debugging on linux, and how to simplify memory management in C++ through the use of smart pointers.

5. Briefly share a practical application/end use of your research:
The tools I have developed facilitate simple and easy profiling of a number of different homomorphic encryption libraries. The tool supports the evaluation of arbitrary circuits generated using HEtest's circuit generator, or manually constructed circuits. The allows the tool to be applied to analyze the performance of homomorphic encryption in many different contexts. By constructing circuits which implement the functions involved in data mining the performance of homomorphic encryption in data mining can be assessed.

6. 50- to 75-word abstract of your project:
Homomorphic encryption has many applications in data mining using private data sets. Current implementations of homomorphic encryption schemes remain slower than insecure computation by factors of several thousand. The aim of this project is to access the performance of popular homomorphic encryption libraries by applying them to evaluate circuits, and determine how their parameters influence performance in practice.

7. References for papers, posters, or presentations of your research:

1. Craig Gentry. A fully homomorphic encryption scheme. PhD thesis, Stanford University, 2009. https://crypto.stanford.edu/craig/craig-thesis.pdf

2. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based. Cryptology ePrint Archive, Report 2013/340, 2013. https://eprint.iacr.org/2013/340.pdf

3. Zvika Brakerski, Craig Gentry, Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. Cryptology ePrint Archive, Report 2011/277, 2011. https://eprint.iacr.org/2011/277.pdf

4. Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. http://eprint.iacr.org/2012/144.pdf

5. Mayank Varia, Sophia Yakoubov and Yang Yang. HETest: A Homomorphic Encryption Testing Framework. Cryptology ePrint Archive, Report 2015/416, 2015. https://eprint.iacr.org/2015/416.pdf

6. Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic Encryption for Arithmetic of Approximate Numbers. Cryptology ePrint Archive, Report 2016/421, 2016. https://eprint.iacr.org/2016/421.pdf

NDnano

Center for Nano Science and Technology

One-page project summary that describes problem, project goal and your activities / results:

Fully homomorphic encryption has become a significant area of research within cryptography since Gentry's publication of the first truly fully homomorphic encryption scheme in 2009 [1]. The rising popularity of cloud computing makes homomorphic encryption a highly attractive option to ensure privacy along with conveniences of cloud-based computing. Homomorphic encryption is also very applicable to areas such as health care, defense, electronic voting, finance, and research involving sensitive private information. Unfortunately, the performance of even the fastest fully homomorphic encryption schemes is many thousands of times slower than computation performed on insecure data. The asymptotic performance of most homomorphic encryption schemes is well understood, but there is limited work which aims to access concrete efficiency.
The ultimate goal of this project was to access how the parameters of a number of homomorphic encryption scheme implementations affect performance in data mining. Achieving this goal requires both a theoretical understanding of the parameters of the schemes, and a software testing environment where scheme implementations (libraries) can be tested and compared.

My first task was to research homomorphic encryption scheme implementations in order to familiarize myself with their parameters and understand their theoretical behavior. I read the papers for some of the best-known schemes, such as GSW, BGV and BFV [2], [3], [4]. From this research I learned about the Learning With Errors (LWE) problem upon which many homomorphic encryption schemes are based. As all of the schemes I studied were LWE-based it was important that I understood the parameters involved in generating an LWE instance as they were common between all the schemes, thus allowing the dimensional parameter and ciphertext modulus to be fixed for comparisons between schemes. However, fixing these parameters alone is not sufficient for fair comparison between schemes. In order to test the schemes under uniform conditions both the security parameter and plaintext modulus (or plaintext size) must also be fixed. All other parameters could be adjusted between tests.

My second task was to install the SEAL, HElib, and HEAAN homomorphic encryption libraries and learn how to use them. The libraries implement the BFV, BGV, and CKKS schemes ([4], [3], [6]) respectively and I was therefore already familiar with many of the underlying operations they implemented. I compiled and installed all the libraries on my Manjaro linux installation. I wrote some basic test programs to familiarize myself with the encryption, decryption, homomorphic evaluation, and serialization operations in each library.

My final task was to create a testbench which would facilitate the profiling of the three libraries. The simplest approach to this was to extend my previous test programs and measure the time taken for performing homomorphic operations. However, this approach is inflexible as the homomorphic circuits (circuits are constructed by iteratively applying homomorphic operations) would need to be hard-coded and would thus significantly reduce the maximum complexity of circuits that could reasonably be constructed. In order to support evaluation of arbitrary circuits, I elected to use the HEtest framework [5]. HEtest is a framework designed for evaluating the performance of homomorphic encryption software. The framework provides tools for generating circuits, parsing circuits, evaluating circuits, and measuring the time taken to evaluate circuits. In order to apply the HEtest framework in this project, I developed a HEtest client and server for each library. The clients were responsible for the key generation, encryption and decryption operations in each library. The servers were responsible for circuit parsing and homomorphic evaluation. I also modified the HEtest framework to support operations on non-binary plaintexts. Ultimately, I did not have time to complete the performance analysis for each library, however, the tools I have provided will enable Professor Jung to further this research.