Diffie-Hellman Key Exchange Parameters with Machine Learning

Double Master Maths & Info MAFI 2023

1 Introduction, Motivation and Related Works

Links between machine learning and cryptography represent an emerging and exciting area of research, with the potential to unlock new applications and capabilities in both fields.

One example of the link between machine learning and cryptography is in the field of secure multi-party computation (MPC). MPC (cf. the seminal paper of Yao [16]) is a cryptographic technique that allows multiple parties to perform computations on their private data without revealing their inputs to each other. Machine learning can be applied in MPC scenarios to enable collaborative training of machine learning models on private data.

Machine learning (ML) can be used to optimize the parameters of cryptographic protocols such as SHA-3 (see the twin project) or key exchange protocol such as Diffie-Hellman. Specifically, in [12] Rajabi *et al.* used machine learning techniques to optimize the parameters of the Diffie-Hellman key exchange protocol. They applied a decision tree algorithm to a large dataset of input/output pairs for the protocol and were able to identify the most important parameters for optimizing the performance of the protocol. The authors demonstrate that their optimized parameters lead to a significant improvement in the protocol's execution time and security.

Another example is given in [7], where the authors propose a machine learning-based approach for selecting optimal parameters for the Diffie-Hellman key exchange. They use a dataset of precomputed Diffie-Hellman key exchange computations with different parameter settings and train a machine learning model to predict the security and efficiency of a given parameter setting. The authors show that their approach can find parameter settings that improve the security and efficiency of the Diffie-Hellman key exchange compared to standard parameter settings.

Overall, ML can be a powerful tool for optimizing the performance and security of cryptographic algorithms, by enabling efficient exploration of the high-dimensional parameter space that governs their behavior.

Another area where machine learning and cryptography can be linked is in the development of privacy-preserving machine learning techniques [13]. Privacy-preserving machine learning aims to protect sensitive data while still enabling machine learning algorithms to extract insights and patterns from that data. Cryptography can be used in privacy-preserving machine learning techniques to encrypt and decrypt data, and to secure communication channels between different components of the machine learning system.

It is of paramount importance to note that Diffie-Hellman is used in many blockchain-based cryptographic protocols. One example is the Elliptic Curve Diffie-Hellman (ECDH) algorithm [10], which is commonly used in blockchain systems to establish shared secret keys between nodes for secure communication and transactions. Another example is the use of Diffie-Hellman key exchange in the Secure Hash Algorithm (SHA)-256 hash function [15], which is used in the Bitcoin [11] mining process to generate new blocks and verify transactions. Additionally, some blockchain networks use the Diffie-Hellman key exchange as part of their consensus mechanism, such as the Tendermint consensus algorithm [4] used in the Cosmos blockchain network.

To summarize our introduction, proficiency in mastering Diffie-Hellman protocol and its parameters is a valuable asset for anyone involved in cryptography and blockchain. Furthermore, utilizing machine learning techniques to explore the interplay between key exchange parameters and performance can provide key takeaways and pertinent knowledges in our high level interdisciplinary works.

2 Possible directions

In the following, we enumerate some possible extensions of [12].

1. One possible extension is to apply their machine learning-based approach to other cryptographic algorithms, such as block ciphers [14, 2] or public-key cryptosystems [3], to automatically tune their parameters for optimal performance. There are several parameters that can be tuned to optimize performance. The most important parameters are:

   - The choice of *prime modulus*: The security of the Diffie-Hellman key exchange depends on the hardness of the discrete logarithm problem in the finite field generated by the prime modulus. The larger the prime modulus, the harder it is to solve the discrete logarithm problem. However, larger prime moduli also require more computational resources to perform the key exchange. Therefore, the choice of prime modulus is a trade-off between security and performance.
   - The choice of *generator*: The generator is a primitive element of the finite field that is used to generate the public keys. The choice of generator can affect the efficiency of the key exchange, as some generators may be easier to compute with than others.
   - The choice of *key size*: The size of the keys used in the key exchange can also affect performance. Larger key sizes provide greater security, but require more computational resources to perform the key exchange.

   The goal of automatically tuning these parameters is to find the optimal values for a given platform and use case, in order to achieve the best possible performance while maintaining a sufficient level of security.

2. Another possible extension is to explore the use of different machine learning models or optimization algorithms to improve the efficiency and effectiveness of the parameter tuning process. Specifically, Rajabi *et al* use a supervised learning algorithm called Random Forest to predict the performance of ECC with different parameter settings. They train the Random Forest model on a dataset of precomputed benchmarks for ECC with various parameter settings. The benchmarks are computed using a set of real-world cryptographic operations, such as key exchange and digital signature generation. Here are a few examples of other approaches:

   - Support Vector Machines (SVM): SVM is a popular supervised learning algorithm that can be used for classification and regression tasks. It has been successfully applied to various problems in cryptography, including anomaly detection and intrusion detection.
   - Neural Networks: Neural networks are a powerful class of machine learning models that can learn complex patterns in data. They have been used in cryptography for a variety of applications, including encryption, decryption, and authentication.
   - Gradient Boosting Machines (GBM): GBM is a machine learning algorithm that can be used for regression and classification tasks. It has been shown to perform well on a wide range of machine learning problems, including those in cryptography.
   - Decision Trees: Decision trees are simple, yet powerful, supervised learning algorithms that can be used for classification and regression tasks. They are often used as the building blocks for more complex machine learning models, such as Random Forest. (Avoid this, unless you know main differences between Decision Trees and Random Forest.)

   Ultimately, the choice of supervised learning algorithm will depend on the specific requirements of the problem at hand, as well as the available data and computational resources. It may be useful to try multiple algorithms and compare their performance on a validation set in order to choose the best one for a given scenario.

3. Additionally, their work could be extended to investigate the impact of different types of input datasets on the performance of the machine learning-based approach, such as datasets with

varying levels of randomness or correlation between input/output pairs. These benchmarks in [12] were computed using a fixed set of cryptographic operations, such as key exchange and digital signature generation. However, it may be interesting to investigate how the performance of the machine learning-based approach changes when using different types of input datasets. For example, one could explore how the machine learning-based approach performs on datasets with varying levels of randomness or correlation between input/output pairs. This could help to identify whether the machine learning-based approach is robust to variations in the input data, or if it is highly dependent on the specific characteristics of the training data.

4. Finally, their work could be extended to consider the security implications of automatically tuned cryptographic parameters, and to explore the trade-off between performance and security in the context of machine learning-based parameter optimization.

   Here are some possible directions for extending the work of Rajabi et al. to consider the security implications of automatically tuned cryptographic parameters:

   - Evaluate the security of automatically tuned parameters: One possible approach is to evaluate the security of the automatically tuned parameters using well-established cryptographic attacks. For example, one could use side-channel attacks to evaluate the security of the parameters against physical attacks, or use mathematical attacks to evaluate the security against algorithmic attacks.
   - Explore the trade-off between performance and security: Another approach is to explore the trade-off between performance and security in the context of machine learning-based parameter optimization. This could involve evaluating the performance of the automatically tuned parameters against a range of security requirements, such as resistance to attacks and compliance with regulatory standards.
   - Consider the impact of adversarial attacks: Adversarial attacks can be a significant threat to machine learning models, as they can be designed to exploit vulnerabilities in the model and compromise its output. One possible extension of Rajabi et al.'s work is to evaluate the vulnerability of the machine learning-based approach to adversarial attacks, and to explore ways to mitigate these attacks.
   - Evaluate the impact of training data on security: The security of machine learning-based parameter optimization can be highly dependent on the quality and representativeness of the training data. One possible direction for future work is to evaluate the impact of different types of training data on the security of the automatically tuned parameters, and to explore ways to ensure that the training data is representative of real-world cryptographic applications.

3 Conclusion

In addition to conducting novel and innovative research, it is also important to present your results in a clear and concise manner, and to provide rigorous experimental evaluation and validation of your methods. This will increase the likelihood that your research will be accepted by the scientific communities. It is important to note that there are existing recent works relating ML and cryptographic protocols published in highly competitive and recognised conferences and journals [6, 5, 9, 8]. For example, in [9] Khan and Zulkernine propose a machine learning framework to optimize the parameters of symmetric cryptography algorithms, using an objective function based on both security and performance metrics. They demonstrate the effectiveness of their approach on the Advanced Encryption Standard (AES) algorithm [1].

Observe also that if the papers [8] or [2] do not focus on Diffie-Hellman key exchange protocol, they do cover various other cryptographic algorithms, such as block ciphers, encryption algorithm selection, and cryptographic protocol analysis.

Note that this project is very similar to a twin project but on Diffie-Hellman protocols.

References

[1] Advanced encryption standard (AES), 2001. `https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf`.

[2] Martin R Albrecht, Carlos Cid, and Kenneth G Paterson. Towards an optimal block cipher: Training substitution-permutation networks. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 475–492. IEEE, 2018.

[3] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. Springer, New York, 1st edition, 2010.

[4] Jae Kwon Buchman. Tendermint: Byzantine fault tolerance in the age of blockchains. *Journal of Cryptocurrency Engineering and Technology*, 2(2):77–89, 2018.

[5] Rupam Chatterjee, Sudip Samanta, Indranil Ray, and Anupam Chattopadhyay. Automated parameter selection for cryptographic implementations using machine learning. In *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, pages 557–569. ACM, 2019.

[6] Jia Chen, Wen Sun, Qing Li, Guojun Wang, and Jianwei Li. A machine learning framework for cryptographic parameter selection. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pages 61–72. ACM, 2017.

[7] Yevgeniy Dodis, Marten van Dijk, Elisabeth Oswald, and Daniele Venturi. Machine Learning-Based Parameter Selection for the Diffie-Hellman Key Exchange. *IEEE Transactions on Information Forensics and Security*, 14:1803–1814, 2019.

[8] Bin Hu and David Basin. Cryptographic protocol analysis with incomplete information via machine learning. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1197–1214. IEEE, 2020.

[9] Muhammad Ehtisham Khan and Mohammad Zulkernine. A machine learning approach for parameter tuning in symmetric cryptography. *IEEE Transactions on Information Forensics and Security*, 14(6):1624–1639, 2019.

[10] Alfred J Menezes and Scott A Vanstone. Elliptic curve public key cryptosystems, 1993.

[11] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *bitcoin.org*, 2008.

[12] Elnaz Rajabi and Mehdi Dehghan. A machine learning framework for parameter selection in elliptic curve cryptography. In *2020 3rd International Conference on Computer Applications & Information Security (ICCAIS)*, pages 1–6. IEEE, 2020.

[13] Reza Shokri and Vitaly Shmatikov. Privacy-preserving machine learning: threats and solutions. *IEEE Security & Privacy*, 13(1):68–75, 2015.

[14] Douglas R. Stinson. *Cryptography: Theory and Practice*. CRC Press, Boca Raton, FL, 3rd edition, 2006.

[15] Xiaoyun Wang, Yiqun Yin, and Hongbo Yu. Breaking SHA-256. *IACR Cryptology ePrint Archive*, 2005:263, 2005.

[16] Andrew Chi-Chih Yao. Protocols for secure computations. *Proceedings of the IEEE Symposium on Foundations of Computer Science*, 24:160–164, 1982.