The Philippa Fawcett Mathematics Prize 2019-2020

How does mathematics protect your privacy online? **Miranda Conn**

Focusing on encryption, this essay will explore how and the *extent* to which the 'hard' problems of mathematics can protect your privacy online, both now and in a future more technologically and mathematically advanced world.

How hard, is 'hard'? Mathematically, problems are considered hard if they cannot be solved in polynomial time[1], and so solving for large inputs is not practical. There exist mathematical functions which are simple to compute one way, but hard[2] to reverse; such functions are labelled 'one-way', and form the back-bone of most online privacy mechanisms.

Personal data sent online must only be read by those it was intended for, currently achieved through encryption. Encryption algorithms scramble data into a useless form for transmission, so that unauthorised interceptors cannot understand it. These algorithms are essentially just one-way mathematical functions, such that it is easy to encrypt the data, but impractically hard for an interceptor to reverse the encryption. However, their distinctive feature is a so-called 'trapdoor': a separate function which, given specific information, allows the recipient to decrypt the data. Finding a strong one-way function with a trapdoor is often complex, but reveals some of the most beautiful relationships and difficult problems in mathematics.

Our confidence in today's encryption algorithms depends solely on the 'hardness' of the mathematical problems at their roots, so we must consider: will these problems always be as hard as we believe them to be? It is natural to assume that what is considered hard today will be simple in the future, and with quantum computation on the horizon, problems used in today's cryptosystems will likely no longer be hard *enough* to defend our privacy. Thus, we must evaluate critically the mathematics of today's encryption algorithms, and look ahead to their future.

The RSA Cryptosystem

First, exploring arguably the most famous cryptosystem: the RSA algorithm (1977), which has been a titan of online privacy protection for over 43 years. RSA encryption is asymmetric, meaning separate keys are used to encrypt and decrypt the message. The decryption key is never made public, thus eliminating the possibility of an interceptor acquiring both the encrypted data and the decryption key. There are two keys, the public key and the private key, which are simply very large numbers. Public keys are visible and available for anyone to use. When a message is sent, the sender accesses the recipient's public key and uses it to encrypt their message. However, the public key alone cannot be used to decrypt the data. Access to the trapdoor that decodes the message requires the receiver's private key; this is unique to and only known by the receiver.

The RSA Algorithm

Before exploring the 'hard' problem which secures RSA, we first detail the steps of the algorithm itself, which uses relationships of modular arithmetic to establish a one-way cipher.

A basic, but very significant principal of modular arithmetic states that two numbers are 'congruent' if their difference is perfectly divisible by a given 'modulus'. For example, 20 is congruent to 6 with

[1] Polynomial time describes the time complexity of algorithms whose running time is a polynomial function on the size of the data set.

[2] Note, 'NP-Hard' is a specific set of the problems which cannot be solved in polynomial time (see explanation later in this essay). However, the word 'hard' alone is often used to refer to **all** problems unsolvable in polynomial time , and so in this essay, 'hard' will not refer to NP-hard specifically, but to the more general definition of 'hard' problems.

modulus 7, because $7 \mid (20-6)$. This can be expressed with the notation $a \equiv b(mod\ m)$, read 'a is congruent to b with modulus m'.

Say Alice wants to send a message to Bob. The first step to encrypting the message is generating Bob's keys. Bob's computer will

1. Generate two very large prime numbers, p and q
2. From this, compute $n = pq$, and $m = lcm\{p-1, q-1\}$, the least common multiple of $p-1$ and $q-1$
3. Choose the random integer, r, where $1 < r < m$ and r is coprime[3] to m
4. Determine the unique number, s, such that $rs \equiv 1\ (mod\ m)$.

Bob now declares n and r, which together form the public key, but keeps $p,\ q$ and s private. Due to the semiprime[4] factorisation problem (discussed later), p and q cannot be derived from n, and s cannot be determined without p and q. Thus, $p,\ q$ and s always remain private.

Alice now uses Bob's public key to encrypt her message. First, Alice converts her message into an integer, M, using an agreed reversible method, where $1 < M < n$ and M is coprime to n.

Alice finds the cipher text, M_c by computing $M_c \equiv M^r(mod\ n)$. Anyone who intercepts M_c, cannot determine the real message, M, because solving $M^r = M_c - kn,\ k \in \mathbb{Z}$, for M without knowing the factorisation of n is considered hard.

Bob, however, is able to decrypt M_c using the private key, s. The following beautifully neat theorem is the trapdoor that allows Bob to decrypt the message:

$$M \equiv (M_c)^s(mod\ n)$$

Since $1 < M < n$, there is only one M for which this congruence is true. Bob finds this M and converts it back to the plaintext using the same agreed reversible method used by Alice. Thus, Bob can now see Alice's message.

Exploring deeper

This algorithm is neat mathematically, yet on a surface level, a little unexciting. The most fascinating maths hides in the theory that underpins the two fundamental parts of the cryptosystem: firstly, why that final trapdoor relationship, $M \equiv (M_c)^s(mod\ n)$, holds true; secondly, how the core hard problem–the semiprime factorisation problem–prevents the private key ever being derived from the public key. Thus to further explore RSA, we can prove the final trapdoor relationship from the fundamental theorems of mathematics, showing how some of the most basic ideas in number theory are at the core of protecting our online privacy.

Proof that $M \equiv (M_c)^s\ (mod\ n)$

First, observe the following three facts:

i. If $\boldsymbol{a \equiv b(mod\ n)}$, then $\boldsymbol{ac \equiv bc(mod\ n)}$. This is obvious because if n divides $(a-b)$, then n also divides $c(a-b)$.

ii. If $\boldsymbol{a \equiv b(mod\ n)}$, then $\boldsymbol{a^k \equiv b^k(mod\ n)}$. This is because $a^k - b^k$ always has a factor $(a-b)$, and so if $n|(a-b)$, then $n|(a^k - b^k)$.

[3] Two integers are coprime if their greatest common divisor is 1.

[4] A semiprime is a number that is a product of only two prime numbers.

iii. Fermat's Little Theorem (1736) states that $\boldsymbol{a^{p-1} \equiv 1 (mod\ p)}$, where p is a prime number and a is an integer not divisible by p.

The algorithm states that M and n are coprime, and $n = pq$, so it follows that M is coprime to p, and to q.

Then by Fermat's Little Theorem,

$$M^{p-1} \equiv 1(mod\ p) \quad 1.1$$

$$M^{q-1} \equiv 1(mod\ q) \quad 2.1$$

Now $m = lcm\{p-1, q-1\}$, so $m = j(p-1) = k(q-1)$, $j, k \in \mathbb{Z}$. Thus,

$$M^m = (M^{p-1})^j = (M^{q-1})^k$$

Applying fact ii to statements 1.1 and 2.1 we obtain

$$(M^{p-1})^j \equiv 1^j (mod\ p) \quad 1.2$$

$$(M^{q-1})^k \equiv 1^k (mod\ q) \quad 2.2$$

Hence,

$$M^m \equiv 1(mod\ p) \quad 1.3$$

$$M^m \equiv 1(mod\ q) \quad 2.3$$

Thus both p and q divide $M^m - 1$. Since $n = pq$, it follows that n also divides $M^m - 1$. So,

$$M^m \equiv 1(mod\ n) \quad 3.1$$

Now, $rs \equiv 1\ (mod\ m)$, so $rs = mt + 1$, $t \in \mathbb{Z}$. Thus we can write

$$M^{rs} = (M^m)^t (M)$$

Applying facts i and ii to statement 3.1, we obtain

$$(M^m)^t (M) \equiv 1^t (M)\ (mod\ n) \quad 3.2$$

Hence,

$$M^{rs} \equiv M(mod\ n) \quad 3.3$$

Recall that $M_c \equiv M^r\ (mod\ n)$, giving $M^{rs} \equiv (M_c)^s\ (mod\ n)$. Thus since M^{rs} is congruent to both M and $(M_c)^s$ modulus n,

$$(M_c)^s \equiv M(mod\ n)$$

Therefore,

$$M \equiv (M_c)^s (mod\ n)$$

■

This proves the validity of the RSA trapdoor, and thus why the desired recipient is able to use the private key, s, to decrypt the message. The final step in fully understanding RSA, is to understand why *only* the desired receiver is able to decrypt the message. To crack the cipher, an interceptor must either solve $M_c \equiv M^r \ (mod\ n)$ for M, which becomes simple if the factors of n are known, or derive s and use the trapdoor for decryption. s depends on $p-1$ and $q-1$, thus an interceptor must either find the factors of n, namely p and q, or find $p-1$ and $q-1$ to decrypt the message. It can be shown that the latter is equally as hard as the former, and so fundamentally the difficulty of cracking the cipher is dependent on the difficulty of factoring n to find p and q.

The problem of determining two large prime numbers by factoring their product is known as the semiprime factorisation problem, and is a famous hard problem in mathematics. Semiprime factorisation is classed as an NP-problem, meaning that whilst given answers can be *verified* quickly[5], there exists no known algorithm that *solves* the problem quickly. Therefore, since n is extremely large, it is impossible in any practical situation for an interceptor to determine p and q from n. Thus, fundamentally it is this hard problem which secures RSA. Naturally, NP-problems are commonly used in cryptosystems to provide the core one-way function, as computing the cipher is very fast, and verifying a correct decryption key is also quick, yet it is practically impossible for an interceptor to reverse the encryption. Another famous NP-problem used in the Diffie Hellman cryptosystem (1976) is the discrete logarithm problem, which asks whether it is possible to determine the exponent, x, where $g^x \equiv h(mod\ p)$, p is a large prime and $h \in 1, g, g^2, \dots, g^{p-2}$.

However, although no quick solutions to these NP-problems are known, that does not certify they don't exist. No mathematician has ever been able to prove that these problems are for certain unsolvable in polynomial time, leading to the most famous question in all of mathematics and computer science: does $\mathrm{P} = \mathrm{NP}$?[6] P is the class of problems which *can* be solved in polynomial time, and so this question asks whether all problems which can be verified quickly, also have quick solutions. A proof that $\mathrm{P} = \mathrm{NP}$ would confirm there exists quick solutions to all problems that are quick to verify, meaning there would certainly exist an algorithm to crack every cryptosystem that currently relies on an NP-problem being hard to solve (almost all asymmetric cryptosystems). Thus, without a proof that $\mathrm{P} \neq \mathrm{NP}$, we cannot be 100% confident in the ability of mathematics to protect our privacy online.

Per contra, a fair and reasoned analysis must take into account that solving the $\mathrm{P}\ vs\ \mathrm{NP}$ problem is likely very far off, and for now it is widely believed that $\mathrm{P} \neq \mathrm{NP}$, so this small uncertainty in the maths of current cryptosystems should not much shift our confidence in them. What poses a much greater threat to today's algorithms, is the rise of quantum computation. The NP-complete class is a subset of NP, where every NP-problem is reducible to each NP-complete problem in polynomial time. Thus a solution to any NP-complete problem can be used to solve all NP problems. Quantum computers are particularly efficient at certain types of computation, and due to these 'distinctive non-classical characteristics' (Hoffstein, Pipher & Silverman, 2010: p.483), will be able to solve some NP-complete problems in polynomial time. Therefore, once a practical, stable and reliable quantum computer is developed, it is likely all of NP-complete, and so NP, will become quickly solvable. As a result, the development of quantum computers will force mathematicians to re-evaluate the definition of a 'hard' problem. Such problems heavily relied on by encryption algorithms today, will likely not be hard *enough* to protect our privacy in the future. There already exists Shor's algorithm

[5] 'Quickly' has a precise definition when talking about complexities, meaning 'in polynomial time'.
[6] The P vs NP problem is one of Clay Institute's Millennium Problems, and so has a $1,000,000 prize attached to it.

(1994), which would factorise semiprimes quickly on a quantum machine, and so would topple RSA. So, can we still be confident in the ability of mathematics to protect our privacy in a post-quantum world?

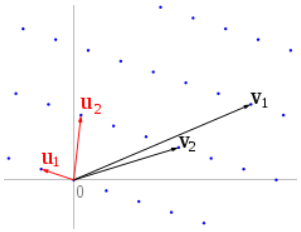
The future of cryptography – lattice-based cryptosystems?

The search for new mathematical problems which are hard even on quantum computers has brought the mathematics of lattices to the attention of cryptographers.

Take a set of linearly independent[7] vectors, $\boldsymbol{v_1, v_2, \dots v_n}$. From this a lattice, L, of points can be generated, where each point is formed by a linear combination of the vectors $\boldsymbol{v_1, v_2, \dots v_n}$ with integer coefficients:

$$L = \{a_1\boldsymbol{v_1} + a_2\boldsymbol{v_2} + \cdots + a_n\boldsymbol{v_n} : \ a_1, a_2, \dots, a_n \in \mathbb{Z}\}$$

A basis of the lattice, is any set of independent vectors that generates L, meaning any point in the lattice can be reached using a combination of the basis vectors. Below demonstrates that many different bases can generate the same lattice:



Every point in the adjacent lattice can be reached using integer coefficient combinations of $\boldsymbol{v_1, v_2}$, or of $\boldsymbol{u_1, u_2}$.

Notice that both bases have the same number of elements; this is true for all bases of L, and this number of elements is the lattice 'dimension'.

The value of lattices to cryptography resides in two problems:

1. Shortest Vector Problem (SVP): given a basis for a lattice, find the shortest non-zero vector in the lattice.
   This is difficult because many 'bad'[8] bases for the lattice exist, making it unlikely the given basis includes the shortest vector, and so finding it from the given basis complex.
   However, there exists the Lenstra-Lenstra-Lovász algorithm (1982), which can find sufficiently short vectors in a lattice in polynomial time of the lattice dimension, but becomes impractically slow and inaccurate in large dimensions. No accurate, polynomial time solution to the SVP is known, thus making it a hard problem of interest to cryptographers.
2. Closest Vector Problem (CVP): given a vector, $\boldsymbol{w}$, that is not in L, find the vector $\boldsymbol{v} \in L$ that is closest to $\boldsymbol{w}$.
   This problem is a generalisation of the SVP, but in practice is considered to be 'a little bit harder than SVP' (Hoffstein, Pipher and Silverman, 2010: p.371) , since CVP can often be reduced to a higher dimension version of the SVP.

Both of these problems are NP-Hard[9] on Turing Machines[10] (although only under certain conditions for the SVP), but have the added allure that all theory suggests they are also hard on quantum computers. This is because even the worst-case hardness of these problems can be used to prove the security of the algorithms which use them. The most promising lattice-based cryptosystem currently in development is NTRU (1996), which is supported by the Open Quantum Safe Project as

[7] Linear independence means no vector in the set can be written as a linear combination of other vectors in the set.
[8] A 'bad' basis is usually one where the angle between the basis vectors is very small, and so the different pairs of sides in the fundamental parallelepiped are very different in length.
[9] NP-Hard problems are those at least as hard as the hardest NP problems.
[10] A Turing Machine is a standard mathematical model of computation that uses only the fundamental logic functions.

a promising post-quantum encryption algorithm. Perhaps then, the future of protecting online privacy will no longer lie in the mathematics of number theory and modular arithmetic, but in the maths of vector spaces.

In conclusion, it is the fundamentally hard problems of mathematics that protect our privacy online. Whether it's the semiprime factorisation, discrete logarithm, or closest vector problem, it is not cutting edge mathematical discoveries that protect our data, but instead the problems no mathematician knows how to solve. Our online privacy relies so heavily on the 'hardness' of such problems, that we must evaluate: how hard is hard enough? Today, the semiprime factorisation that secures RSA is hard enough to give confidence in the ability of maths to prevent our bank details and passwords getting into the wrong hands; this will not be true in a more mathematically and technologically advanced world. The way in which mathematics protects our privacy online will therefore have to change, and so as we search for problems even harder than those already known, perhaps the future of encryption will rely on the mathematics of lattices? Whilst there are promising new algorithms in development, the extent to which mathematics will be able to protect future privacy is uncertain. However, the beauty of mathematics is that there is always more to explore, and always more problems to discover. Thus, I remain optimistic that a new hard problem unexplored before, or totally new approach to mathematical encryption, will allow mathematics to remain the trustworthy guardian of our privacy for many more years to come.

[Total word count: 2495]

Bibliography

i. Computerphile Youtube Channel. *'Diffie Hellman -the Mathematics bit- Computerphile'* (2017). Retrieved January 2020 from https://www.youtube.com/watch?v=Yjrfm_oRO0w&list=PL0LZxT9Dgnxfu1ILW0XnLnq3mb0L5mUPr&index=5

ii. Diffie, W. & Hellman, M.E. (1976). *New Directions in Cryptography.* Available from Stanford University archives, https://ee.stanford.edu/~hellman/publications/24.pdf

iii. Hoffstein, J., Pipher J., & Silverman J. H. (2010). *An Introduction to Mathematical Cryptography.* Springer.

iv. Hoffstein, J., Pipher J., & Silverman J. H. (1996) *NTRU: A Ring-Based Public Key Cryptosystem.* Information retrieved from source [iii], full paper available from web archives:https://web.archive.org/web/20071021011338/http://www.ntru.com/cryptolab/pdf/ANTS97.pdf

v. Holden J. (2018). *The Mathematics of Secrets: Cryptography from Caesar Ciphers to Digital Encryption*. Princeton University Press.

vi. Lenstra, A. K.., Lenstra, H. W., Lovász, L. (1982). *Factoring polynomials with rational coefficients.* Information retrieved through source [iii], full paper available at doi:10.1007/BF01457454

vii. *P versus NP problem*. Retrieved January 2020 from https://en.wikipedia.org/wiki/P_versus_NP_problem

viii. Pipher, J. [2018]. Lecture on *'Mathematical Ideas in Lattice Based Cryptography'*. Retrieved February 2020 from https://www.youtube.com/watch?v=msPrqDwLhi8.

ix. *Post-quantum Cryptography.* Retrieved February 2020 from https://en.wikipedia.org/wiki/Post-quantum_cryptography.

x. *Public Key Cryptography* (2004). Retrieved December 2019 from https://nrich.maths.org/2200

xi. Rivest, R. L., Shamir, A., & Adelman, L. (1977). *A Method for Obtaining Digital Signatures and Public-Key Cryptosystems.* Available from https://people.csail.mit.edu/rivest/Rsapaper.pdf.

xii. Shor P. W (1994). *Algorithms for quantum computation: discrete logarithms and factoring.* doi:10.1109/sfcs.1994.365700. Information retrieved February 2019 from https://qudev.phys.ethz.ch/static/content/QSIT15/Shors%20Algorithm.pdf

xiii. *UKMT Webinar on Modular Arithmetic.* Participated 24/01/20. Available at https://www.youtube.com/watch?v=ZnAW3IQNhTw.

xiv. Image source: Wikimedia Commons, retrieved from https://en.wikipedia.org/wiki/Lattice_reduction