We currently use RSA (Rivest Shamir Adleman) and ECDSA (Elliptic Curve Public-Key Cryptography) for a lot of our encryption methods. These methods are generally what is called public-key cryptography. In a public-key system, there are two keys used, one is called the public-key which is a specific number or formula used for encrypting messages, and the other is called the private key, which is generally an incredibly large prime number. Public-key cryptography works incredibly well because normal computers are not good at factoring prime numbers, which means that obtaining the private key is pointless in terms of computational time, power, and overall resources. For example, on a normal computer, a password comprised of 3763863863761 would take approximately 4 minutes to crack for a given computer, however, as we increase the length we can see that the time that it takes to crack increases exponentially. So if we just use that number two times in a row to get 376386386376137638638637 61, then the new time to crack the password becomes 79 million years. Obviously, there is more complexity to passwords than that, but it gives a good idea of how large prime numbers are harder for computers to crack. However, in 1994 a mathematician at MIT named Peter Shor came up with an algorithm that uses a lot of mathematical cleverness to create better guesses for the factors of a prime number. The algorithm works best in quantum systems, especially quantum systems with a higher number of what is referred to as qubits. In a normal computer system, the most basic unit of information is a bit, which represents a logical state where a value is either 1 or 0. Bit is actually an amalgamation of binary integer, and can be thought of as representing either an on or off state, and then using binary code and increasing layers of complexity, we get the computers that we use today. In a quantum system the most basic unit of information is not a bit, but a qubit (short for quantum bit). These qubits do not represent either on or off like a regular bit, but rather both numbers at once. Essentially, Shor's algorithm takes advantage of

how qubits function to create a system that efficiently and effectively outputs the highest probability prime factors of a number. What would normally take current computers say 79 million years to crack, with Shor's algorithm it would take quantum computers a few minutes. This is obviously an incredibly big issue, because we use public-key cryptography all of the time, and so much of the internet, hardware, and infrastructure rely on these methods of encryption that would be rendered almost useless. Luckily, we still have time before reaching that point, because according to several estimates, for Shor's algorithm to be effective enough, it would require a system that has around 1300 to 1600 qubits, and currently the most advanced systems are only beginning to near 1000 qubits. The point of this project is to help update hardware for the transition period when Shor's algorithm does become a concern to encryption, and to open avenues for future quantum encryption methods in our software and hardware systems.

https://en.wikipedia.org/wiki/Qubit#:~:text=In%20quantum%20computing%2C%20a%20qubit,with%20a%20two%2Dstate%20device.

https://news.mit.edu/2016/quantum-computer-end-encryption-schemes-0303

QRL Paper by Peter Waterland