March 2022

Encryption and Quantum Computing

Robin Wilton
wilton@isoc.org

First, the disclaimer:

“If you think you understand quantum mechanics, you don’t understand quantum mechanics.”

Prof. Richard Feynman (Nobel physicist, pioneer in sub-atomic particles and quantum computing, and world-class explainer of hard stuff.)

What follows makes heavy use of metaphors…

Main goals of this presentation

- Introduce the basics of quantum computing
- Show how quantum computing relates to cryptanalysis
- Pave the way for the following presentations

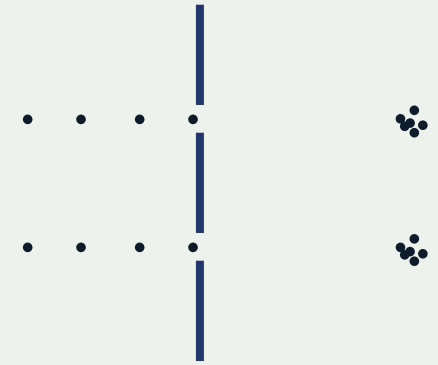
Topics

- A quantum of physics
- A bit of encryption
- A qubit of quantum cryptanalysis
- Some closing thoughts

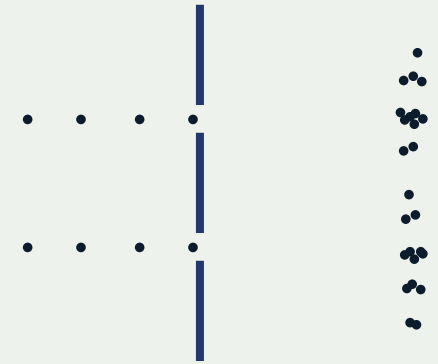
A Quantum of Physics

- Classical physics cannot explain some of the things we observe in the universe* - for instance, how light can seem to act as both a wave and a particle simultaneously.
- But quantum physics can explain these phenomena at a sub-atomic level.
- In the case of light, quantum physics says that photons "superpose" different states at the same time - a wave-like state and a particle-like state.

*For more examples, I recommend Prof. Jim Al-Khalili's amazing programme on Quantum Biology, "Let There Be Life": (https://www.youtube.com/watch?v=q4ONRJ1kTdA)



A: How a beam of photons (particles) should behave

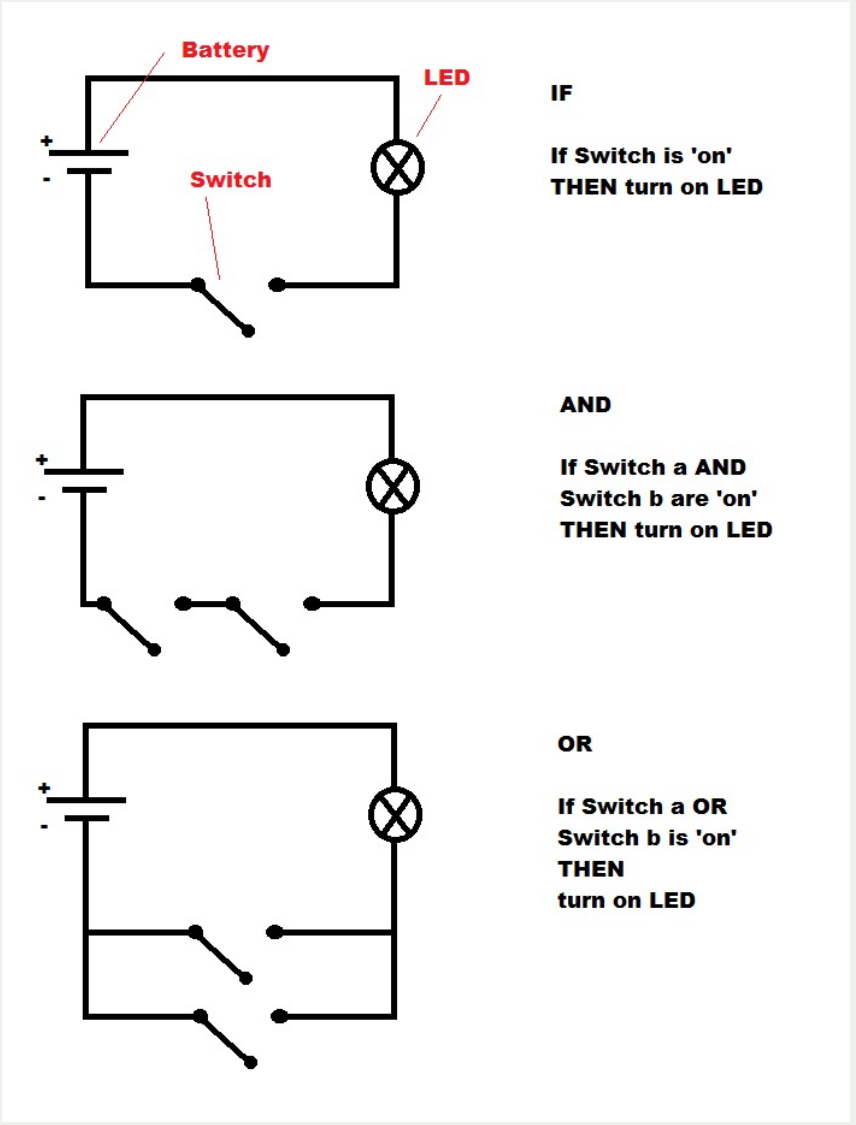


B: Photons exhibiting wave-like interference patterns

How does this relate to computing?

- Classical (digital) computing is based on ones and zeroes:
  - A bit can store a value of 1 or 0
  - Using binary values, we can do arithmetic and we can also construct switches.
  - Combining binary switches gives us logic gates: AND, OR, IF-THEN, etc..
  - Think of a light in your home with dual switching (an OR gate), or a dual-key missile launch button (an AND gate).

- Quantum computing changes that model…

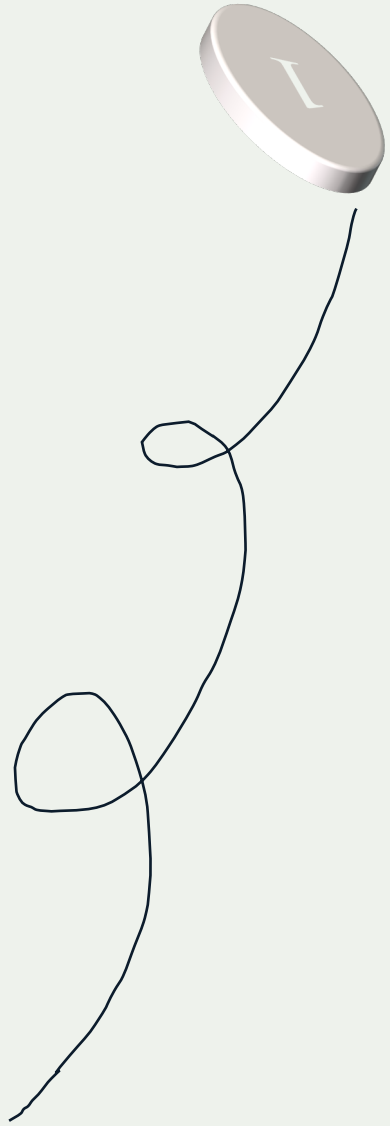


Source: http://www.thebioneer.com/wp-content/uploads/2017/04/Logic-Gates.png

Quantum computing

- Quantum computing is the application of quantum physics to computer processing. Specifically, it applies the idea of superposition to bits, allowing them to superpose the states of 1 and 0…
- … a qubit.
- This is rather like flipping a coin; you know it must eventually land in one of three states, but until it does so, it is “superposing” all three – or, at least, superposing a set of probabilities of its end state.
- When Schroedinger applied this principle to his (hypothetical) cat-in-a-box, the thought experiment was this:
    1. The cat’s death is determined by a quantum event;
    2. The event happens;
    3. The state of the event is observed.

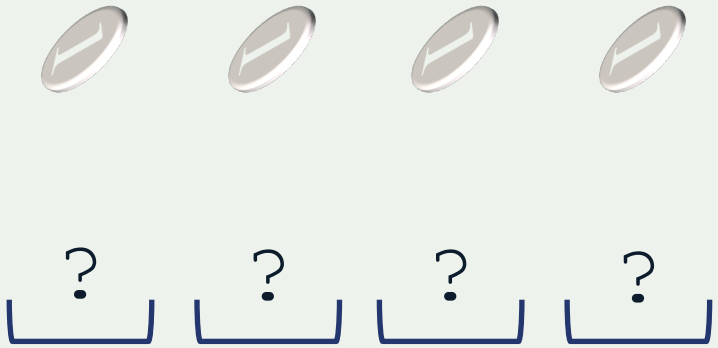
  The phrase Schroedinger used was that, until the state of the event is observed, "the living and dead cat (pardon the expression) [are] mixed or smeared out in equal parts”. Its properties (alive/dead) have many values at the same time, and all we can do is to assign them a probability.



Quantum computing

- To illustrate the potential of quantum computing, let's take a trivial example.
- Suppose you have a really basic computer, with 12 bits of storage, an add function and a compare function, and you want to do some arithmetic on two 4-bit binary numbers. Here's the sum we want to do:

```
0 0 0 1
        +
? ? ? ?
        =
1 0 0 0
```



- Using binary bits, you'd have to run through 8 possible values of the missing number, one after another, until you found the correct one.
- If we had 4 qubits to work with instead, each qubit could superpose the values of 1 and 0, and therefore all the possible values of the missing number, simultaneously – so the right answer must be in there somewhere.
- Then, "all" you have to do is identify which of the possible states is the one you're looking for. More about that in a moment.

Topics

- A quantum of physics
- A bit of encryption: symmetric and asymmetric
- A qubit of quantum cryptanalysis
- Some closing thoughts

Encryption and “hard problems”

Encryption relies on the principle that some kinds of problem are objectively hard to solve with the tools at our disposal.

The two main species of encryption used currently are symmetric (or secret key) and asymmetric (or public key).

They rely on the difficulty of different problems:

- For symmetric encryption: exhaustive search of an infeasibly large keyspace
- For asymmetric encryption: (i) factorization, or (ii) discrete logarithms

In both instances, quantum computing can help ease the task, but not in isolation.

Symmetric encryption

- Symmetric encryption scrambles data, in combination with a secret key, such that the original data can be recovered by reversing the scrambling process with the same key.
- It works like a cash box: whatever key is used to lock it, the box can be unlocked with that key or an exact copy.

- A good symmetric encryption algorithm is designed so that, in the absence of the secret key, there is no more efficient way to recover the original cleartext than by trying every possible key on the ciphertext until you hit the right one – an *exhaustive or brute force attack*.
- If the number of possible keys is big enough, the chances of finding the correct one *at random* are negligible, and the task of finding the correct one *systematically* is described as "computationally infeasible".

- In "Applied Cryptography", Bruce Schneier sets out, in terms of pure physics, what would be involved in an exhaustive attack on 256-bit symmetric keys. (2nd edition, pp. 157-8), or here, on his blog: https://www.schneier.com/blog/archives/2009/09/the_doghouse_cr.html

Symmetric encryption

The previous slide referred to brute-force attacks. To quantify the threat from quantum computing, we need to look at work factor and keyspace.

The effort required for an exhaustive search is referred to as the "work factor". Work factor is quantifiable in terms of cycles, time and money – but ultimately, in terms of matter and thermodynamics.

The keyspace is the total number of possible keys for a given key length. For binary keys, this number is 2^n, where n is the key length in bits. (So, our simple 4-bit arithmetic problem earlier had a "keyspace" of 2^4, i.e. 2x2x2x2 = 16 possible answers.)

Current symmetric algorithms should use 256-bit keys…

Each time you add a bit to the key length, you double the keyspace.

If you double the key length, the keyspace is squared.

Some big numbers…	
2^{37}	Number of stars in our galaxy
2^{77}	Approx. stars in the observable universe.
2^{92}	Mass of the Earth, in grams
2^{170}	Atoms in our planet
2^{223}	Estimated atoms in our galaxy
2^{256}	Keyspace of your browser's TLS key

Asymmetric encryption

- Symmetric algorithms suffer from the problem of secure key distribution. If you send someone a message in a locked box, how do you securely get the key to them?

- Asymmetric or public-key encryption offers a solution to this problem.

- Each user has a pair of keys: a public one for encryption and a private one for decryption. To send Bob a message, Alice looks up Bob's public key and encrypts her message with it. Bob decrypts the message with the corresponding private key, which only he knows.
- Crucially, unlike symmetric encryption, "reversing" the encryption function, with Bob's public key as input, does not recover the plaintext.

- If Alice's message is, in fact, the secret key for a symmetric algorithm, we've achieved secure key distribution – hybrid systems like this, in the form of TLS, are the most widely-deployed encryption technology on the planet.

- Guaranteeing that Bob's public key really belongs to him relies on a series of digital signatures, making them an interesting target for cryptanalysis.

Topics

- A quantum of physics
- A bit of encryption
- A qubit of quantum cryptanalysis
- Some closing thoughts

Symmetric encryption

- To attack a symmetric algorithm by brute force, the problem you're trying to solve is finding the correct key in a potentially huge keyspace.

- Using a quantum computer, with a qubit for every bit of the key, you can represent (superpose) all possible values of the key at once – however, you still have to identify which of all those possible values is the one you want.

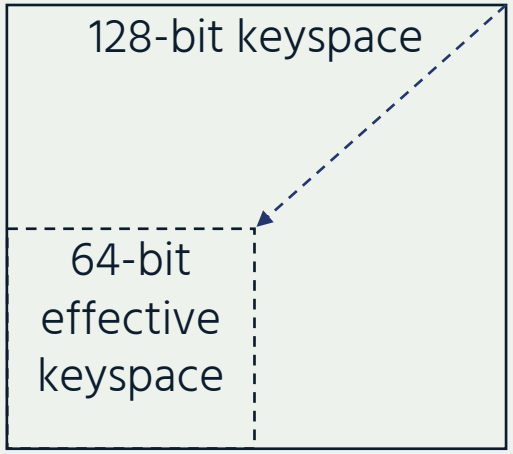
But how?

- tl;dr: it's hard. Hard to understand, hard to do, hard to explain. The answer is based on sorting algorithms and probability weightings. The Wikipedia article on "Grover's Algorithm" is a useful starting point, but quickly enters a realm where the mathematics contains more letters than numbers…

- The bottom line is that if a classical computer has to try N possible values to be sure of finding the correct one, Grover's algorithm reduces that task to $\sqrt{N}$, which, in the encryption domain, is a significant change.

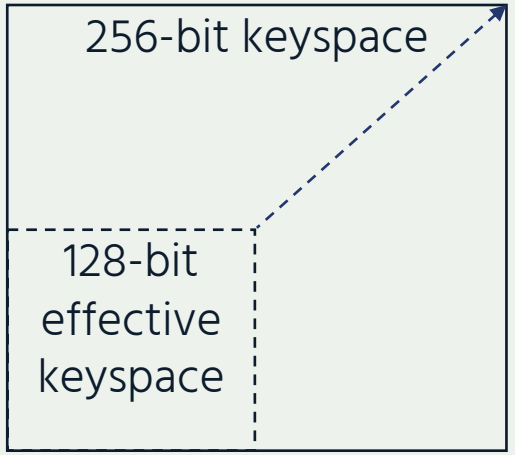
Defending symmetric encryption against quantum cryptanalysis

- A hybrid system combining quantum computing, Grover's algorithm and classical computing reduces the effective keyspace to its square root.



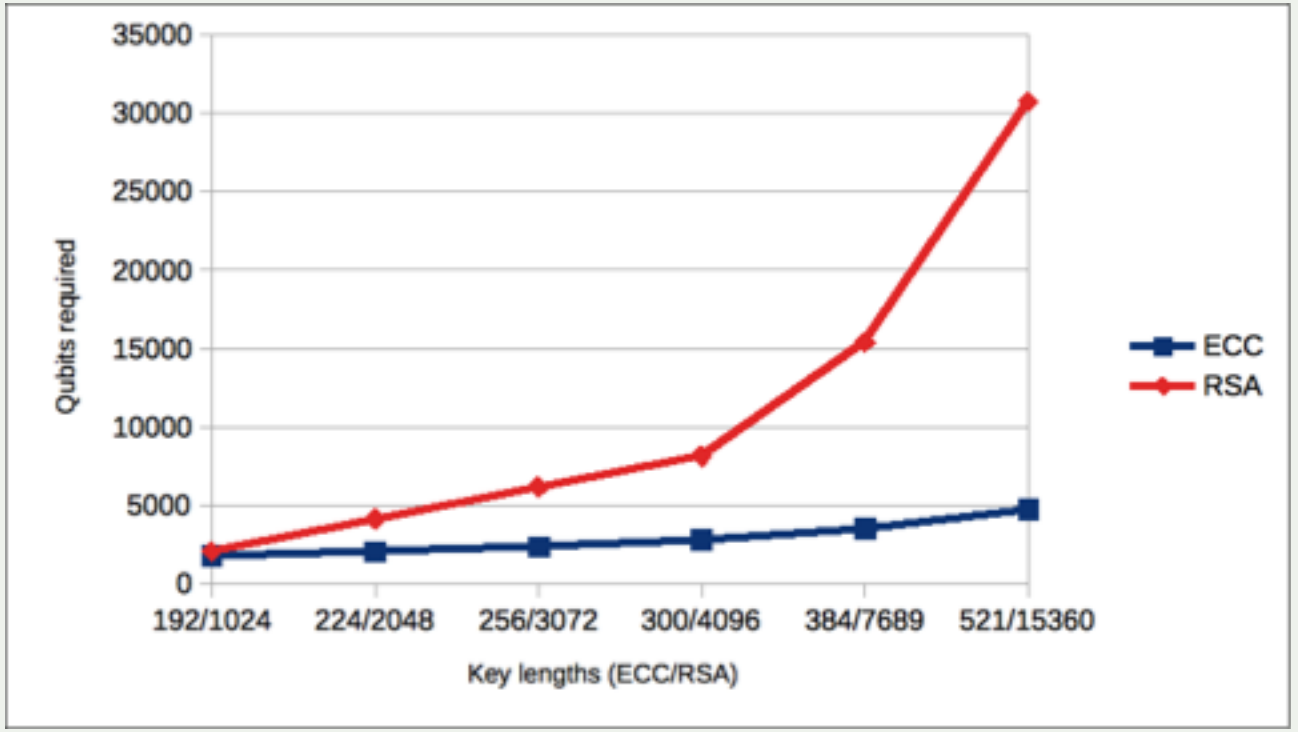
- The simple* countermeasure is to double the key length (thus squaring the keyspace again).

*replacing deployed encryption systems is known to be slow, especially if it means upgrading all your devices/applications, and those of all your communicating partners...



Quantum cryptanalysis of asymmetric (public key) encryption

- The most common public key algorithms in use are RSA and elliptic curve cryptography.
- These are based on the mathematical difficulty of factoring large primes (RSA) or solving discrete logarithm problems (ECC).
- Shor's algorithm can reduce the difficulty of these problems to the point where they could be solved "in polynomial time" – which would mean they are no longer complex enough to deliver safe encryption – but it does need a lot of qubits…
- For more about "polynomial time" as a measure of difficulty, see
  https://en.wikipedia.org/wiki/Complexity_class
  https://en.wikipedia.org/wiki/P_(complexity)



Further reading: Quantum Resource Estimates For Computing EC Discrete Logarithms: (Roetteler, Naehrig et al., Microsoft Research, Sept. 2017)

https://www.microsoft.com/en-us/research/publication/quantum-resource-estimates-computing-elliptic-curve-discrete-logarithms/

Post-Quantum Cryptography (PQC)

- The prospect of quantum computing has spurred research into mathematical problems that are hard even for quantum computers to solve.
- In June 2021 NIST held a standardization conference for the 15 candidate algorithms (7 finalists and 8 alternatives), in two categories:
  - Encryption and secure key exchange
  - Digital signature
  - Proceedings are published here:
    https://csrc.nist.gov/events/2021/third-pqc-standardization-conference

- NIST also has more PQC-related resources here:
  https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization

- It seems likely that classical computing will be able to implement harder algorithms faster than quantum computing can break them – but that's no good if you're still on an obsolete algorithm.

Topics

- A quantum of physics
- A bit of encryption
- A qubit of quantum cryptanalysis
- Some closing thoughts

Topics

- Other domains of quantum computing
- Conclusions

Quantum computing is not a single technology

- Quantum computing takes many forms, not all of which are applicable to cryptanalysis:
  - Quantum metrology
    - Sensor technology
    - Navigation
    - Time sources
  - ‘General purpose’ quantum computers
  - Quantum networking and key distribution
  - Quantum computers for cryptanalysis

- General purpose quantum computers are unlikely to be the most efficient tools for cryptanalysis, and as we have seen, resource constraints soon become a factor.

- Greater scale and efficiency can be achieved by designing for a particular problem, but the resulting system will then be specific to one algorithm, which again increases cost.

- Quantum cryptanalysis is likely to remain a specialized domain, of interest to some specific stakeholder types.

What can/should we do?

- As individual users, probably not much.
- Governments, enterprises, infrastructure providers and other organisations should:
  - Monitor this space for inflection points
  - Track progress towards PQC
  - Factor PQC into their threat model: what would happen if they suddenly had to re-encrypt all their securely archived data? What would happen if a digital signature algorithm suddenly became unreliable for signing, authentication and key exchange?
  - Do what they can to maximise algorithm agility (for instance, hybrid signature systems).

March 2022

Encryption and Quantum Computing

Robin Wilton
wilton@isoc.org