School of

Engineering and Computer Science

Te Kura Mātai Pūkaha, Pūrorohiko

CYBR 171 T1 2023

Ngā whakapūtanga o Te Haumaru rorohiko
Cybersecurity Fundamentals

Modern Cryptography III
RSA and Quantum computing

Learning goals

- Apply RSA cryptography to **confidentiality**.
- Apply RSA cryptography to check **authenticity**, **integrity** and provide **non-repudiation**
- Explain how we can **establish trust** in **public keys** using RSA cryptography.
- Explain the **threat** to cryptography posed by **quantum computing** and alternatives that might still be safe against these new types of computers.

PART I:
RSA Cryptography

RSA

- Ron Rivest, Adi Shamir and Leonard Adleman (MIT, 1977).
- Difficulty of factoring large integers that are the product of two **large prime numbers**.
- Multiplying two numbers is easy, determining the **original** prime numbers from the total (factoring) is **infeasible**.
- Example key lengths are 256, 512, 1024 and 2048 bits.
- **Four steps** – key generation, distribution, encryption and decryption.
- Public and private key generation is **the most complex** and **expensive** part of RSA.

Comparison Between *RSA* and *DHKE*

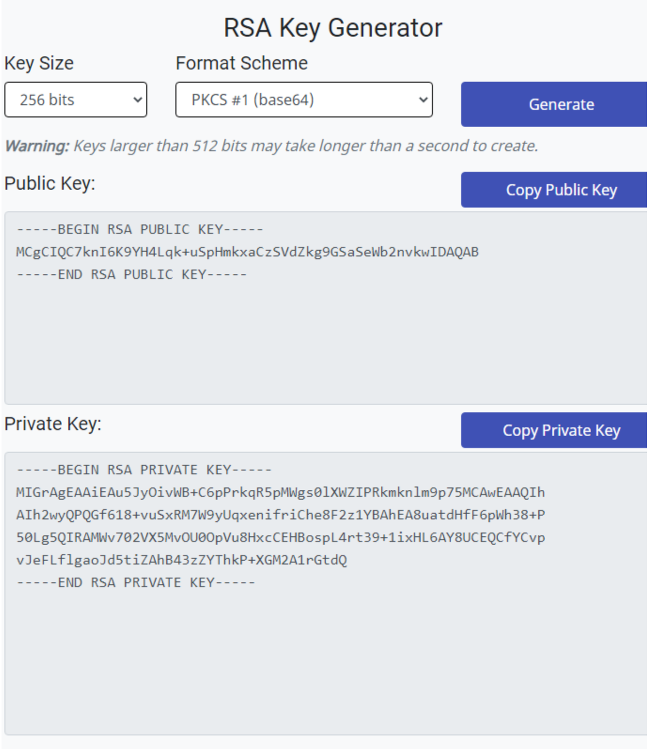
- Diffie-Hellman Key Exchange (DHKE) protocol has two public keys and one symmetric secret key.
- Diffie-Hellman susceptible to MiTM attacks unless fingerprinting is used.
- RSA uses pairs of public and private keys for asymmetric encryption and is resistant to MiTM attacks.
- **RSA supports** **authentication** of senders and messages **unlike Diffie-Hellman**.

Relationship with Symmetric Encryption

- Online commerce because **no need to meet** to set up a key for secure communication.
- Public key systems using **asymmetric** encryption and decryption is much **slower** than symmetric systems.
- Use public key system to send a new symmetric key for **each session**.
  E.g. each time I connect to https://ecs.vuw.ac.nz/

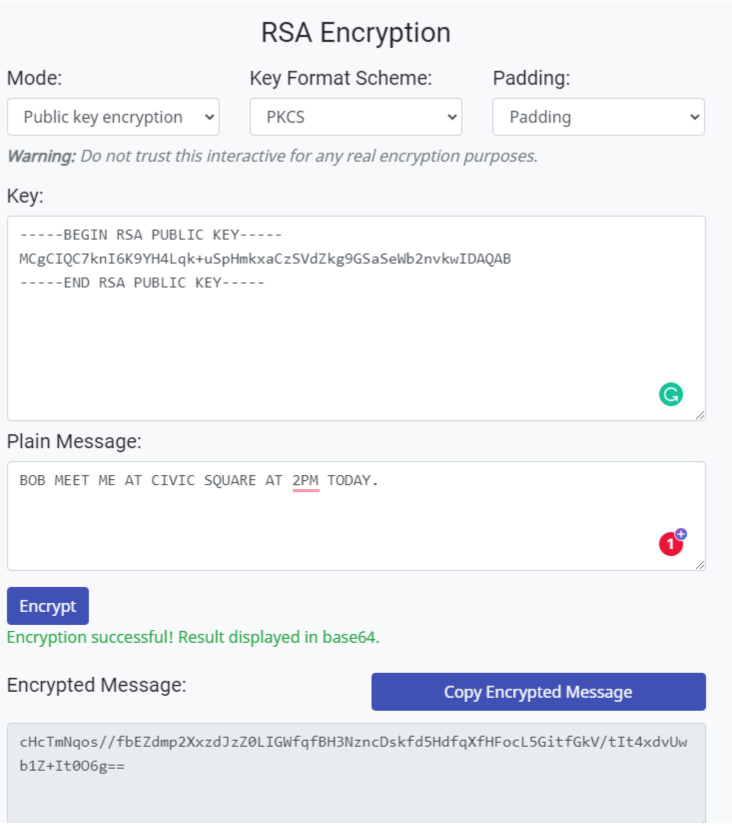
Generating the Keys

- Imagine Bob wants Alice to communicate securely with him.
- He generates a private and public key pair.
- The **private key** is secret and **never shared**.
- The **public key** is publicly announced.
- Mathematics beyond the scope of this course, illustrate using online tool. https://www.csfieldguide.org.nz/en/interactives/rsa-key-generator/



Encrypting Messages

- Alice knows Bob's public key.
- Alice encrypts her message with Bob's public key.
- RSA properties mean that only Bob's private key can decrypt a message encrypted with his public key.



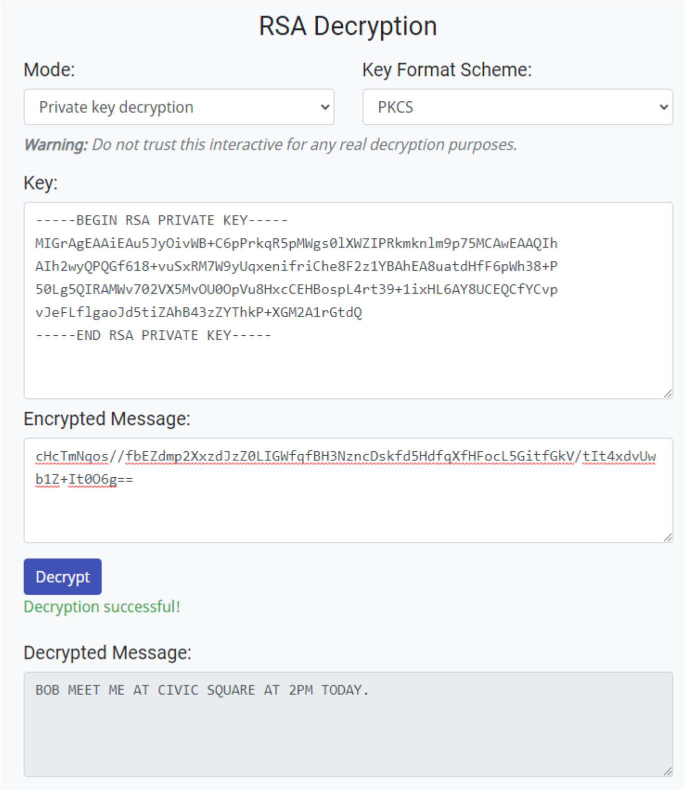
https://www.csfieldguide.org.nz/en/interactives/rsa-encryption/

Decrypting Messages

- Bob receives the encrypted message.

  cHcTmNqos//fbEZdmp2XxzdJz
  Z0LIGWfqfBH3NzncDskfd5Hdf
  qXfHFocL5GitfGkV/tIt4xdvU
  wb1Z+It0O6g==

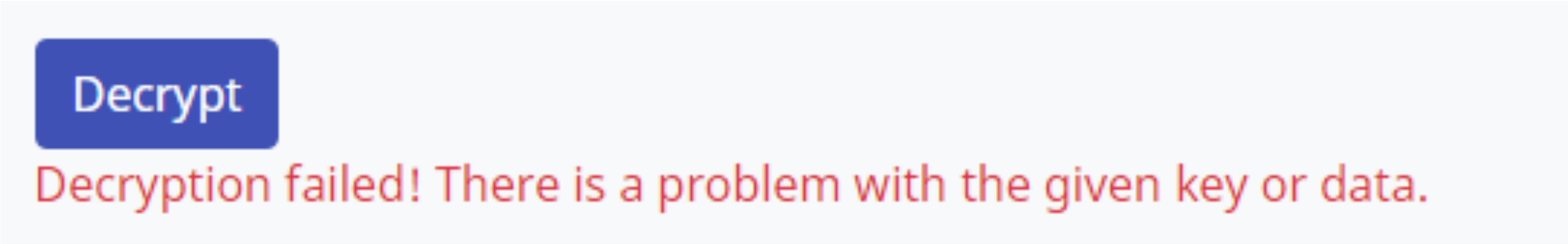
- Bob uses his private key to decrypt it.



https://www.csfieldguide.org.nz/en/interactives/rsa-decryption/

Frustrating Eve

- Bob has publicly announced his key
- Eve intercepts the encrypted message
- She tries decrypting using the public key



- She must have the private key that no-one but Bob has possession.

POLL: RSA

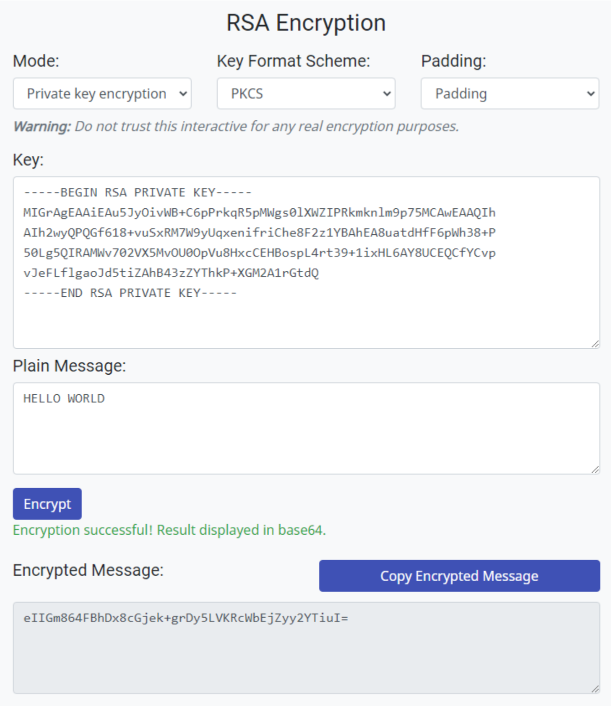
- **Bob** wants to send **Alice** a message saying, "*I am Bob, trust me*". He encrypts the message with **her public key**. **Eve** is listening to the conversation.
  - **CLAIM#1** Only Alice can read the message.
  - **CLAIM#2** Eve can read the message.
  - **CLAIM#3** Only Bob can send Alice a message "I am Bob, trust me".
  - **CLAIM#4** Eve can send Alice a message "I am Bob, trust me".

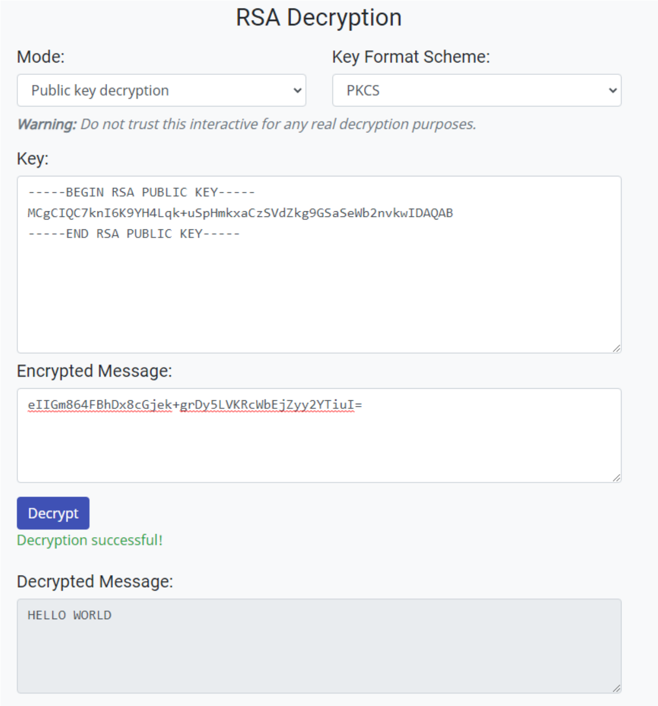
a) 1 + 3
b) 1 + 4
c) 2 + 3
d) 2 + 4

PART II:
Authenticity, non-repudiation and integrity of messages

Encrypting Using Private Key

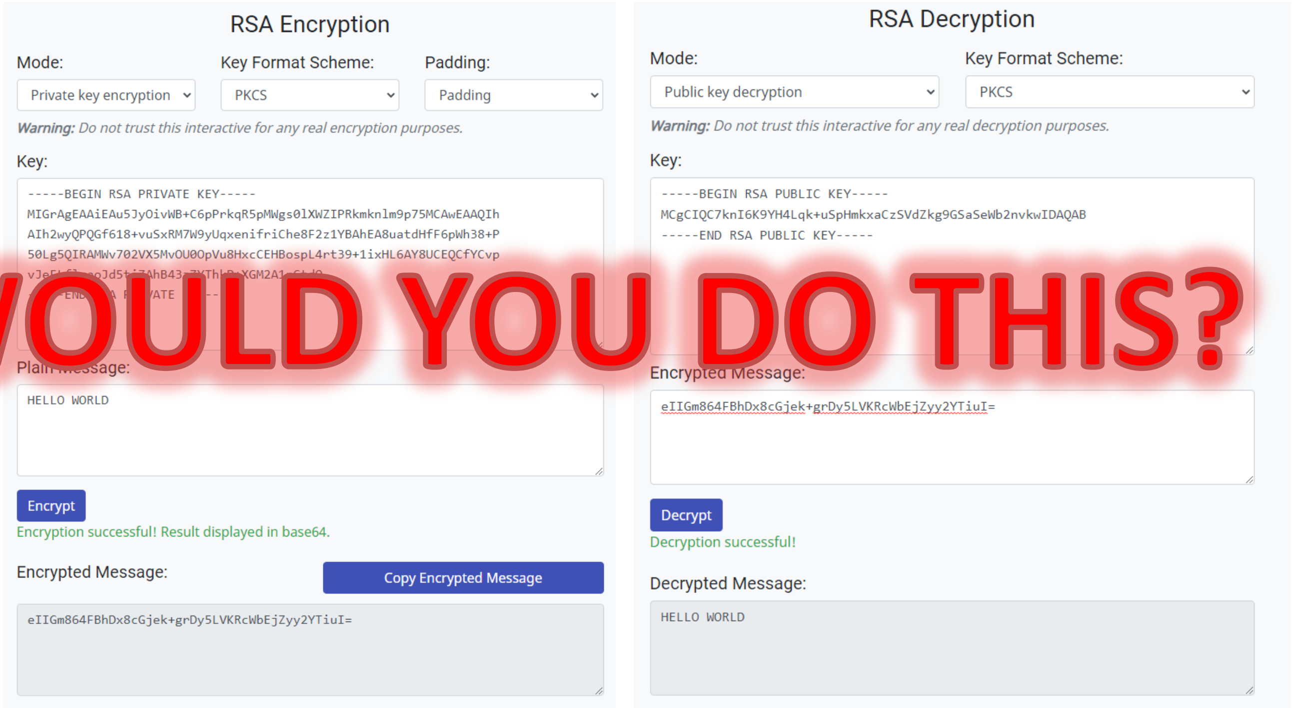
- Bob can also encrypt using his **private** key.
- Anyone possessing his **public** key can decrypt the message.





Encrypting Using Private Key

- Bob can also encrypt using his private key.
- Anyone possessing his public key can decrypt the message.



WHY WOULD YOU DO THIS?

Encrypting Using Private Key

- **Only** person able to encrypt a message using the private key is the owner of the associated public key.
- The public key will only decrypt successfully if it corresponds to the private key used for encryption.
- Decryption fails if a **different** private key is used.
- Only authentic private key holder could have encrypted message.

Authenticating Bob

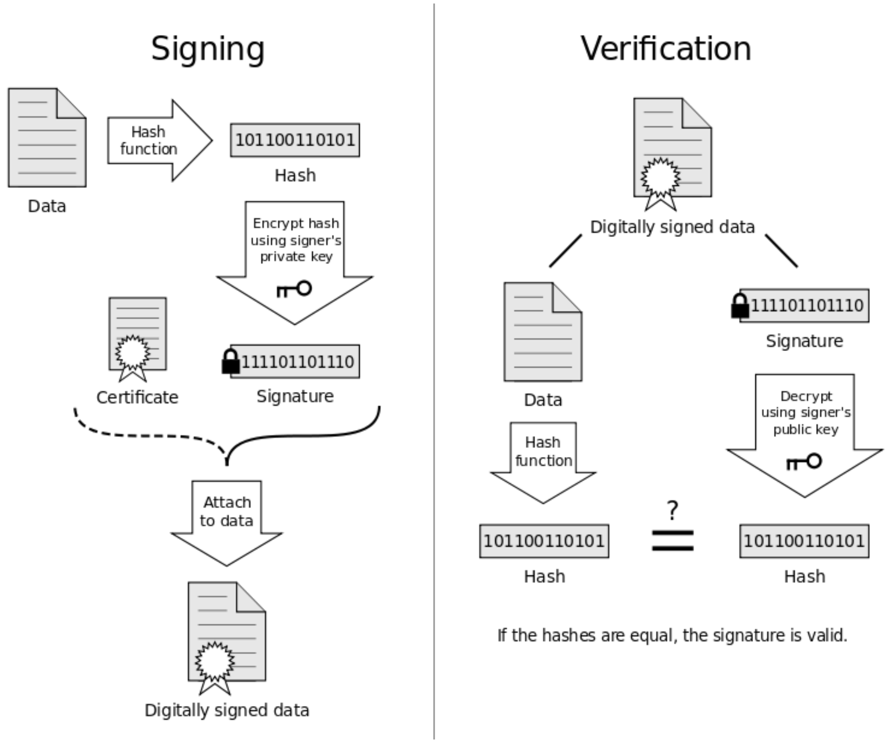
- Alice is suspicious that Eve is carrying out a MiTM attack
- Alice sends Bob a challenge phrase to encrypt with his private key
  - E.g. THE MOON IS GREEN TONIGHT
- Alice decrypts response using Bob’s public key to check that Bob was able to do the encryption
- **Would Eve ever be able to create a response that would decrypt successfully?**

Digital Signatures

- Encrypting using private key can be used as a **digital signature**.
- Three key properties:
  - **Authenticate** who encrypted a message.
  - **Non-repudiation** of the sending of the message.
  - Check that **integrity** hasn't been compromised.
- Legally binding in New Zealand
  - Electronic Transactions Act of 2002
  - Part 4 of the Contract and Commercial Law Act 2017

Digital Signatures (cont.)

- Encrypt and decrypt a hash of a message (data) for **efficiency** reasons.
- Note signature is attached to the **plaintext message**.
- Only encrypt the message if it must be **confidential**.



https://commons.wikimedia.org/wiki/File:Digital_Signature_diagram.svg

POLL: DIGITAL SIGNATURE

- Alice **received** a message from Bob with a **digital signature**. The message has been **altered** by Mallory. What would be the result of **checking the digital signature**?
  a) Decryption of digital signature fails.
  b) Decryption of message fails.
  c) Hash of message matches decrypted signature hash.
  d) Hash of message doesn't match the decrypted signature hash.

PART III:
Establishing Trust

MitM attack on RSA cryptography

- Resistance to MiTM based on ***trust in public keys***
- Example:
  - Eve might perform MiTM on Alice and Bob
  - **Eve** gives Alice **her public key** saying ”I AM BOB"
  - Alice encrypts the message using "Bob's" public key
  - Eve decrypts and re-encrypts with Bob's **real key**
  - Eve **passes** the message to Bob pretending to be Alice
- We're back to the same problem was with DHKE
  - Alice and Bob believe they are communicating securely
  - Eve can read and modify any message

How do we establish trust in public key?

- Authenticate of the binding between a public key and its owner
- Similar to authenticating a message is from the claimed sender
- Method:
    - **Publish** the **public key** in a public place
    - **Adds information** identifying the owner
    - **Sign** using a digital signature (fingerprint it)
    - The recipient can **verify** the digital signature to check integrity and non-repudiation

Establishing trust

- Example:
  - "THIS IS BOB'S KEY" + Bob's Public Key
  - Digitally sign "THIS IS BOB'S KEY" + Bob's Public Key
  - Alice receives and verifies signed public key
  - ve can't replace key with her own because will fail verification step

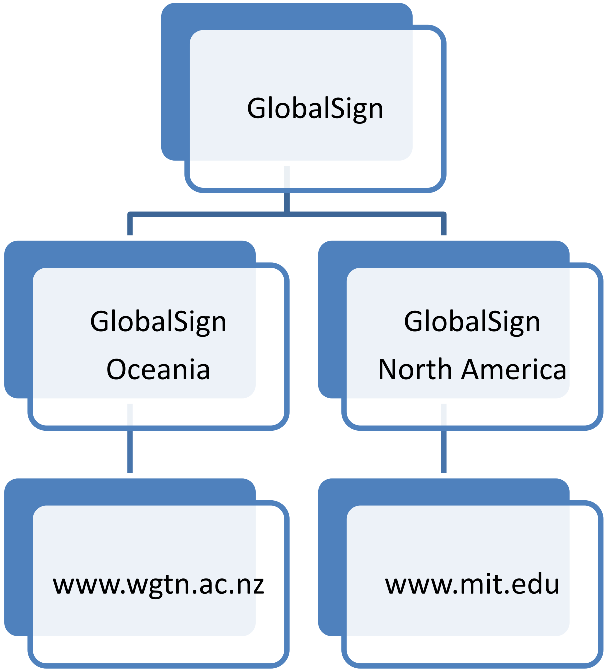
QUESTION: WHOSE KEY DO WE USE TO DO THE SIGNING?

Two Key *Approaches*

- Public Key Infrastructure (PKI)
  - Establishment of trust
  - People who have never met can establish trust
- Certificates and certificate authorities
  - Web browsers, corporate email systems, government systems
- Web of trust
  - PGP, GnuPGP, OpenPGP-compatible systems

Public Key Certificate Authority

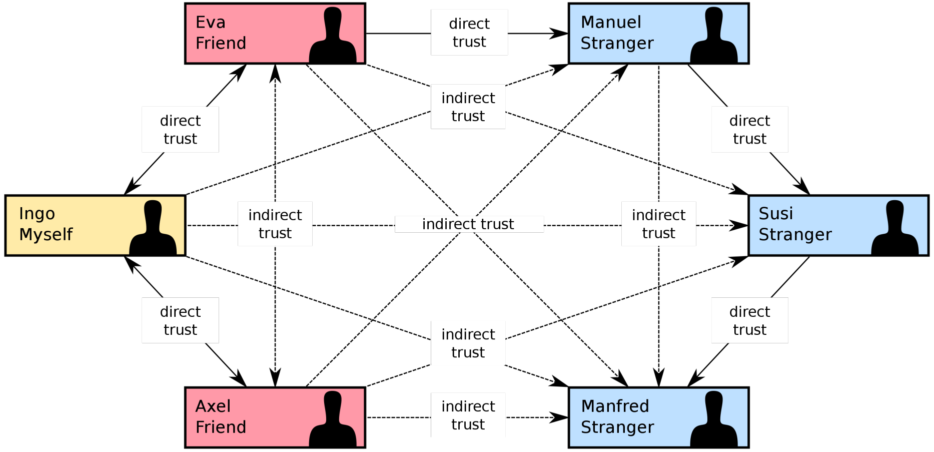
- Hierarchical relationships
  - Trusted authorities called certificate authorities (CAs)
  - CA signs organisation or person's public key
  - That key used to sign other trusted people’s keys
  - Repeat ...
- Root CA and all children are trusted 100%



More later under network security

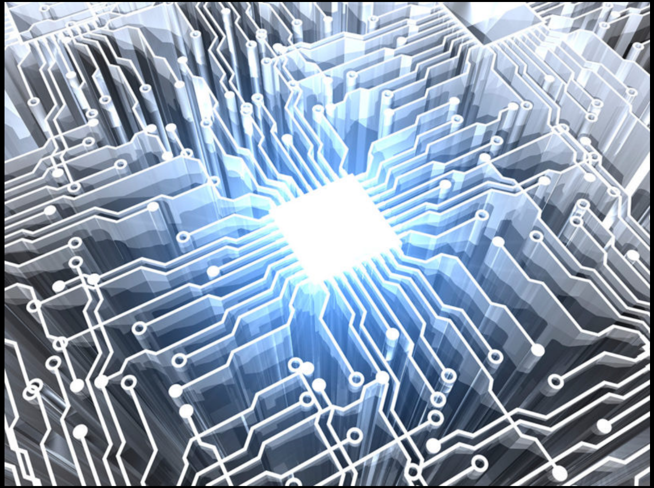
Web of trust

- You decide who to trust based upon set of relationships
  - Direct trust where two people have met and can sign each other's key
  - Indirect trust where someone trust signs someone else's key
  - Assign trust values to each and mathematical formula for calculating overall trust
- Backed by keyservers storing signed public keys



https://en.wikipedia.org/wiki/Web of trust

PART III:
The Quantum Threat



Quantum computers

- **Ordinary** computers use **bits** for computation – 0 or 1.
- **Quantum** computers use a property allowing atomic particles to **exist in more than one state** at a time.
- Quantum bits (**qubits**) have 3 states available to them - 0, 1 and both "superposition"
- You can carry out a massive number of tasks in **parallel**.
- Require supercold temperatures (-273 Celsius).
- Google, Intel and IBM have built machines.

Quantum computers

- Google provides access to its computer via a service.
  - Sycamore processor
  - 54 superconducting qubits

Quantum Computing Service

The platform enabling researchers to access beyond-classical computational resources

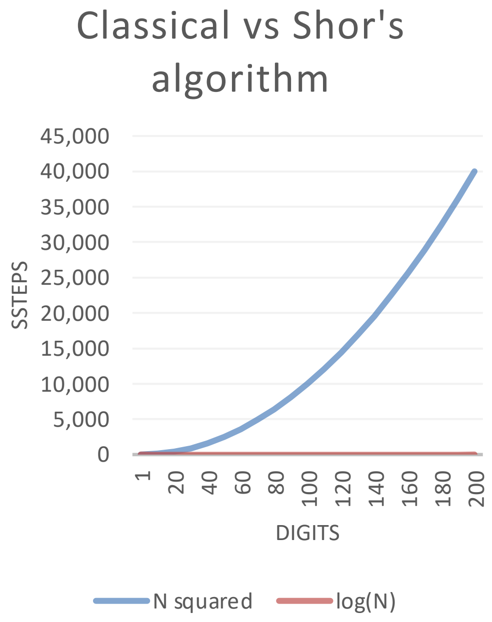
Our quantum computing service provides chaperoned access to NISQ processors and our simulator for researchers who aim to advance the state-of-the-art in quantum computing and publicly share their results in algorithms, applications, tools, and processor characterizations.

See datasheet   Read the docs

```
import cirq
import sympy

sampler = cirq.google.get_engine_sampler(
    project_id=PROJECT_ID,
    processor_id=PROCESSOR_ID,
    gate_set_name='sqrt_iswap')

circuit = cirq.Circuit(
  cirq.XPowGate(exponent=sympy.Symbol('t'))(cirq.GridQubit(5,4)),
  cirq.measure(cirq.GridQubit(5,4), key='meas'))
rabi_sweep = cirq.Linspace('t', start=0, stop=1, length=20)

results = sampler.run_sweep(circuit,
  repetitions=1000,params=rabi_sweep)
for t in range(20):
  print(results[t].histogram(key='meas'))
```

Threat **to cryptography I**

- Asymmetric cryptography **depends** upon **solving hard number problems** (primes).
  - Brute forcing a 2048-bit RSA key = 6.4 **quadrillion years**
  - Peter Shor's (1995) **quantum algorithm** for a sufficiently powerful quantum computer
  - **log N** steps versus **N^2** for classical
- 2048-bit key has approximately 616 digits
  - 3 steps versus 379,456 steps



Threat to cryptography II

- What is a **sufficiently powerful** computer?
  - Factoring 2048-bit key reliably requires billion of qubits due to noise from electronics or materials
- Security researchers felt safe until 2019, when a new more efficient algorithm developed
  - Gidney (Google) and Ekerå (KTH Sweden)
  - 20 million qubits versus billions of qubits
- Predicted lifetime of 2048-keys around **20 years**
  - IBM and Google predicted to build a 1 million qubits computer by 2030.
  - Probably doesn't matter for **credit card details** but what about **military or government secrets**?

Post quantum computing

- Some problems are **hard to solve using the quantum** approach – lattices and packing problems.
- Cryptography researchers exploiting these to make quantum-safe algorithms.
- Some old algorithms might be safe (McEliece 1978 system).
- **Bad news** is we have to change our **protocols** to use these and it isn't known yet whether you really cannot solve these problems.

https://www.wired.com/2015/09/tricky-encryption-stump-quantum-computers/

PART IV:
What’s next

What’s up next

- Assignment 1 was released yesterday.
  - Four weeks to complete
  - 20% of the final grade

- Next
  - Guest lecture
    - Application of cryptography (Blockchain and bitcoin)