Quantum Time Capsule Project:

**RSA Encryption:**

Really big number, N is the product of two prime factors, p and q, e.g. p x q = N. Therefore, only factors of N are 1,p,q,N. So, for a really large N, its very hard to find p and q since you have to sift through so many numbers – long time to break Encryption (p and q are needed to break encryption).

*Is there a faster way to break encryption other than guessing the factors ?*

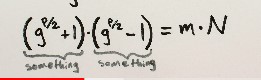
*Yes*

**Shor’s Algorithm:**

Mathematical fact = For two numbers A,B that have no common factors : A^P = m\*B + 1 for some power p and multiple m.

e.g. let A, B = 7,15: 7^4 = 160\*15 +1

So for big N and some guess, g:



* These are the two new factor guesses for p and q for N. 37.5% chance of finding factors using Euclid’s Algorithm with a random guess, 99% chance with 10 guesses.

*How to find P?*

Very long to find P on classic computers.

Quick on quantum computer – simultaneous calculate a bunch of answers for a single input.

* Send a superposition of possible numbers for p, computation will be done simultaneously on all of them with the remainders (we want remainder 1) .
* The solution is that the remainders are periodic for powers added by multiples of p, e.g. if we guess a power 42 with remainder 3, power 42 + p will also have a remainder 3 ext…
* Therefore, if we measure for remainders of 3, we will get all the powers that give remainder 3 and period of p between them, therefore we can find p.
* Frequency = 1/p, therefore we use a quantum Fourier transform – it gives 1/p.

*Quantum computers can’t do it atm since they don’t have enough memory, which only allows factoring of small numbers.*

**Quantum Time Capsule Website Development Timeline**

1. **Initial Setup on PythonAnywhere**
   * **Platform**: Started the project on PythonAnywhere, creating a free account to host the website.
   * **Sitemap & User Flows**: Shash helped design a sitemap and user flows to determine the website’s structure.
   * **HTML Development**: Created HTML files for each page following the sitemap, establishing the website’s foundational structure.
2. **UI Enhancements**
   * **Design Improvements**: Over time, enhanced the user interface by adjusting colors, adding buttons, and refining visual elements to improve usability and aesthetics.
3. **Migration to AWS**
   * **GitHub Integration Needs**: Transitioned from PythonAnywhere to AWS for automated GitHub integration, allowing changes pushed to GitHub to update the live website without manual intervention.
   * **Docker Setup**: With Dominik Koller’s assistance, set up Docker on the local machine to containerize the website, which allowed local access before pushing it to AWS for global access.
   * **Docker-AWS Connection**: Configured Docker to work with AWS, establishing access to the website via an Elastic IP (13.48.64.218), enabling public access.
4. **SSH Setup**
   * **SSH Commands**: Used SSH commands (like ssh -i quantum\_time\_capsule\_aws.pem ec2-user@13.48.64.218) to securely connect to the AWS EC2 instance associated with the Elastic IP.
5. **Repository Management**
   * **Gitignore Setup**: Created a .gitignore file to exclude specific files (e.g., sensitive data, configuration files) from the GitHub repository.
   * **Sensitive Data Handling**: Added the .env file to the .gitignore to prevent sensitive information from being committed. This file holds environment variables for passwords and other confidential details.
6. **Docker Configuration**
   * **Dockerfile & Docker-Compose**: Created a Dockerfile to define the website’s image configuration and a docker-compose.yml file to manage components like the database, Traefik (reverse proxy), ports, and containers.
   * **Docker Security**: Configured Traefik within docker-compose to handle secure traffic routing, simplifying management of multiple containers and handling HTTPS requests if needed in the future.
7. **Environment Variable Configuration**
   * **Local and Remote Environment**: Since the .env file is excluded from GitHub, manually created it on the AWS instance using vim .env to ensure consistency in environment variables across the local and live environments.
8. **Domain and Final Touches**
   * **Temporary Domain**: Currently using Dominik’s domain as a placeholder for the website; planning to acquire a dedicated domain soon.
   * **Bug Fixes**: Working through bugs introduced by the AWS migration, resolving compatibility issues to ensure website stability.