Final Report

Quantum-Safe Cryptography: Mitigating Vulnerabilities in Post-Quantum era

Unit: COIT20265

Student 1: AYUSH KESHAR PRASAI (12198371)

Student 2: JALAY SHAH (12232969)

Student 3: RONIT MAHESHWORI (12179419)

Student 4: VIRAJSINH RAHEVAR (12198387)

Group : PG-15

Project Mentor: MOHAMMED MOHAMMED

Date: 2024/10/07

CQUniversity Australia

# **Introduction**

The topic of this project is Quantum-Safe cryptography: Mitigating Vulnerabilities in Post-Quantum era. The topic relates traditional methods of encryption and how advancement and invention of large-scale quantum computers which is inevitable will affect encryption methods being used today with all the vulnerabilities outlined and ways we can mitigate such vulnerabilities in post-quantum era.

This report conceptualizes advancement from modern era when classical computers are considered rudimentary and a thing of the past. Such era in this report will often be signified as post-quantum era. Post-quantum era refers to the technological advancement in computational ability that any form of security relied upon in modern times through calculation from classical computers are now vulnerable.

The aim of this project is to:

* Identify vulnerabilities and risks related to traditional cryptography with advancement of quantum computers.
* Analyse various encryption algorithms to find vulnerabilities in cryptographic techniques.
* Compare such encryption algorithms to find the safest cryptographic standards.
* Provide general probability of when such vulnerabilities might be a risk factor.
* Advance encryption policies to post-quantum cryptography standards.
* Analyse ways of mitigation of such vulnerabilities.
* Document how post-quantum cryptography might affect traditional cryptography in all industries that relate to business, government and general user.
* Research and document new cryptographic algorithms safe for post-quantum era.
* Provide how transition might occur or is necessary from traditional cryptography techniques.

The problem this project aims to solve will be of analysing various risks associated with traditional cryptography that includes study of vulnerabilities of traditional algorithms from quantum computers. The problem is largely based on how some of the algorithms that are widely in use might not be able to cope with quantum computing when large-scale quantum computers can be used by threat actors to implement present day theoretical algorithms such as Shor’s algorithms.

Up until now, cryptography is practised when algorithms are enforced, and standards are maintained. Post-quantum era now imagines an edge in this constant battle between the attacker and the protector through means of exceptionally powerful hardware. Thus, quantum-safe cryptography refers to post-quantum era where standards, algorithms and interpretation of data is practised with depiction of ability of quantum computers. This report focuses primarily on finding vulnerabilities in modern day cryptography from post-quantum era and mitigating such vulnerabilities for quantum-safe cryptography. Furthermore, this report studies vulnerabilities of algorithms used for cryptography today and assesses its risks to post-quantum cryptography.

# **System Overview**

The system in design for this project in essence boils down to two implementation parts i.e.

1. Showing vulnerabilities in RSA through **Fermat’s algorithm**.
2. Decryption of encrypted message through **Shor’s algorithm**.

It is noted that unlike other system implementations, the system overview for this project requires understanding of why the implementation is needed for this project. This can be further simplified when each of the system implementation is broken down into further sub-headings based upon the tasks completed for this project such that each implementation gives accurate portrayal of mission objective for this task. Thus, to design such implementation,

1. **Showing vulnerabilities in RSA through Fermat’s algorithm ; consists of:**
   1. **Comparison between present day computational ability to post-quantum computational ability for cryptography.**
   2. **Listing of present-day algorithms and assessing vulnerabilities.**
   3. **Technical description of RSA (vulnerable) and its justification.**
   4. **System implementation to show vulnerability in RSA from quantum computers through Brute-force attack.**

Similarly,

1. **Decryption of encrypted message through Shor’s algorithm; will provide opportunity to:**
   1. **Derive risk associated with modern day cryptography from quantum computers.**
   2. **Prove the need for transition to post-quantum cryptographic techniques.**
   3. **Proposes such transition.**

## **Showing vulnerabilities in RSA THROUGH Fermat’s algorithm.**

To understand post-quantum cryptography, let us first dive into cryptography today. Since the birth of internet, it can be argued that there are equal number of exceptional practitioners that have used the combination of hardware and software to protect and attack the transmitted data in various ways. There is this thin thread of security that has revolutionized cryptography which depends upon standards, algorithms and combination of 0’s and 1’s. it is due to this standard which everyone follows, algorithms that is calculated, and 0’s and 1’s that are interpreted in specific manner from which confidentiality, integrity and availability is provided to the data over the internet or anywhere else.

Quantum computers now pose threat of exceptional hardware from which calculations done will surpass the ability of classical computers. Due to such ability, cryptography in practise now will be vulnerable to post-quantum era. Such vulnerability can be assessed from the start by understanding the computational ability of quantum computers first. Thus,

### **Comparison between present day computational ability to post-quantum computational ability for cryptography.**

This is the first instance in this project where a technical artefact is referred. The reference is **PG15-Comparision-of-abilities-1.1.docx.** From this artefact, we can summarize some key aspects that are essential **to implement Fermat’s algorithm**. The key aspects taken into consideration are:

* **Algorithm efficiency**

Algorithm efficiency in the context of this project and specially to the system implementation to show vulnerability in RSA refers to the ability to do factorization of large numbers. **It is concluded that post-quantum computer’s ability to factorize large numbers efficiently will initiate the vulnerability of cryptographic techniques.**

* **Speed**

Speed in which the post-quantum computers operate will be the key to irrelevance of modern cryptographic techniques. Due to speed in which factorization can be performed by post-quantum computers, **RSA is also considered vulnerable**.

### **Listing of present-day algorithms and assessing vulnerabilities**

This part of the project outlines that out of two types of algorithms in use for cryptography today, symmetric algorithm is not vulnerable to quantum computers whereas all asymmetric algorithms are vulnerable (RSA). The technical artefact it refers to for such conclusion is **PG15-Algorithm-vulnerabilities-1.2.docx**.

* **Symmetric cryptography**:

All symmetric cryptography is quantum safe. Symmetric algorithms such as SHA and AES do not rely upon mathematical calculations making it quantum safe(Azure, 2024).

* **Asymmetric cryptography**:

RSA is an asymmetric cryptography algorithm consisting of pair of keys i.e. public key and private key. The keys are generated with cryptographic algorithms that rely on mathematical calculations based on one-way functions thus making it vulnerable to quantum cryptography. RSA amongst other asymmetric algorithms is chosen in modern day cryptography for its ability to provide key distribution and secrecy along with digital signatures which reduces the use of multiple asymmetric algorithms.

### 1.3 **Technical description of RSA (vulnerable) and its justification**

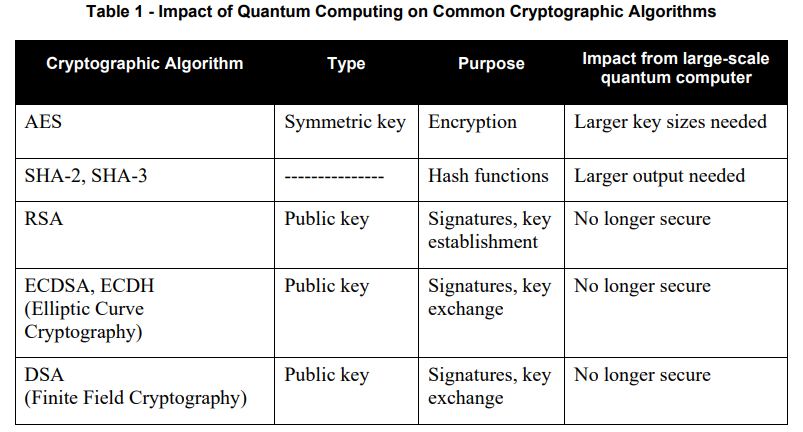


Figure 1 : Impact of quantum computing on Algorithms

From the sub-headings above, RSA is the most relied upon and vulnerable algorithm to quantum computers and is no -longer secure for post-quantum cryptography because of the **Algorithm efficiency** of quantum computers from which prime numbers can be factorized efficiently thus, breaking the RSA which is depicted **through system implementation of Fermat’s algorithm below**:

### **System Implementation to show vulnerabilities in RSA from quantum computers through brute force attack (Fermat’s Factorization Algorithm).**

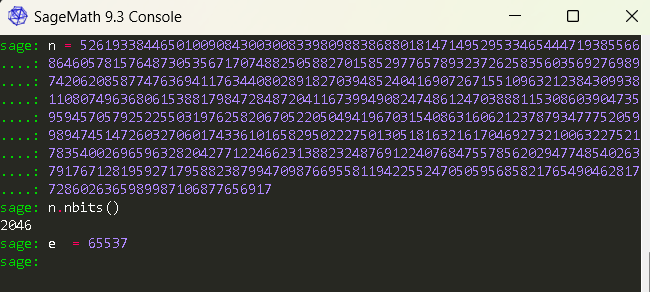
Now that the computational ability of quantum computers to solve complex calculations efficiently is understood, it was realized that RSA, which is the most used cryptographic technique, is vulnerable through factorization of large numbers. All the sub-headings above have led to this system implementation in which we break the RSA using Fermat’s factorization algorithm.

To do so, we must first make some assumptions since implementation of this system is done on a classical computer. For implementation purposes we have chosen python and sagemath as our environment. It is referenced in technical artefact **PG15-System-Implementation-Tools.docx**.

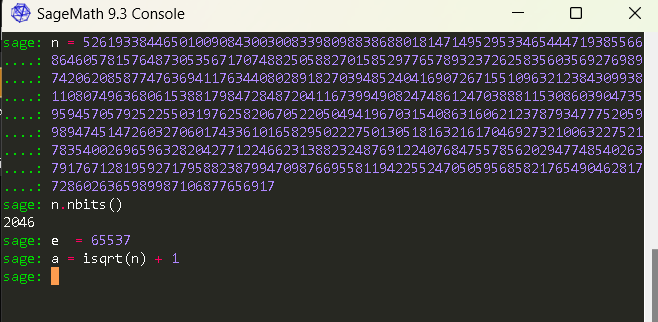
Computational ability of quantum computers mainly, **algorithm efficiency** can be assumed in this implementation and is referenced to the technical artefact **PG15-Fermat’s-algo-1.4.docx** when:

* Fermat’s algorithm can be used in classical computer to find the private key from public key of RSA when keys are chosen very close to each other and not in random **to imitate algorithm efficiency of quantum computer on a classical machine.**
* n is a composite number because it produces two prime numbers when **prime factorization** is conducted.

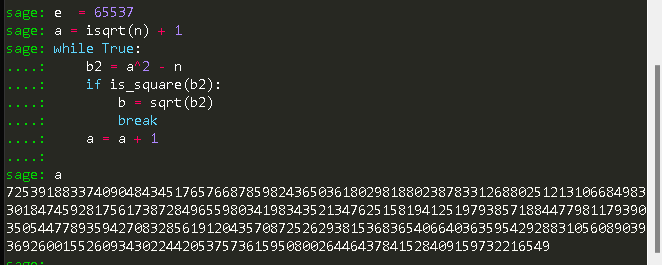
Therefore, we take the value of n,



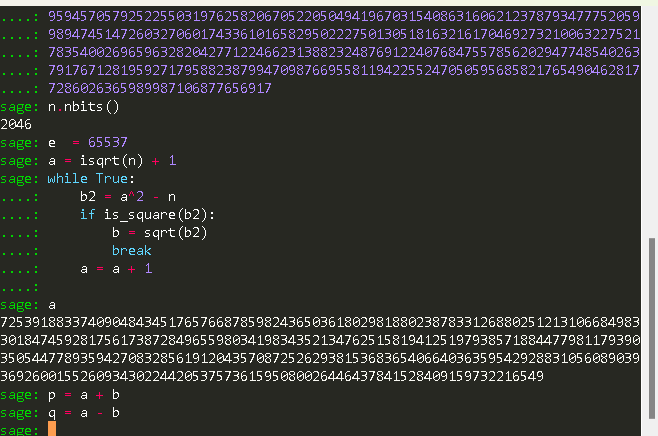
Here n is a 2046-bit number not taken at a random and exponent (e) is given as 65537, which makes up a private key (n,e) in RSA. d is a private key (d) whose value when determined through factorization from the value of n and e will in theory have broken RSA in post quantum era. To advance,



Here we apply a ceiling function a , whose primary objective is to find the value of n to its nearest integer which can be square rooted such that later, the value of p and q which makes up n i.e. n = (p,q), where p and q are both prime numbers since n is a composite number can be derived using the Euler’s totient function.

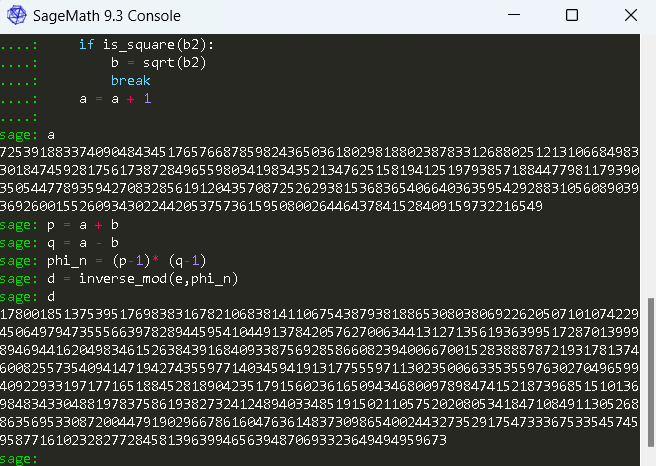


Here we introduce a loop to find the value of a into the ceiling function and round it up into the nearest integer until the value of b is found. This is an accurate depiction of a **Brute force attack** where all possible combination of number is tried until the correct value of b is found.



We have it given that the value of p and q is as depicted above since n = (a^2 – b^2) as it is a composite number. Since both a and b can be squared and make up the value of n, we can say the value of p and q as such or vice-versa.

Now we have the value of a and b therefore has the value of p and q where n = (p\*q).



Finally, we calculate the Euler’s totient function of n i.e. φ(n) = (p-1)\*(q-1). This will give is the value of φ(n) or (phi\_n) as in the code picture above. After that the value of d is found using another equation. what this equation basically means is we want to find a number d, which when we multiply by n will give us an intermediate value than can be reduced by mod φ(n) which will give us 1. Thus, from this equation (e\*d) ≅ 1 mod φ(n), we calculate the value of **private key d** and prove that RSA is broken through Factorization of prime numbers from public key (e,n).

## **Decryption of encrypted message through Shor’s algorithm:**

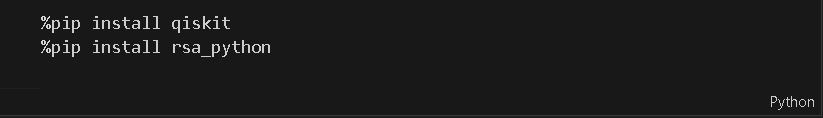
Shor’s algorithm:

Shor’s algorithm is the standard bearer for all quantum algorithms. Although it was discovered three decades ago, Shor’s algorithm remains the key reason for investment of billions of dollars into quantum technologies as it presents clear and evident danger for national security, financial systems and for all cryptography.

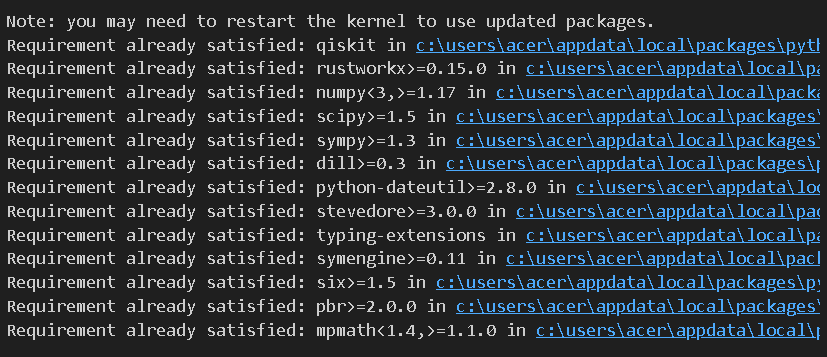
At a high-level Shor’s algorithm is easy to understand. It begins with using an integer smaller than the number to be **factored**. The greatest common divisor (GCD) is then calculated classically between these two numbers to determine whether the target number has already been calculated accidentally. This is when a **quantum computer** comes into play. A quantum computer would then be used to determine the results for cryptographically safe number. Only a quantum computer could evaluate the results and determine whether the sought-after integer could have been calculated or not and if so, use a random integer for testing.

To the use of this project, Shor’s algorithm is framed such that **modular inverse** is determined to find the value of the private key and therefore **crack the cipher text**.

To initialize this system implementation, we are using Microsoft’s Visual studio Code. We then install Qiskit and python to our environment. The codes are referenced from technical artefact **PG15-Shor’s\_Algo2.ipynb.**

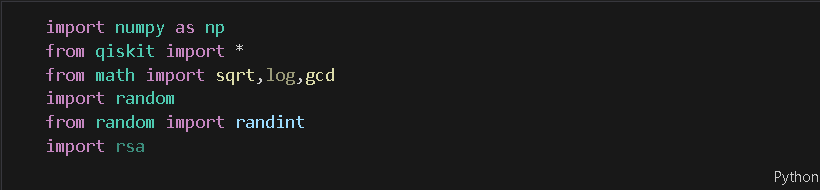


These installations make the environment ready to accept the python code and allows the use of Qiskit and python packages to be installed and functions to be used to simulate a quantum computing environment.

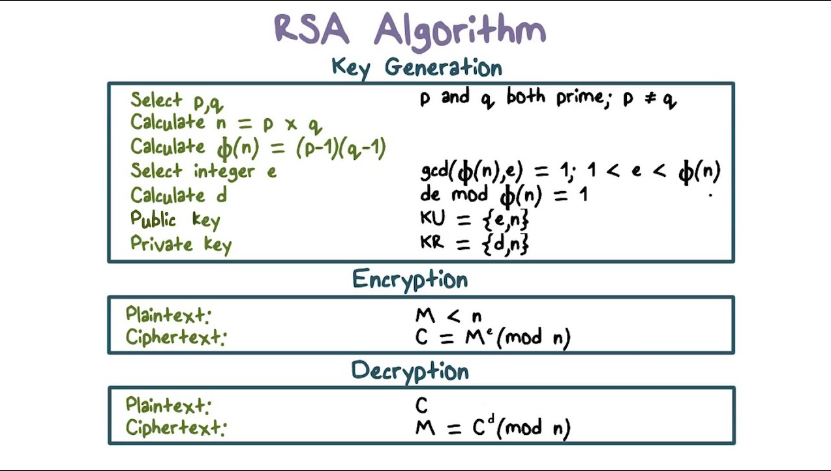


These are the packages successfully installed which are necessary for the system implementation to yield essential outcome.

Now, we import necessary modules to run functions. The modules thus imported are:

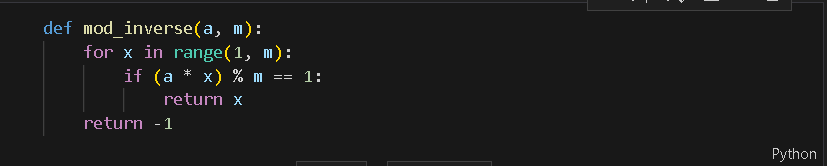


Making RSA algorithm:



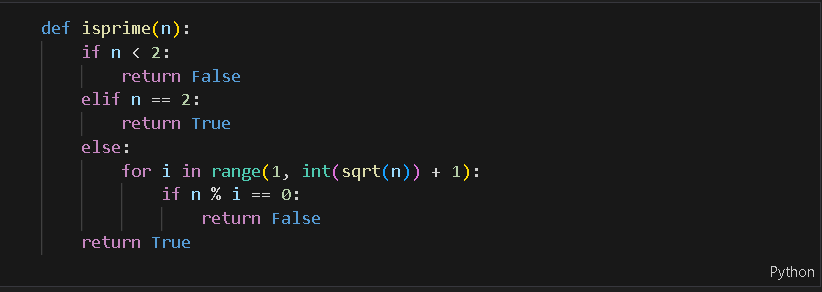
Making of RSA algorithm as depicted in the figure above comprises of three parts in this system implementation. First, we generate the keys i.e. public key and private key as RSA is an asymmetric algorithm. Doing so, we encrypt a plain text using the private key into a cipher text. We then find the value of the private key using Shor’s algorithm and then, decrypt the cipher text into the plain text using the private key.

We calculate the modular inverse first to initialize this process which is:



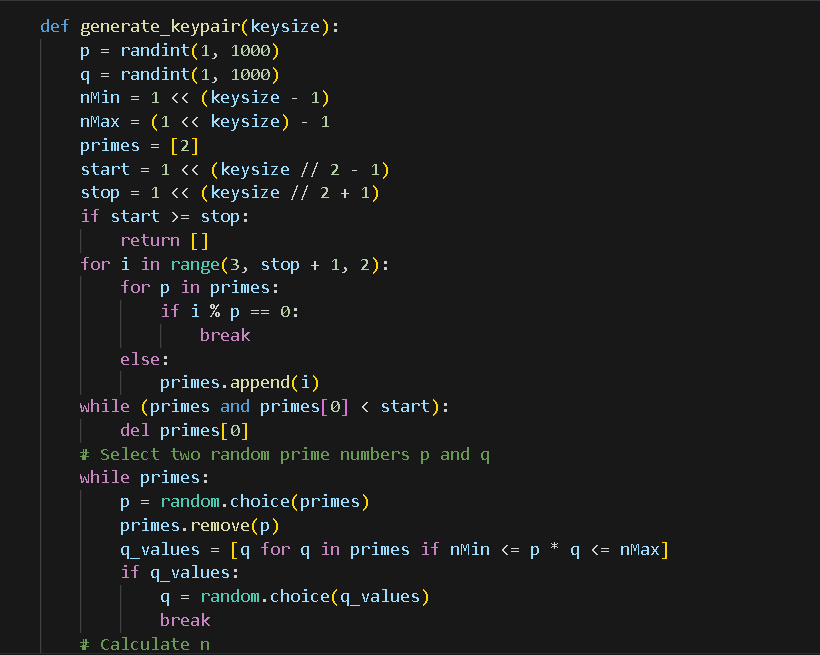
The modular inverse is calculated at the start of the project as it needs no definitive value, and this function can be called upon later to be used when modular inverse is needed to find the value of d.

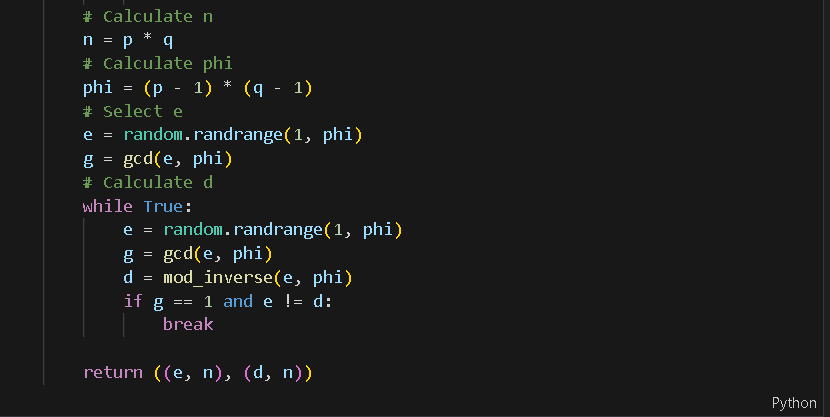
Now, we check the primality of the value n. the factorization only works when n comprises of primary numbers which is later essential for this system to calculate the value of p and q which are both prime numbers.



This returns the value of n to be true or false. In any case the value rounds up until true value is returned thus making sure that n is a prime number.

After that we move to **key generation**:

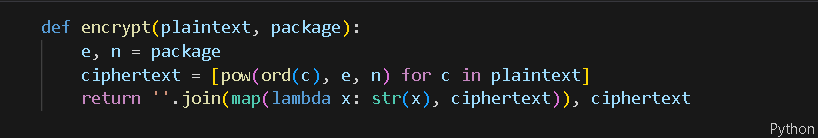




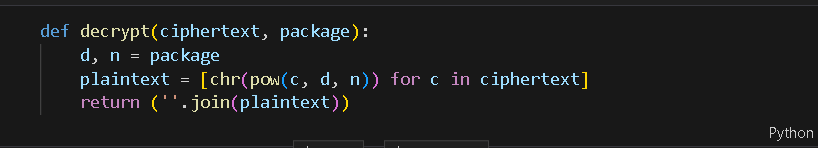
Key generation refers to the process of generating public key (e,n) and private key (d,n). Two random prime numbers from 1 to 1000 of similar bit size is chosen. Its GCD is calculated and if its similar then the exponent e and private key d is calculated form the random number n such that n = p \* q and p ≠ q.

After generating the value of private and public key we can now encrypt or decrypt a plain text into cipher text and vice versa using the code below.

**For encryption**:



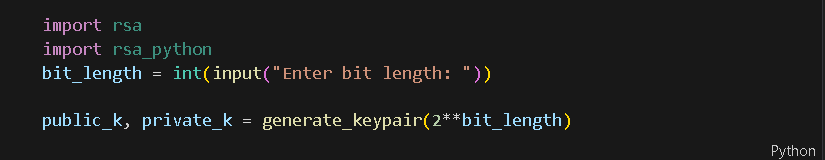
**For decryption**:



**Testing the encryption and decryption functions**:

Thus, after completion of the above implemented steps, we can now generate keys, encrypt and decrypt using few lines of codes which are:

Generate keypair:



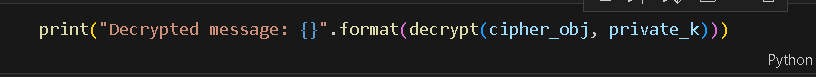
Public key (Public\_k) and private key (private\_k) is generated into the value generate\_keypair.

Encryption:



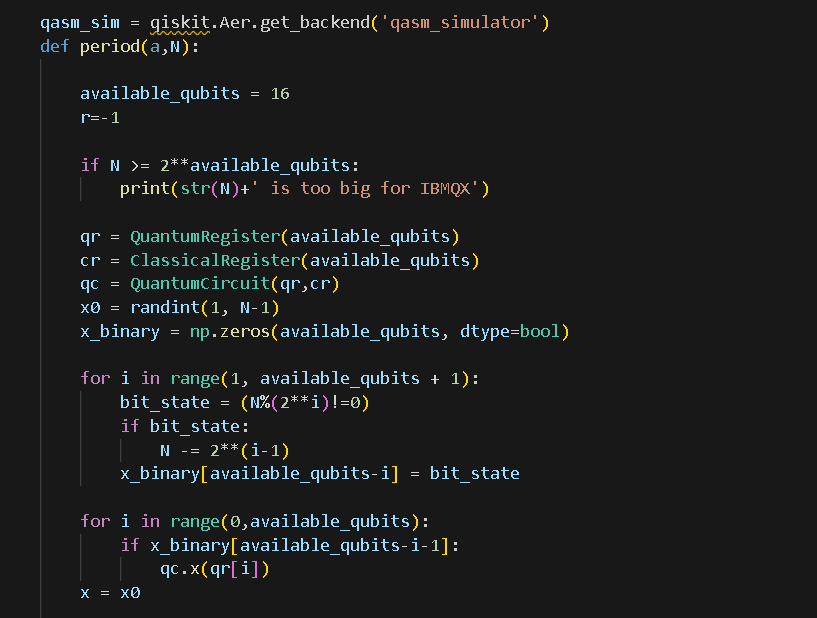
Plain text (plain\_txt) is entered as an input and encrypted into a cipher text (cipher\_txt) and printed out.

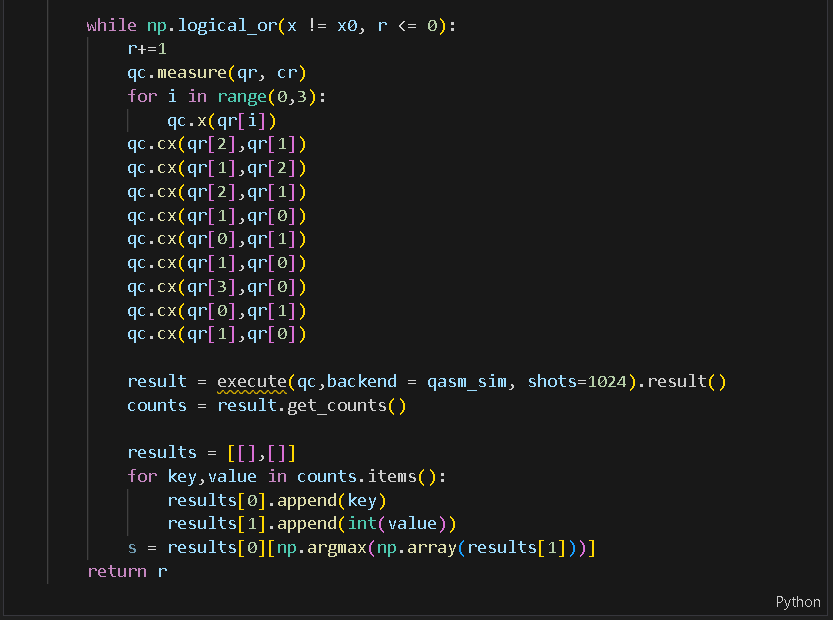
Decryption:



Can now print the decrypted message when private key (private\_k) is known. The message cannot be decrypted else.

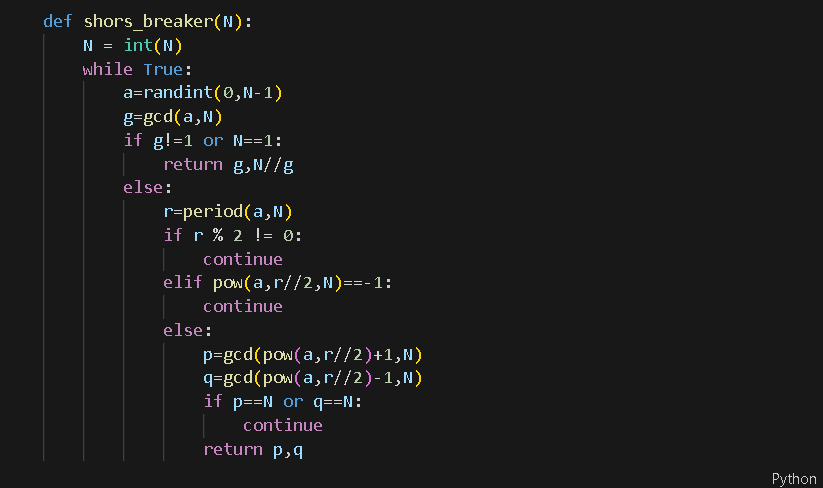
At this point in system integration, we will start **framing Shor’s algorithm**.





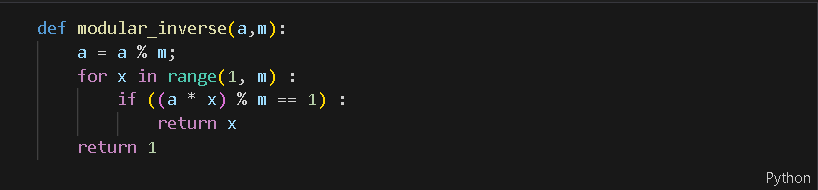
Total of 16 qubits are made available from which QuantumRegister (qr), ClassicalRegister (cr) and QuantumCircuit (qc) is operated. The above operation returns the value of r.

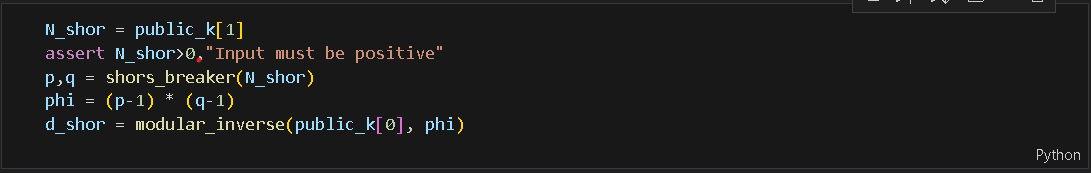
We now implement the code to find the value of p and q from the given N and r. we apply the shors\_breaker() as:



This string of code calculates the gcd to return p and q. the random integer a is used as a ceiling function to round up the value to nearest integer through subtraction of N from 1 i.e. (N-1).

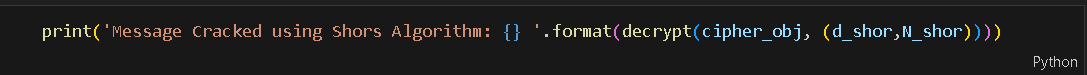
**Modular inverse** is calculated:





The modular inverse gives us the value of d\_shor, shor’s breaker gives is the value of p and q, N\_shor is the public key made available for all and phi is calculated from p and q.

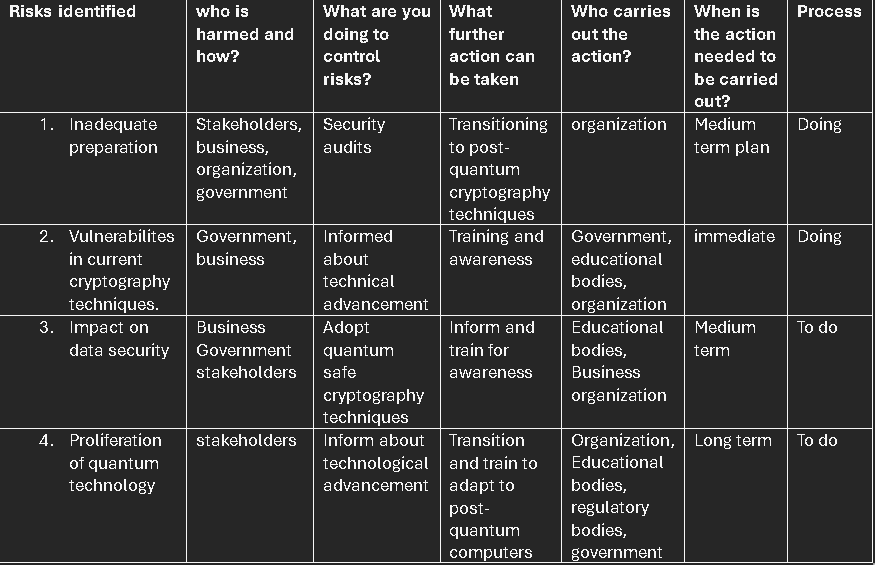
Finally, the cipher text can now be cracked and **deciphered**:



### Derive risk associated with modern day cryptography from quantum computers.

The technical artefact to refer to for this section is **PG15-Risk-Assessment-2.1.docx**. through this document we derive various risks with modern day cryptography from quantum computers amongst which the primary risk mitigating factor is transitioning to post-quantum cryptography.

The risk assessment table derived is depicted below:



### 2.2 Provide the need for transition to post-quantum cryptographic techniques

The need for transition is primarily realised through implementation of **Shor’s algorithm** as it gives significance to the time frame needed for transitioning to post-quantum cryptography. It signifies that although discovered almost 30 years ago, post-quantum era is inevitable and might pose limited timeframe to organizations which poses threat to the security and operational efficiency of the data and standards respectively. This is represented on the technical artefact **PG15-Need-for-transition-2.2.docx**.

# Delivered Technical Artefacts

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **File** | **Description** | **PDF?** |
| Comparison between present day computational ability to post-quantum computational ability for cryptography | PG15-Comparision-of-abilities-1.1.docx | Detailed outlining of post-Quantum computational ability, comparison from classical computers to post-quantum computers, outlining computational limit of classical computers, assessing vulnerabilities of cryptographic techniques due to the limit. | Yes |
| Listing present day algorithms and assessing vulnerabilities | PG15-Algorithm-vulnerabilities-1.2.docx | Due to reliability on calculational ability for cryptography, symmetric and asymmetric algorithms are listed and categorized upon vulnerability due to calculational ability of quantum computers. | Yes |
| Tools and methodology used to implement the system | PG15-System-Implementation-Tools.docx | Use of tools such as sagemath, python and qiskit, visual studio code, GitHub, MS teams etc. | Yes |
| Fermat’s Factorization algorithm using sagemath justification and algorithm | PG15-Fermat’s-Facto-Algo-1.4.docx | Public key (e,n) is identified or assumed, and value of private key (d) is factorized using Euler’s totient function and ceiling loop from public key to break RSA. | Yes |
| Shor’s Algorithm is framed to use qubits from qiskit in python on visual studio code for factorization of prime numbers to encrypt and decrypt messages. | PG15-Shor’s\_Algo2.ipynb | First key pair is generated, and modular inverse is calculated. Using packages encryption and decryption is made possible. Shor’s algorithm is framed and qiskit is used to generate a quantum computing ability through which both encryption and decryption is made possible. | No |
| Risk assessment of cryptography with quantum computers. | PG15-Risk-Assessment-2.1.docx | Risk identification, risk analysis and risk mitigation for post-quantum era. | Yes |
| Need for transition outlines the causes that are triggered from implementation of Shor’s algorithm that signifies the inevitability of post-quantum era. | PG15-Need-for-transition-2.2.docx. | Document explains the need for transition to quantum cryptography from modern cryptography and justifies various reasons. | Yes |
| Progression of the kanban board | PG15-Progression-Of-Kanban.docx | Documents the progression of tasks according to the allocation from week 1 to week 13. | Yes |
| Gantt chart | PG15-Gantt-chart.docx | Gantt chart shows the progression of tasks allocated from start to the end. | Yes |
| Proposing Transition | PG15-Propose-Transition.docx | Proposing the system implementation of post-quantum cryptography algorithm Kyber. | Yes |

# Contributions

|  |  |  |  |
| --- | --- | --- | --- |
| **Student Name** | **Percent** | **Summary of Contributions** | **Technical Lead on Artefacts** |
| Ayush Keshar Prasai | 25% | * Researched and implemented Tools and Methodology needed for system implementation for this project. * Researched Fermat’s Factorization Algorithm and its implementation and outcome. * Researched Shor’s Algorithm and its deliverables. | * PG15-System-Implementation-Tools.docx * PG15-Fermat’s-Facto-Algo-1.4.docx * PG15-Shor’s\_Algo2.ipynb |
| Jalay Shah | 25% | * List the algorithms and assesses vulnerabilities in terms of computational ability of quantum computers. * Assesses risks associated with use of widely used cryptography along with risk assessment table | * PG15-Algorithm-vulnerabilities-1.2.docx * PG15-Risk-Assessment-2.1.docx |
| Ronit Maheshwari | 25% | * Compares computational abilities between era to point the hardware superiority of post quantum era. * Depicts the progression of tasks on the kanban board. | * PG15-Comparision-of-abilities-1.1.docx * PG15-Progression-Of-Kanban.docx |
| Viraj Sinh Rahevar | 25% | * Created the Gantt chart * Proposed a transition with use of post quantum cryptography algorithm * Researched kyber as post-quant cryptography algorithm and proposed transition to quantum cryptography. | * PG15-Gantt-chart.docx * PG15-Propose-Transition.docx * Need for transition |

# Next Steps

### 2.3 propose transition

The next step to take would be to propose a transition with the use of post-quantum cryptography algorithm as referred to from **PG15-Propose-Transition.docx**. This document proposes such transition by:

* Introducing Kyber.
* Justifying its design and functionality.
* Evaluating Kyber’s security.
* Imagining its challenges and future implementation.