Implementación del protocolo criptográfico Six-State

Andrea Hernández-Martín
Universidad de La Laguna
Tenerife, Spain
andreahdezm99@gmail.com

Pino Caballero-Gil
Universidad de La Laguna
Tenerife, Spain
pcaballe@ull.edu.es

Daniel Escanez-Exposito
Universidad de La Laguna
Tenerife, Spain
jdanielescanez@gmail.com

***Resumen*—Se presenta en este trabajo una breve descripción de la distribución de claves cuánticas mediante el protocolo *Six-State*, así como de los principales detalles de la implementación realizada de ese protocolo. Dicha implementación ha sido objeto de ampliación de la librería de código abierto *QuantumSolver*, cuyo objetivo es facilitar el desarrollo cuántico, permitiendo la ejecución de una colección creciente de algoritmos y protocolos criptográficos cuánticos. Además, aquí se hace especial hincapié en el análisis de los resultados obtenidos con la implementación del protocolo.**

***Index Terms*—Computación cuántica, Protocolo Six-State, Criptografía cuántica, Qiskit**

Tipo de contribución: *Investigación en desarrollo.*

I. Introducción

La computación y criptografía cuánticas son temas que recientemente han ido ganando importancia debido a su rápido avance apoyado por grandes empresas tecnológicas como IBM [1], Google [2] o Amazon [3], que están invirtiendo en el desarrollo e investigación de esta tecnología.

En 2022, IBM presentó su último procesador cuántico, Osprey [4], que cuenta con 433 cúbits, lo que representa un gran adelanto en la capacidad de procesamiento cuántico. Este computador es parte de la próxima generación del sistema *IBM Quantum System Two*, que está diseñado para ofrecer mayor rendimiento y capacidades avanzadas de computación cuántica. Desde 2019 habían publicado una hoja de ruta de desarrollo [5] en la que se incluía el lanzamiento de nuevos servicios y herramientas de software de computación cuántica. Hasta el momento, se han cumplido los objetivos establecidos en dicha hoja, enfocados hacia nuevas formas de integrar la computación cuántica con la informática clásica para lograr el máximo aprovechamiento de la tecnología. Esa planificación predice la construcción de sistemas cuánticos de próxima generación con más de 1000 cúbits para abordar problemas cada vez más complejos.

Recientemente, la revista científica *Nature* ha publicado el segundo gran hito de Google en computación cuántica [6], lo que representa un importante paso hacia la creación de esa máquina cuántica capaz de realizar cálculos complejos de manera más rápida que cualquier supercomputadora actual. Concretamente el logro de Google se basa en el concepto de ”supremacía cuántica”, anunciado por Google desde 2019 [7]. Ese concepto se refiere al punto en el que una máquina cuántica puede realizar una tarea que sería prácticamente imposible de realizar por una computadora clásica. Un ejemplo es la tarea para generar un patrón aleatorio de números y verificar si este patrón es realmente aleatorio.

Todos esos avances evidencian el creciente interés en las tecnologías cuánticas. Por ese motivo, con el objetivo de fomentar su uso y estudio por usuarios con o sin experiencia en informática nació la propuesta *QuantumSolver* [8]. Con esa meta en mente, se potenciaron dos características de la herramienta, referentes a los modos de acceso al software desarrollado. Por un lado, ofrece una interfaz web en la que se pueden ejecutar los algoritmos disponibles predefinidos obteniendo los resultados de manera visual e intuitiva. Por otro lado, tiene una interfaz por línea de comandos, que está más orientada a usuarios con algún conocimiento informático.

QuantumSolver ha ido enriqueciéndose con la implementación de varios algoritmos cuánticos como generación de números aleatorios, Deutsch-Jozsa , Bernstein-Vazirani , Grover , teleportación cuántica y codificación superdensa ; y varios protocolos criptográficos como BB84 , E91 , B92 , versiones cuánticas de ElGamal y RSA. Todos esos esquemas se pueden ejecutar en simuladores o computadores cuánticos de IBM.

En el presente trabajo se ha realizado un esfuerzo para mejorar el diseño de implementación de la herramienta, que inicialmente no diferenciaba entre algoritmos y protocolos. Concretamente, se ha realizado una mejora y ampliación de la librería, separando los esquemas en dos módulos diferentes: *QuantumSolver Basic* para los algoritmos sencillos que se pueden representar con un circuito parametrizado y *QuantumSolver Crypto* para los protocolos de criptografía cuántica. Además se ha decidido añadir un nuevo esquema criptográfico a este último módulo, mediante la implementación de un protocolo de distribución de claves cuánticas (QKD, Quantum Key Distribution) llamado *Six-State*.

El presente trabajo se estructura de la forma siguiente. La sección II incluye los principales detalles de la propuesta. La sección III aportan el fundamento teórico mientras que las secciones IV y V se centran en la implementación y el programa. La sección VI contiene una análisis de resultados. Finalmente el trabajo se cierra con la sección VII de conclusiones.

II. Mejora de la librería QuantumSolver

La librería cuántica *QuantumSolver* se ha implementado en el lenguaje de programación Python3 gracias a *Qiskit*, que es un Kit de Desarrollo Software (SDK, Software Development Kit) que proporciona IBM para trabajar con ordenadores cuánticos a nivel de circuitos, pulsos y algoritmos [9]. Para la ejecución de la librería por Línea de Comandos (CLI, Command-Line Interface), *QuantumSolver* cuenta con dos

ISBN:978-84-8158-970-2

programas principales: uno para el módulo *QuantumSolver Basic* y otro para *QuantumSolver Crypto*.

II-A. QuantumSolver Basic

En el módulo *QuantumSolver Basic* se encuentran todos los algoritmos cuánticos implementados hasta el momento en la librería. Cada algoritmo se ha implementado dentro del módulo básico de manera separada en submódulos. Para la ejecución de dichos algoritmos se encuentra disponible el programa principal del módulo básico (*'main_quantum_solver.py'*). En primer lugar se da la opción de o bien utilizar un *token* de API (Application Programming Interface) de *"IBM Quantum Experience"* [10], o bien entrar como usuario invitado en caso de no poseer un *token*. A continuación, se muestra un menú que permite elegir el *backend* que se va a utilizar para la simulación, que en este caso solo permite hacer uso de *'aer_simulator'*. Una vez elegido el *backend*, se pueden elegir el algoritmo cuántico que se quiere simular y los parámetros para la ejecución. Por último, se da la opción de simular el algoritmo directamente mostrando el resultado y el circuito que se genera, o también se puede simular de forma experimental, observando el comportamiento del algoritmo ejecutado varias veces mediante un histograma.

II-B. QuantumSolver Crypto

El módulo *QuantumSolver Crypto* cuenta con todos los protocolos de criptografía cuántica que se han implementado en la librería. Para su ejecución se encuentra el programa principal dedicado a la criptografía (*main_cryto.py*), que de manera muy similar al programa principal básico, permite elegir uno de los protocolos y el *backend* a utilizar. Además, el modo experimental está implementado de forma que además de mostrar el resultado y un mapa de calor del protocolo, también muestra su traza de ejecución.

III. Fundamento teórico del protocolo Six-State

hEl protocolo criptográfico *Six-State* es un método utilizado en la computación cuántica para la transmisión segura de información entre dos usuarios [11]. De todos los estados posibles de un cúbit, este protocolo utiliza los estados que están proyectados sobre los ejes z, x e y, es decir $\{|0\rangle, |1\rangle\}$, $\{|+\rangle, |-\rangle\}$ y $\{|i\rangle, |-i\rangle\}$, respectivamente.

En este algoritmo dos participantes, Alice y Bob, desean comunicarse de manera segura utilizando un canal de comunicación cuántico unidireccional y un canal clásico bidireccional, ambos públicos. Primero, Alice genera una cadena aleatoria binaria para enviársela a Bob, utilizando los estados básicos $\{|0\rangle, |1\rangle\}$. Después, Alice elige aleatoriamente, para cada cúbit, una de las tres bases de medición (eje z, eje x, eje y), con una probabilidad de $\frac{1}{3}$ para cada una. En el caso de usar el eje z no se aplican transformaciones al cúbit en cuestión. Sin embargo, si se elige el eje x se le aplica la puerta *Hadamard*, también llamada puerta H, cuya representación matricial se puede observar en la Ec. (1).

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{1}$$

Por último, si usa el eje y se le aplica una puerta creada a partir de la puerta Z y la puerta Y, cuyas representaciones matriciales se pueden observar en la Ec. (2) y (3) respectivamente. Además, la puerta creada a partir de ambas se muestra en la Ec. (4).

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{2}$$

$$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \tag{3}$$

$$H_Y = \frac{1}{\sqrt{2}}(Y + Z) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -i \\ i & -1 \end{pmatrix} \tag{4}$$

Después de que Alice envíe estos cúbits aleatorios preparados en las bases aleatorias, Bob realiza una medición en cada cúbit que recibe. Dicha medición se realiza de manera aleatoria entre los ejes disponibles (z, x, y). Aproximadamente $\frac{1}{3}$ de los cúbits serán medidos en el mismo eje que el emisor, por lo que se consideran correctos. Los restantes $\frac{2}{3}$, deberán ser descartados dado que al medirlos en el eje equivocado, se tendrá exactamente un 50 % de probabilidad de emitir el valor codificado correcto, lo que implica la perdida total de la correspondiente información. Para realizar este descarte se hacen públicos los ejes en los que se prepararon y midieron los cúbits, ya que no hay riesgo al realizar tal acción después de haber realizado las mediciones, y se eliminan los que no están medidos en el mismo eje por Alice y Bob. Los valores restantes tras el descarte podrían ser considerados la clave generada, pero antes hay que comprobar su seguridad. Si existen resultados incoherentes entre la clave del emisor y la del receptor, se evidencia la presencia de un atacante. Por ello, Bob debe compartir públicamente una parte de la clave recibida, para que Alice se asegure de que coincide con lo que ella tiene. Si es así, Alice comunica a Bob que puede utilizar el resto de la clave como secreto compartido seguro. En caso contrario, se deduce que se han interceptado los cúbits enviados y que han sido medidos antes de remitirlos al receptor, por alguna entidad intermedia que ha lanzado un ataque de escucha secreta *(eavesdropping)*, y por tanto toda la clave es descartada y se reinicia todo el proceso.

En la *Tabla I* se observan todos los posibles casos, distribuidos en tres agrupaciones de comportamiento similar, en las que se presentan: casos en los que el protocolo es abortado por detectar una comunicación comprometida, y casos en los que no. Concretamente se tienen nueve casos distintos, cada uno con una probabilidad de suceder de $\frac{1}{9}$.

En este protocolo se gana en seguridad con respecto a sus predecesores debido a que las tres bases abarcan toda la *Esfera de Bloch* mostrada en la Fig. 1 con los 6 estados que utiliza el protocolo. Por ejemplo, en el caso del protocolo *BB84* solo se utiliza un plano bidimensional.

IV. Implementación del protocolo Six-State

El desarrollo del protocolo *Six-State* se ha realizado dentro del módulo *QuantumSolver Crypto*. Para implementarlo se han diseñado tres entidades: *Participant*, *Sender* y *Receiver*; siendo la primera la entidad principal base y las demás sus entidades derivadas. La finalidad es que una instancia de *Sender* pueda comunicarse con una instancia de *Receiver* haciendo que la comunicación sea privada gracias al protocolo. Se comprueba que la comunicación es privada gracias a una

Tabla I
CASOS POSIBLES DE LAS MEDICIONES DE UN CÚBIT EN LA FASE DE VERIFICACIÓN DE SIX-STATE

Medición de emisor y receptor legítimo	Medición del atacante intermedio	Conclusión sobre ese cúbit	Probabilidad de suceder
Mismo eje	Mismo eje que ambos	Ha sido interceptado por el atacante sin que se aborte el protocolo, con un 100 % de probabilidad de que el valor final sea correcto	Sucede en $\frac{1}{9}$ casos
Mismo eje	Distinto eje que ambos	En el 50 % de estos casos se abortará el protocolo al detectar la interceptación	Sucede en $\frac{2}{9}$ casos
Distinto eje	Cualquier eje	Es desechado en la fase de descarte de los valores, sin tener en cuenta al atacante	Sucede en $\frac{6}{9}$ casos

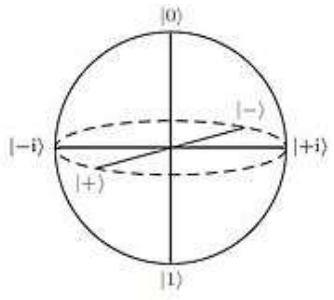

Figura 1. Esfera de Bloch con 3 bases

libreta de un solo uso que se genera en cada comunicación y que además solo la comparten el emisor y el receptor.

La clase principal *Participant* contiene los métodos para la generación y muestra de valores, ejes, claves y libretas de un solo uso. También tiene un método encargado de crear una puerta cuántica capaz de medir o preparar los cúbits enviados respecto al eje y, cuya implementación se puede observar en el Código (1). Esta puerta se ha implementado gracias a las funcionalidades de Qiskit, utilizando la clase *Operator*, para representar los operadores matriciales que actúan sobre el sistema cuántico. Se ha utilizado obteniendo el valor la suma de las puertas cuánticas Y y Z, multiplicada por el factor $\frac{1}{\sqrt{2}}$ (véase Ec. (4)). Para la implementación se crea un circuito cuántico y se utiliza el método *unitary* para añadir la puerta necesaria a partir de la matriz calculada. Por último, se convierte el circuito en una puerta mediante el método *to_gate*.

Código 1. Creación de la puerta Hy

```
def set_hy(self):
    hy_op = qi.Operator(1/np.sqrt(2)*
        (YGate().to_matrix()+
        ZGate().to_matrix()))
    hy_gate = QuantumCircuit(1)
    hy_gate.unitary(hy_op,[0],label='h_y')
    self.hy = hy_gate.to_gate()
```

Las clases *Sender* y *Receiver* son bastante similares. La primera tiene un método para enviar un mensaje, el cual inicializa el circuito cuántico; y la segunda tiene otro para recibirlo añadiendo la fase de medición al circuito. Además hay que tener en cuenta que ambas clases para la preparación o medición de los cúbits, usan la puerta cuántica vinculada al eje y explicada anteriormente en el código (1).

Todas las simulaciones realizadas en el canal cuántico se realizan mediante la utilización de circuitos cuánticos de *Qiskit*. La implementación del protocolo de manera más detallada se puede encontrar en sus respectivos ficheros de código fuente [12].

V. INTEGRACIÓN DEL PROTOCOLO EN LA LIBRERÍA

Al ejecutar el programa principal de *QuantumSolver Crypto*, se muestra un menú que permite elegir *Six-State* como protocolo a utilizar. A continuación, se da la opción de utilizar el token de IBM o entrar como usuario invitado, así como de elegir el *backend* a utilizar. Una vez elegido, se puede simular el protocolo de dos maneras.

La primera forma permite ejecutar el algoritmo una única vez con los parámetros necesarios: una cadena de caracteres como mensaje y un valor entre 0 y 1 como densidad de interceptación, que es la probabilidad con la que el receptor intermedio ilegítimo medirá cada cúbit. Una vez establecidos los parámetros que se utilizarán en la simulación, el programa muestra una traza con los resultados obtenidos. En esta traza se muestran los ejes de preparación o medición de Alice, Bob y Eva (receptor ilegítimo) con valores aleatorios entre 0 y 2, ya que tienen tres ejes para elegir. También se pueden observar los valores preparados o medidos de 0 y 1, y en el caso de Eva toman -1 cuando no se haya realizado esa medición. Además, se muestra la clave privada de Alice y Bob, y la clave que comparte Bob para verificar la seguridad de la misma. Por último, se observa un mensaje de error en caso de que se detectase la interceptación del mensaje, abortando el protocolo; y en el caso contrario, se observa un aviso de que la comunicación fue segura.

La segunda manera de ejecutar el programa es mediante el modo experimental, en el que el algoritmo se ejecuta varias veces, mostrando un mapa de calor que representa mediante colores más claros las condiciones en las que ha habido más ocasiones de comunicación considerada segura. Con colores más oscuros se muestran los casos donde ha habido comunicaciones en las que se han detectado más interceptaciones. Los parámetros a especificar son distintos a los del modo anterior. En este caso hay que establecer: la longitud máxima del mensaje en número de bits como número positivo múltiplo de 5 hasta ese máximo, conformando el eje x en el mapa de calor; la magnitud de la distancia entre casos de la densidad de interceptación, que toma valores entre 0 y 1, y que define el eje y en el mapa de calor; y el número de repeticiones para cada instancia. La traza que se obtiene del modo experimental es diferente a la anterior. En este caso se observan los valores que van tomando los diferentes parámetros durante la ejecución, así como el tiempo estimado de ejecución de las instancias restantes del protocolo.

VI. RESULTADOS

Aprovechando que ambos están implementados en la librería *QuantumSolver* se ha realizado un análisis de resultados

que permite demostrar la mejora que representa el protocolo *Six-State* en comparación con el protocolo *BB84*.

En las Fig. 2 y 3 se muestran dos ejemplos de mapas de calor generados. Por un lado, es importante destacar el tiempo de ejecución de ambos protocolos. Mientras que el *Six-State* tardó aproximadamente 22 horas en finalizar su ejecución para generar el mapa de calor, el *BB84* tardó 22 horas aproximadamente en generar el mapa de calor con los mismos parámetros que el anterior. Esto es debido a la nueva puerta que se tuvo que crear para poder medir en el eje y, ya que esta puerta no está definida dentro de las instrucciones primarias del simulador cuántico que se está utilizado.

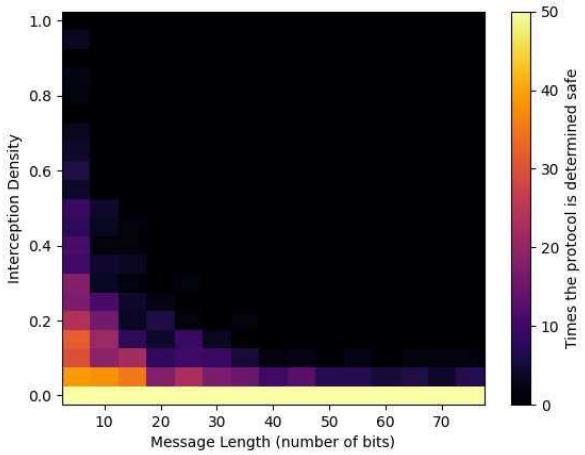

Figura 2. Ejemplo de mapa de calor del protocolo *Six-State*

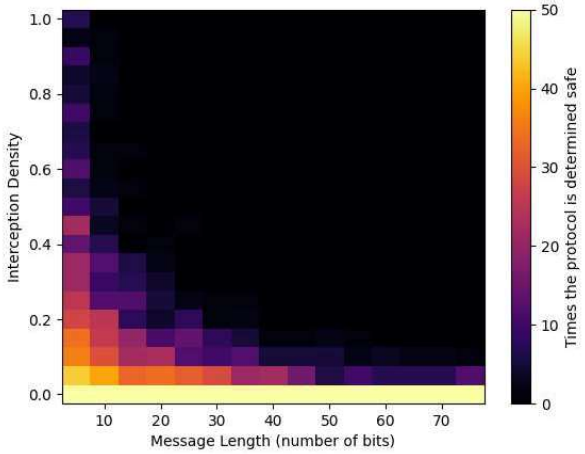

Figura 3. Ejemplo de mapa de calor del protocolo *BB84*

Si se observan ambas imágenes se puede ver como la curva del *Six-State* es ligeramente mejor que la del *BB84*, ya que esta última es ligeramente más alta, por lo que se consideran como buenos casos que no lo son, es decir, se está detectando menos al interceptor. Esto es debido a que el protocolo *Six-State* utiliza tres estados cuánticos no ortogonales que permiten obtener información adicional sobre el estado enviado [13]. Además hace que sea más difícil descifrar el mensaje para el interceptor ya que tiene tres ejes en los que medir, a diferencia del *BB84* que tiene dos. Por tanto, la probabilidad de que Eva mida en el mismo eje que Alice y Bob es menor.

Por otro lado, se debe tener en cuenta que cuanto más claro es el color, en más ocasiones se determina segura la comunicación, y cuanto más oscuro, más insegura. Al observar los gráficos, se considera que cuánto más crece el número de bits del mensaje a enviar se tiene que la probabilidad de detectar al interceptor es mayor, ya que como se puede observar, el gráfico es todo prácticamente oscuro según crece la variable del eje x.

Sin embargo, se ha comprobado el comportamiento en caso de que el número de bits sea pequeño. En ese caso el protocolo *Six-State* es más seguro ya que cuando se tienen pocos bits y la densidad de interceptación es mayor, la curva es más pequeña y los colores son más oscuros. Esto es debido a que se están desechando los casos en los que se intercepta el mensaje, a diferencia del *BB84* que cuando hay casos en los que con un número muy pequeño de bits y la densidad de interceptación es cercana a 1, la comunicación se da como segura cuando en realidad se está interceptando el mensaje prácticamente completo.

Cabe destacar el caso en el que la densidad de interceptación es cero para todos los tamaños del número de bits, observándose en el gráfico como el protocolo se considera seguro en el 100 % de los casos, desde que el protocolo se ha ejecutado en un entorno sin ruido.

Finalmente, se puede llegar a la conclusión de que el protocolo *Six-State* es más seguro que el protocolo *BB84* debido a su capacidad para detectar a los interceptores.

VII. Conclusiones

La librería *QuantumSolver* es una herramienta muy útil ya que permite la ejecución de software cuántico de manera amigable para todo tipo de público interesado. Gracias a ella el aprendizaje sobre criptografía y computación cuánticas resulta más sencillo por la posibilidad que ofrece de simular algoritmos y protocolos cuánticos, obteniendo resultados visuales. Además, permite manejar el correspondiente código software, lo que facilita que esté en constante evolución, pudiéndose prever que en un futuro próximo contenga muchos más algoritmos y protocolos cuánticos.

En este trabajo se ha presentado una propuesta de mejora de la estructura de la librería *QuantumSolver* que facilita las futuras contribuciones a la herramienta. Además se ha descrito la implementación de un nuevo protocolo añadido a la librería: el protocolo *Six-State* para distribución de claves cuánticas. Este protocolo es un método criptográfico cuántico que se espera que en el futuro sea utilizado en aplicaciones prácticas de ciberseguridad dado que, como se ha podido comprobar en los resultados de la implementación realizada, presenta una mayor fiabilidad que el conocido protocolo *BB84* pues mejora la detección de interceptores. Como trabajo futuro se pretende ampliar los estudios sobre este protocolo para intentar obtener resultados más relevantes.

Agradecimientos

Esta investigación ha sido posible gracias a la Cátedra de Ciberseguridad de la Universidad de La Laguna.

REFERENCIAS

[1] IBM Research, “Quantum Computing - IBM Research”. [Online]. Available: https:research.ibm.com/quantum-computing. [Accessed: 27-Mar-2027].

[2] Google LLC, “Google Quantum AI”. [Online]. Available: https://quantumai.google/. [Accessed: 27-Mar-2023].

[3] Amazon Inc., “Amazon Braket”. [Online]. Available: https://aws.amazon.com/es/braket/. [Accessed: 27-Mar-2023].

[4] IBM España Newsroom, “IBM presenta su procesador cuántico de más de 400 qubits y la próxima generación de IBM Quantum System Two”. [Online]. Available: https://es.newsroom.ibm.com/122752. [Accessed: 27-Mar-2023].

[5] Gambetta, J., .Expanding the IBM Quantum roadmap to anticipate the future of quantum-centric supercomputing”. IBM Research Blog, 2022.

[6] Castelvecchi, D., “Google’s quantum computer hits key milestone by reducing errors”. Nature, 2023.

[7] Gibney, E., “Hello quantum world! Google publishes landmark quantum supremacy claim”, Nature, vol. 574, no. 7779, pp. 461-463, 2019.

[8] Escanez-Exposito, D., “QuantumSolver”. [Online]. Available: https://github.com/jdanielescanez/quantum-solver. [Accessed: 15-Mar-2023].

[9] IBM, “Qiskit”. [Online]. Available: https://qiskit.org/. [Accessed: 16-Mar-2023].

[10] IBM, “IBM Quantum”. [Online]. Available: https://quantum-computing.ibm.com/. [Accessed: 17-Mar-2023].

[11] Bruß, D., “Optimal eavesdropping in quantum cryptography with six states”. Physical Review Letters, 81(14), 3018, 1998.

[12] Hernández, A., Escanez-Exposito, D., “Six-State Implementation with QuantumSolver”. [Online]. Available: https://github.com/jdanielescanez/quantum-solver/tree/main/src/crypto/six_state. [Accessed: 17-Mar-2023].

[13] Senekane, M., Mafu, M., Petruccione, F., “Six-State Symmetric Quantum Key Distribution Protocol”. Journal of Quantum Information Science, 5(02), 33-40, 2015.

[14] Fung, C.H.F., Qi, B., Tamaki, K., Lo, H.K., “Phase-Remapping Attack in Practical Quantum Key Distribution Systems”. Physical Review A, 75(3), 032314, 2007.