

Evaluación de una Fuente de Aleatoriedad Cuántica como Generador de Entropía para Sistemas Criptográficos

Profesora: Marcela Rosalba Samela
Autor: Gastón Martín Tedesco

Trabajo Final de Carrera presentado para obtener el título de Licenciatura en Gestión de Tecnología informática

(Julio, 2025)

Índice de Tablas

Tabla 1: Comparación de Generadores de Aleatoriedad 30
Tabla 2: Muestra generada por ANU QRNG 43
Tabla 3: Comparación de Resultados 46
Tabla 4: Muestra generada por la función random.getrandbits(1) 49

Índice General

Table of Contents

Índice de Tablas 2
Índice General 3
1. Resumen 6
2. Capítulo 1: Introducción 7
2.1. Identificación del Problema y Antecedentes 7
2.2. Planteamiento 8
2.3. Objetivo general 12
2.3.1. Objetivos particulares 12
2.4. Pregunta de investigación 13
2.5. Hipótesis 13
3. Capítulo 2: Estado del arte 13
3.1. La Aleatoriedad como Insumo Critico en Criptografia 13
3.2. Limitaciones de los Generadores Algorítmicos Convencionales 15
3.3. Principios y Evolución de los Generadores Cuánticos de Números Aleatorios (QRNG) 16
3.4. Plataformas actuales de acceso a QRNG (incluyendo ANU QRNG) 18
3.5. Comparación entre QRNG y Fuentes Algorítmicas: Estado Actual de la Investigación 20
3.6. Aplicabilidad de la Aleatoriedad Cuántica en Desarrollo de Software 21
3.7. Síntesis y Vacío de Investigación 23
4. Capítulo 3: Marco teórico 25
4.1. Aleatoriedad y su Rol en la Criptografía: Fundamentos y Propiedades Esenciales 25
4.2. Entropía y Calidad Aleatoria 27
4.3. Clasificación de Generadores de Aleatoriedad 28
4.3.1. Generadores Pseudoaleatorios (PRNG) 28
4.3.2. Generadores Pseudoaleatorios Criptográficamente Seguros (CSPRNG) 29
4.3.3. Generadores Verdaderamente Aleatorios (TRNG) 29
4.3.4. Generadores Cuánticos de Números Aleatorios (QRNG) 29
4.3.5. Comparación de Generadores de Aleatoriedad Existentes 30
4.4. Evaluación Estadística de la Aleatoriedad 31
4.4.1. Pruebas de Frecuencia (Monobit) 31
4.4.2. Prueba de Rachas (Runs Test) 31

4.4.3. Entropía de Shannon 32
4.4.4. Prueba de Suma Acumulada (Cumulative Sums) 32
4.5. Aplicaciones de la Aleatoriedad en Seguridad Informática 32
4.5.1. Generación de Claves Criptográficas 33
4.5.2. Generación de Nonces y Vectores de Inicialización 33
4.5.3. Protocolos Criptográficos y Autenticación 33
4.5.4. Inicialización Segura de Sistemas 33
4.5.5. Aplicaciones Emergentes 34
5. Capítulo 4: Metodología de Investigación 35
5.1. Tipo Y Enfoque de Investigación 35
5.2. Técnicas e Instrumentos Utilizados 35
5.3. Criterios de Análisis y Unidades de Observación 37
5.4. Procedimiento y Etapas 38
5.4.1. Delimitación del Objeto de Estudio 38
5.4.2. Obtención de las secuencias Binarias 39
5.4.3. Procesamiento y Preparación de Datos 39
5.4.4. Aplicación de Pruebas Estadísticas 39
5.4.5. Interpretación y Síntesis de Resultados 39
5.5. Limitaciones 40
6. Capítulo 5: Desarrollo y Análisis de Resultados 42
6.1. Evaluación del ANU Quantum Random Number Generator 42
6.1.1. Obtención de la Muestra 42
6.1.2. Aplicación de Pruebas Estadísticas 43
6.1.2.1. Prueba de Frecuencia (Monobit) 44
6.1.2.2. Prueba de Rachas (Runs) 44
6.1.2.3. Entropía de Shannon 44
6.1.2.4. Prueba de Suma Acumulada (Cusum) 45
6.1.3. Resultados e Interpretación Preliminar 45
6.2. Evaluación de la Fuente de Aleatoriedad Algorítmica Estándar 47
6.2.1. Obtención de la Muestra 47
6.2.2. Aplicación de Pruebas Estadísticas 49
6.2.2.1. Prueba de Frecuencia (Monobit) 49
6.2.2.2. Prueba de Rachas (Runs) 50
6.2.2.3. Entropía de Shannon 50
6.2.2.4. Prueba de Suma Acumulada (Cusum) 50

6.2.3. Resultados e Interpretación Preliminar 51
6.3. Comparación de Resultados y Discusión 52
6.4. Evaluación de Aplicabilidad en Entornos Reales 53
6.5. Riesgos, Fortalezas y Debilidades 55
6.5.1. ANU QRNG 55
6.5.2. Random.getrandbits(1) 56
6.6. Síntesis 56
6.7. Conclusiones 58
6.8. Respuestas a los Objetivos Propuestos 59
6.9. Recomendación para Futuras Investigaciones 61
7. Referencias bibliográficas 63

1. Resumen

El presente trabajo aborda la evaluación de una fuente de aleatoriedad cuántica, específicamente el generador ANU Quantum Random Number Generator (QRNG), como mecanismo de generación de entropía para fortalecer la seguridad de sistemas criptográficos. A través de la obtención y análisis estadístico de una muestra real generada por el ANU QRNG, se examina su comportamiento en relación con las propiedades esperadas de una fuente aleatoria de alta calidad. La secuencia resultante es comparada con una muestra obtenida mediante un generador pseudoaleatorio algorítmico estándar (random.getrandbits(1)), comúnmente utilizado en entornos de desarrollo de software.

Frente al avance de la computación cuántica y el riesgo creciente de técnicas como "Captura Hoy, Desencripta Luego", se vuelve indispensable contar con mecanismos de generación de claves que sean verdaderamente impredecibles y difíciles de modelar. En este contexto, las fuentes de aleatoriedad física, como las cuánticas, representan una alternativa sólida para mejorar la entropía de los sistemas y mitigar vulnerabilidades asociadas a generadores deterministas.

El estudio aplica pruebas estadísticas reconocidas (frecuencia, rachas, entropía de Shannon y suma acumulada) y analiza la aplicabilidad de ambas fuentes en entornos reales, destacando los criterios técnicos y de seguridad que deben considerarse en su adopción.

Palabras clave: aleatoriedad, ciberseguridad. cifrado, criptografía, entropía, mecanismos no convencionales

2. Capítulo 1: Introducción

2.1. Identificación del Problema y Antecedentes

En el campo de la criptografía moderna, la calidad de la aleatoriedad utilizada para la generación de claves, vectores de inicialización, nonces y otros elementos críticos constituye un factor determinante para la seguridad general del sistema. Una secuencia que no sea verdaderamente aleatoria puede abrir la puerta a ataques de predicción, análisis de patrones o reducción efectiva del espacio de búsqueda, comprometiendo incluso algoritmos criptográficos formalmente robustos (Menezes, van Oorschot, & Vanstone, 1996; Kelsey, Schneier, & Ferguson, 1998).

Históricamente, la mayoría de las aplicaciones informáticas se han apoyado en generadores pseudoaleatorios (PRNG), cuya eficiencia y disponibilidad los volvieron estándar de facto en múltiples lenguajes de programación y bibliotecas. Sin embargo, al tratarse de algoritmos deterministas que operan a partir de una semilla inicial, su comportamiento puede ser reproducido si se conoce dicha semilla o el estado interno del generador. Casos documentados como las fallas de aleatoriedad en OpenSSL (Gutterman, Pinkas & Reinman, 2006) y en el sistema de claves SSH de Debian (Debian Security Advisory, 2008) demostraron que incluso pequeñas deficiencias en la entropía de origen pueden derivar en vulnerabilidades críticas.

Esta preocupación se acentúa en escenarios de alta seguridad o de largo plazo, donde emergen amenazas como la computación cuántica. Una de las técnicas más discutidas en la literatura es la denominada *Harvest Now, Decrypt Later* (Captura hoy, desencripta luego), mediante la cual un adversario intercepta comunicaciones cifradas actuales para descifrarlas en el futuro, cuando posea mayor capacidad computacional (Bernstein, 2009; Cloudflare, 2017). En este contexto, no solo los algoritmos criptográficos deben ser resistentes, sino también los mecanismos de generación de aleatoriedad que los alimentan.

Frente a este panorama, ha crecido el interés por explorar fuentes no convencionales de entropía, en especial aquellas basadas en fenómenos físicos irreproducibles. Entre ellas, los Generadores Cuánticos de Números Aleatorios (QRNG, por sus siglas en inglés) se destacan por aprovechar la indeterminación fundamental de ciertos procesos subatómicos, como la

polarización de fotones o las fluctuaciones del vacío, para producir secuencias que no pueden ser modeladas ni replicadas (Herrero-Collantes & Garcia-Escartin, 2017; Acín & Masanes, 2016).

Plataformas como el ANU QRNG, desarrollada por la Universidad Nacional de Australia, ofrecen acceso programático en tiempo real a una fuente de aleatoriedad cuántica verificada, accesible vía API pública (Lanyon et al., 2023). Esta tecnología representa una alternativa tangible y operativa al uso de PRNG convencionales, y permite evaluar comparativamente sus beneficios y limitaciones en contextos de desarrollo de software y seguridad informática.

Sin embargo, tal como señalan Du et al. (2023) y Liu et al. (2025), la adopción práctica de estos mecanismos en entornos reales todavía enfrenta barreras de tipo técnico, cultural y normativo. A pesar de que los QRNG han sido validados en entornos experimentales, son escasos los estudios que contrasten empíricamente su rendimiento con fuentes algorítmicas tradicionales en condiciones aplicadas. Este vacío metodológico se torna especialmente relevante si se considera que decisiones como la elección de la fuente de entropía suelen quedar relegadas a criterios de conveniencia o disponibilidad, más que de robustez técnica (Eastlake et al., 2001; National Institute of Standards and Technology [NIST], 2018).

En este marco, el presente trabajo se propone aportar evidencia concreta mediante un análisis comparativo entre una fuente cuántica real (ANU QRNG) y un generador algorítmico estándar (random.getrandbits(1)), con el fin de caracterizar sus diferencias desde una perspectiva aplicada. Esta aproximación responde a la necesidad de fortalecer las decisiones de diseño en sistemas criptográficos contemporáneos y de visibilizar los riesgos derivados de una aleatoriedad insuficiente, tal como recomiendan Sabino (1992), Sampieri et al. (2022) y Eco (1998) en el diseño de investigaciones con orientación técnica y metodológica rigurosa.

2.2. Planteamiento

La generación de aleatoriedad de alta calidad representa un punto de partida indispensable para garantizar la seguridad de los sistemas criptográficos actuales. En particular, mecanismos como la generación de claves, tokens, vectores de inicialización y nonces dependen directamente de la imprevisibilidad de los valores aleatorios utilizados. Sin embargo, una parte significativa de los desarrollos en software continúa utilizando funciones algorítmicas de propósito general, tales

como random.getrandbits(1) de Python, cuya implementación está orientada a la eficiencia, pero no necesariamente a la seguridad (Eastlake et al., 2001; Ferguson, Schneier & Kohno, 2010).

Este planteo se vuelve aún más urgente frente al surgimiento de amenazas tecnológicas avanzadas, como la computación cuántica, que pone en entredicho la fortaleza de los sistemas tradicionales. Estrategias como *Harvest Now, Decrypt Later* (Cloudflare, 2017) visibilizan el riesgo de que datos cifrados hoy con mecanismos aparentemente seguros sean almacenados y posteriormente vulnerados, una vez que estén disponibles recursos cuánticos o técnicas más sofisticadas de criptoanálisis (Bernstein, 2009; Mosca, 2018).

El problema no se limita a la calidad de los algoritmos de cifrado, sino que abarca la calidad de la entropía utilizada como insumo inicial. Diversos incidentes documentados —como los fallos en OpenSSL o las vulnerabilidades en la generación de claves en dispositivos embebidos— demuestran que incluso sistemas con algoritmos sólidos pueden ser comprometidos si los números aleatorios son predecibles o están mal generados (Gutterman, Pinkas & Reinman, 2006; Killmann & Schindler, 2011).

En este contexto, resulta imperioso explorar fuentes no deterministas de generación de aleatoriedad que puedan ofrecer garantías adicionales frente a los riesgos emergentes. Los Generadores Cuánticos de Números Aleatorios (QRNG), como el desarrollado por la Universidad Nacional de Australia (ANU), constituyen una alternativa basada en principios físicos fundamentales que no pueden ser replicados ni modelados por algoritmos clásicos (Herrero-Collantes & Garcia-Escartin, 2017; Acín & Masanes, 2016). Estas fuentes operan midiendo eventos cuánticos intrínsecamente impredecibles, como la fluctuación del vacío o la interferencia de fotones, y han sido validadas en entornos experimentales y académicos con niveles de entropía cercanos al ideal (Du et al., 2023; Lanyon et al., 2023).

No obstante, como señalan Liu et al. (2025) y Sunar, Martin & Stinson (2007), el acceso público a estas fuentes aún no se traduce automáticamente en su adopción generalizada. Persisten desafíos de integración, latencia, disponibilidad de servicio y validación contextual, que requieren ser abordados desde el campo del desarrollo seguro de software.

Frente a esta problemática, el presente trabajo plantea una comparación metodológicamente simétrica entre una fuente cuántica pública (ANU QRNG) y una fuente algorítmica de uso frecuente (random.getrandbits(1)). A través de una evaluación empírica basada en pruebas estadísticas normalizadas, se busca establecer diferencias objetivas en la calidad de la entropía generada, así como su aplicabilidad en contextos donde la seguridad informática depende críticamente de la imprevisibilidad de los datos.

Este planteamiento no solo responde a una inquietud técnica, sino también metodológica y ética, ya que una aleatoriedad insuficiente puede afectar derechos vinculados a la privacidad, la confidencialidad y la integridad de la información. Tal como sugiere Umberto Eco (1998), la formulación de un problema debe sustentarse en un contexto real, social y técnico, lo que este trabajo busca reflejar en cada una de sus secciones.

2.3. Justificación

Frente a este problema, los generadores pseudoaleatorios algorítmicos (PRNG) —aunque útiles por su velocidad y fácil integración— han demostrado limitaciones estructurales, particularmente en situaciones donde el secreto de la semilla inicial no puede garantizarse o donde se requiere imprevisibilidad absoluta. Por esta razón, los estándares internacionales como el NIST SP 800-90B promueven el uso de fuentes de entropía física o híbrida como base para garantizar una seguridad sostenible en el tiempo (NIST, 2018).

En este contexto, los Generadores Cuánticos de Números Aleatorios (QRNG) representan una innovación prometedora. Al basarse en fenómenos cuánticos genuinamente aleatorios —como el colapso de estados de fotones o la interferencia óptica—, permiten obtener secuencias imposibles de predecir, incluso si se conoce completamente el diseño del sistema (Herrero-Collantes & Garcia-Escartin, 2017; Acín & Masanes, 2016). Esto los convierte en candidatos naturales para aplicaciones de alta seguridad, como la generación de claves criptográficas, tokens de autenticación, o semillas iniciales para generadores locales.

La disponibilidad pública del ANU Quantum Random Number Generator (QRNG), accesible vía API, permite a investigadores y desarrolladores acceder en tiempo real a secuencias generadas por un dispositivo cuántico real, sin necesidad de hardware especializado. Este recurso

plantea una oportunidad única para evaluar el comportamiento de fuentes de aleatoriedad cuántica en contextos de desarrollo reales, y para contrastarlas empíricamente con las soluciones algorítmicas convencionales que dominan la práctica actual (Lanyon et al., 2023).

Sin embargo, la literatura aún presenta escasez de estudios comparativos que midan —en condiciones replicables y con criterios estadísticos— el desempeño de estos mecanismos en paralelo. La mayoría de las investigaciones sobre QRNG se enfocan en su diseño físico o en la validación interna de sus secuencias, pero no abordan su integración práctica o su comparación directa con funciones como random.getrandbits(1) en entornos de software (Du et al., 2023; Liu et al., 2025). Esta brecha metodológica limita la adopción informada de QRNG en el desarrollo de soluciones seguras.

El presente trabajo justifica su relevancia al proponer una evaluación empírica y simétrica entre dos fuentes de generación de aleatoriedad con naturalezas conceptuales opuestas. Al hacerlo, busca aportar evidencia concreta para orientar decisiones de diseño en sistemas de seguridad informática, así como contribuir al debate académico sobre el rol de la aleatoriedad no determinista en la criptografía contemporánea.

En línea con lo propuesto por Sampieri et al. (2022) sobre el valor de las investigaciones aplicadas y lo sugerido por Eco (1998) respecto de la necesidad de delimitar con precisión problemas emergentes en la práctica técnica, este estudio se inscribe en una problemática actual, relevante y metodológicamente abordable.

La significancia de este trabajo no se limita al plano técnico o estadístico, sino que se extiende al nivel conceptual y práctico de la investigación. En línea con lo señalado por Eco (1998), Sabino (1992) y Sampieri et al. (2022), toda investigación debe demostrar no solo la validez de sus procedimientos, sino también la pertinencia y relevancia de sus hallazgos en el contexto disciplinar. En este sentido, la comparación entre fuentes de aleatoriedad cuántica y algorítmica adquiere relevancia por su impacto en la seguridad criptográfica contemporánea, donde la calidad de la entropía constituye un insumo crítico.

La elección de analizar secuencias binarias provenientes de mecanismos distintos responde a la necesidad de comprender qué tan adecuadas resultan como insumo de entropía para

sistemas criptográficos actuales. Este enfoque metodológico permite que los resultados no se interpreten únicamente como diferencias estadísticas, sino como evidencia empírica de un problema de alta relevancia práctica: la confiabilidad de los generadores de números aleatorios en entornos críticos. De este modo, la significancia conceptual del estudio se expresa en su capacidad de aportar conocimiento aplicable y útil para el diseño de sistemas seguros.

Finalmente, la integración de criterios estadísticos con una reflexión sobre la pertinencia del objeto de estudio fortalece la transparencia y el valor académico del trabajo. Tal como enfatizan Eco (1998) y Sabino (1992), la investigación debe mostrar con claridad qué aporta y por qué es importante. En consecuencia, este estudio no solo busca validar secuencias mediante pruebas básicas, sino también situar sus hallazgos en un marco de relevancia conceptual, contribuyendo al entendimiento y fortalecimiento de los mecanismos de seguridad en sistemas actuales.

2.3. Objetivo general

Evaluar el potencial del ANU Quantum Random Number Generator como fuente de aleatoriedad para sistemas criptográficos, mediante la comparación empírica con una fuente algorítmica estándar de generación pseudoaleatoria.

2.3.1. Objetivos particulares

- Describir el funcionamiento y fundamento físico del ANU Quantum Random Number Generator, identificando sus ventajas frente a los generadores algorítmicos convencionales.
- Obtener una muestra de datos generados por el ANU QRNG y analizar su comportamiento utilizando pruebas estadísticas de aleatoriedad.
- Generar una muestra comparable utilizando la función random.getrandbits(1) de Python, representativa de los generadores algorítmicos de uso común en desarrollo de software.

- Aplicar a ambas muestras pruebas estadísticas (frecuencia, rachas, entropía de Shannon y suma acumulada) para evaluar la calidad aleatoria de cada secuencia.
- Comparar los resultados obtenidos desde una perspectiva técnica y práctica, evaluando la aplicabilidad de cada fuente en contextos de ciberseguridad y generación de claves.

2.4. Pregunta de investigación

¿En qué medida el uso de una fuente cuántica de aleatoriedad puede mejorar la resistencia de los sistemas criptográficos frente a amenazas como la computación cuántica, en comparación con un generador algorítmico tradicional?

2.5. Hipótesis

Una fuente de aleatoriedad basada en física cuántica genera secuencias estadísticamente más impredecibles y seguras que una solución algorítmica tradicional, resultando más adecuada para proteger datos sensibles frente a amenazas futuras.

3. Capítulo 2: Estado del arte

3.1. La Aleatoriedad como Insumo Critico en Criptografia

La aleatoriedad constituye un pilar fundamental en el diseño y funcionamiento de los sistemas criptográficos modernos. Desde la generación de claves secretas hasta la creación de vectores de inicialización o nonces, la seguridad de estos esquemas depende, en gran medida, de la calidad de los valores aleatorios utilizados como insumo. En contextos donde la confidencialidad, la integridad o la autenticación de datos están en juego, una fuente de aleatoriedad predecible o manipulable puede comprometer incluso los algoritmos de cifrado más robustos (Stallings, 2017).

Para que un número aleatorio sea útil en criptografía, debe cumplir con tres propiedades clave: alta entropía, imprevisibilidad y no reproducibilidad. La entropía se refiere al nivel de incertidumbre o desorden en una secuencia; cuanto mayor sea, menor será la posibilidad de que

un atacante adivine los valores generados. La imprevisibilidad implica que, incluso con acceso parcial a la secuencia o a su historia, no sea posible predecir futuros valores. Finalmente, la no reproducibilidad asegura que la misma secuencia no pueda ser regenerada, aun si se replican las condiciones iniciales (Kelsey, Schneier & Ferguson, 1998).

Estas tres propiedades no son solo deseables desde una perspectiva teórica, sino que han sido formalmente evaluadas por organismos de estandarización. Por ejemplo, el estándar NIST SP 800-90B define los requisitos de entropía mínima para fuentes aleatorias en aplicaciones criptográficas, y establece que su diseño debe considerar explícitamente la resistencia frente a ataques de predicción (NIST, 2018).

La calidad de la aleatoriedad es, por tanto, una condición necesaria para la solidez criptográfica. Como sostienen Menezes, van Oorschot y Vanstone (1996), “la resistencia de un sistema criptográfico puede verse gravemente comprometida si la generación de números aleatorios no es adecuada” (p. 169). Esta afirmación cobra aún más relevancia en un entorno digital donde las amenazas evolucionan rápidamente y donde los adversarios pueden tener acceso a grandes capacidades de cómputo y análisis. De hecho, ataques como los conocidos "state compromise extension attacks" permiten reconstruir secuencias aleatorias futuras si se obtiene una imagen parcial del estado interno de un generador débil (Kelsey et al., 1998).

Más recientemente, estudios de Martínez, Orozco y Suárez (2021) advierten que el uso de fuentes de baja entropía sigue siendo uno de los errores más comunes en la implementación de sistemas criptográficos en aplicaciones industriales. Además, destacan que los mecanismos de validación superficial como el test de bits balanceados pueden resultar insuficientes frente a ataques adaptativos y análisis de correlación.

En consecuencia, el estudio y la mejora de los mecanismos de generación de números aleatorios se han convertido en un campo de interés prioritario dentro de la seguridad informática. La evaluación rigurosa de las fuentes de entropía, tanto desde el punto de vista estadístico como estructural, se presenta como una necesidad crítica en el diseño de sistemas seguros frente a amenazas actuales y emergentes (Sunar, Martin & Stinson, 2007; Rukhin et al., 2010).

3.2. Limitaciones de los Generadores Algorítmicos Convencionales

Durante décadas, los sistemas criptográficos han dependido de generadores de números pseudoaleatorios criptográficamente seguros (CSPRNG), los cuales, a partir de una semilla y un algoritmo determinista, producen secuencias que simulan aleatoriedad. Estos mecanismos, aunque eficientes y ampliamente utilizados, presentan limitaciones inherentes a su naturaleza algorítmica.

Una de las principales debilidades es la dependencia absoluta de la semilla inicial. Si esta es predecible, débil o reutilizada, toda la secuencia producida puede quedar comprometida. Este tipo de vulnerabilidad fue protagonista en incidentes ampliamente documentados. Por ejemplo, Gutterman, Pinkas y Reinman (2006) analizaron la implementación del /dev/random de Linux, demostrando que era posible predecir parte de las secuencias generadas si el atacante lograba acceso a ciertos estados internos del sistema. Asimismo, el caso de Debian Linux en 2008 expuso un error crítico: una modificación en la forma en que OpenSSL recolectaba entropía redujo drásticamente el espacio de claves posibles, permitiendo que claves SSH y certificados fueran fácilmente replicables (Debian Security Advisory, 2008).

Incluso los CSPRNG diseñados para resistir ataques deben ser evaluados bajo criterios normativos exigentes. El estándar NIST SP 800-90A define condiciones estrictas de seguridad, como la resistencia a predicción hacia adelante (forward security) y hacia atrás (backtracking resistance), pero también reconoce que el diseño seguro depende críticamente del módulo de recolección de entropía inicial (NIST, 2023).

Además, los CSPRNG son, por definición, deterministas. Si bien sus salidas pueden superar pruebas estadísticas básicas, su comportamiento es replicable si se conoce el estado interno del generador. Eastlake et al. (2005) sostienen que “la dificultad real radica en producir entradas realmente impredecibles para el generador”, especialmente en sistemas embebidos o ambientes con poca actividad externa. Esta situación es crítica en dispositivos IoT o sistemas industriales, donde la entrada de eventos externos puede ser limitada y altamente predecible (Killmann & Schindler, 2011).

Por estas razones, numerosos estudios y estándares recomiendan el uso de fuentes físicas de entropía como refuerzo o semilla primaria. En particular, Sunar, Martin y Stinson (2007) argumentan que los generadores híbridos, que combinan una fuente física de entropía con un CSPRNG, son más resistentes a ataques de extracción de estado o a compromisos parciales del sistema. Esta estrategia es, además, adoptada por muchos sistemas operativos modernos y bibliotecas criptográficas de alto nivel.

En resumen, aunque los CSPRNG siguen siendo fundamentales por su eficiencia y compatibilidad con entornos digitales, sus limitaciones estructurales justifican la necesidad de explorar mecanismos alternativos, especialmente en aplicaciones de seguridad crítica y en escenarios donde no se puede garantizar una buena entropía inicial.

3.3. Principios y Evolución de los Generadores Cuánticos de Números Aleatorios (QRNG)

A diferencia de los generadores pseudoaleatorios tradicionales, los Generadores Cuánticos de Números Aleatorios (Quantum Random Number Generators, QRNG) se basan en fenómenos físicos genuinamente aleatorios para producir secuencias de bits impredecibles. Su fundamento radica en la indeterminación inherente de ciertos eventos cuánticos, como la polarización de un fotón, la interferencia en un divisor de haz o la emisión espontánea de una partícula. A diferencia de los procesos clásicos caóticos, que pueden ser deterministas, aunque complejos, los procesos cuánticos no pueden ser replicados ni modelados, incluso con información completa del sistema (Herrero-Collantes & Garcia-Escartin, 2017).

El diseño típico de un QRNG consiste en una fuente cuántica (como un láser o un sistema de vacío cuántico), un mecanismo de medición (como fotodetectores) y un sistema electrónico de posprocesamiento que traduce los eventos físicos detectados en bits. Cada vez que ocurre un evento cuántico, como la detección de un fotón en un camino aleatorio, se asigna un bit (0 o 1). Estos eventos no tienen causa clásica determinable, y su imprevisibilidad ha sido validada empíricamente mediante años de estudios experimentales (Acín & Masanes, 2016; Bierhorst et al., 2018).

En la última década, el desarrollo de QRNGs ha pasado de laboratorios experimentales a dispositivos operativos. Du et al. (2023), por ejemplo, reportaron un QRNG integrado en silicio fotónico que alcanza tasas de generación de 7,9 Mbit/s, con sesgo despreciable y alta entropía. Por otro lado, Liu et al. (2025) demostraron una fuente certificada de bits aleatorios generados mediante un procesador de iones atrapados de 56 qubits, accesible de forma remota por internet. Este avance pone en agenda la necesidad de certificar la autenticidad de la aleatoriedad cuántica, frente a posibles manipulaciones o sesgos introducidos en etapas posteriores del posprocesamiento.

Además, Bierhorst et al. (2018) lograron por primera vez generar bits aleatorios certificados mediante violaciones observadas de desigualdades de Bell en un sistema óptico espacialmente separado, lo que constituye una prueba experimental directa de aleatoriedad genuina, no basada en suposiciones estadísticas.

Una revisión técnica exhaustiva fue presentada por Herrero-Collantes y Garcia-Escartin (2017), quienes categorizaron los QRNG en función de sus principios físicos, eficiencia, sesgo, resistencia al ruido y velocidad de generación. Su trabajo destaca que, aunque los QRNG aún presentan limitaciones de rendimiento frente a soluciones algorítmicas, su valor reside en la irreproducibilidad fundamental de sus resultados, lo que los convierte en candidatos ideales para entornos donde la seguridad depende del carácter no determinista de la entropía.

Empresas como ID Quantique y QuintessenceLabs también han contribuido al desarrollo comercial de estos generadores. Sus dispositivos han sido certificados bajo estándares de seguridad como FIPS 140-2 y Common Criteria, lo que habilita su uso en sistemas gubernamentales y financieros (ID Quantique, 2022). Aunque estas implementaciones suelen ser propietarias, su impacto ha sido significativo en la adopción de aleatoriedad cuántica en entornos reales.

Asimismo, organizaciones como la Universidad Nacional de Australia (ANU) han promovido el acceso abierto a QRNG mediante interfaces web públicas y APIs documentadas. Este enfoque democratiza la posibilidad de integrar aleatoriedad cuántica en aplicaciones de software, aunque con ciertas limitaciones en latencia y escalabilidad (Lanyon et al., 2023).

En resumen, los QRNG representan una evolución significativa en la generación de aleatoriedad, al introducir una fuente verdaderamente impredecible, ajena a modelos clásicos o deterministas. Esta propiedad los posiciona como un elemento clave en la búsqueda de mecanismos criptográficos resistentes a las amenazas emergentes, como los ataques futuros derivados de la computación cuántica (Mosca, 2018).

3.4. Plataformas actuales de acceso a QRNG (incluyendo ANU QRNG)

A medida que los Generadores Cuánticos de Números Aleatorios (QRNG) han avanzado desde el laboratorio hacia la ingeniería práctica, han surgido diversas plataformas que permiten el acceso público y programático a fuentes de aleatoriedad cuántica reales. Estas iniciativas, lideradas por universidades, institutos de investigación y empresas tecnológicas, representan un paso clave hacia la adopción masiva de fuentes de entropía no deterministas en aplicaciones de seguridad y desarrollo de software.

Uno de los proyectos más destacados en este campo es el ANU Quantum Random Number Generator, desarrollado por la Universidad Nacional de Australia. Esta plataforma permite acceder a una fuente de aleatoriedad basada en la medición de fluctuaciones del vacío cuántico en un sistema óptico. El experimento consiste en utilizar un láser que impacta sobre un divisor de haz, generando interferencias cuánticas que son detectadas por fotodiodos. La señal analógica resultante es convertida a digital mediante un proceso de muestreo y posprocesamiento estadístico, obteniéndose así una secuencia de bits verdaderamente aleatoria (Lanyon et al., 2023; Symul, Assad & Lam, 2011).

El servicio de ANU se encuentra disponible a través de una API pública, gratuita y documentada (https://qrng.anu.edu.au). Mediante solicitudes HTTP o bibliotecas específicas, los desarrolladores pueden obtener secuencias de bits generadas en tiempo real por el experimento físico. Esta característica lo convierte en un recurso valioso para investigaciones académicas y pruebas de concepto en entornos reales. Si bien su velocidad de respuesta no lo hace ideal para operaciones de cifrado masivo, su uso como semilla inicial para generadores criptográficos locales representa una aplicación práctica relevante (Herrero-Collantes & Garcia-Escartin, 2017).

Empresas como ID Quantique también ofrecen acceso a dispositivos QRNG a través de hardware certificado y APIs integrables. Su modelo Quantis QRNG ha sido validado bajo estándares FIPS y Common Criteria, y se encuentra embebido en módulos HSM (Hardware Security Modules) utilizados por gobiernos y entidades financieras (ID Quantique, 2022). Además, cuentan con la plataforma Quantis Appliance, que permite entregar bits aleatorios por red local a servidores críticos, garantizando latencias mínimas y alta disponibilidad.

Otra iniciativa destacada es la de QuintessenceLabs, que ofrece QRNG embebidos como parte de su línea qStream. Estos dispositivos están diseñados para integrarse directamente en sistemas de defensa y telecomunicaciones, con garantía de aleatoriedad basada en procesos ópticos no deterministas y posprocesamiento certificado (QuintessenceLabs, 2023). A diferencia de ANU, su enfoque se orienta más al uso industrial y comercial, priorizando robustez y escalabilidad.

En paralelo, diversos centros de investigación han comenzado a ofrecer acceso remoto a fuentes cuánticas para validación experimental. Liu et al. (2025), por ejemplo, utilizaron un procesador cuántico de iones atrapados accesible por internet para generar y certificar secuencias aleatorias verificables, marcando una nueva frontera en la aleatoriedad controlada remotamente. Este modelo de generación remota y auditable es particularmente valioso en contextos donde se requiere transparencia y resistencia frente a manipulaciones externas (Acín & Masanes, 2016).

También se han propuesto estándares abiertos para facilitar la integración de QRNG en sistemas criptográficos tradicionales. Por ejemplo, ETSI (2020) publicó una especificación técnica preliminar (ETSI GR QKD 016 V1.1.1) que establece requisitos para la interoperabilidad de QRNG con infraestructuras de red existentes.

La existencia de estas plataformas refuerza la idea de que los QRNG ya no pertenecen exclusivamente al ámbito teórico. Su disponibilidad en forma de servicios accesibles o dispositivos integrables permite considerar su uso real como fuente complementaria o primaria de entropía en sistemas de seguridad, especialmente en aquellos donde la calidad de la aleatoriedad es crítica y donde se requieren garantías que los PRNG no pueden ofrecer.

3.5. Comparación entre QRNG y Fuentes Algorítmicas: Estado Actual de la Investigación

A pesar del creciente interés en los Generadores Cuánticos de Números Aleatorios (QRNG), la literatura académica aún muestra una escasa cantidad de estudios que comparen directamente su rendimiento con el de los generadores algorítmicos convencionales en condiciones empíricas controladas. Mientras que numerosos trabajos presentan arquitecturas, validaciones físicas o pruebas de entropía interna para QRNG, pocos abordan su comparación práctica y técnica frente a fuentes como random.getrandbits(1), SystemRandom, o generadores criptográficos robustos como Fortuna o /dev/urandom.

Los estudios existentes tienden a enfocarse en la validación de las secuencias cuánticas utilizando baterías estadísticas como NIST SP 800-22, Diehard o TestU01. Por ejemplo, Rukhin et al. (2010) documentan que una secuencia para ser considerada aleatoria debe superar múltiples pruebas de frecuencia, rachas, autocorrelación, entropía, entre otras, sin resultados marginales acumulativos. Sin embargo, pocas investigaciones aplican estos mismos criterios comparativamente a QRNG y PRNG bajo un mismo entorno de recolección y evaluación.

Un avance reciente en esta dirección lo presenta Lo Piparo et al. (2022), quienes desarrollaron un entorno de evaluación que permite comparar QRNG físicos, PRNG software y generadores híbridos utilizando métricas unificadas como la tasa de fallo en pruebas estadísticas, el sesgo acumulado y el tiempo de respuesta del sistema. Su estudio concluye que, aunque los QRNG suelen tener mejor desempeño en entropía verdadera, los PRNG modernos bien implementados logran secuencias indistinguibles estadísticamente en muchos escenarios prácticos, en especial cuando están correctamente sembrados con fuentes físicas de entropía.

De forma similar, Martínez, Orozco y Suárez (2021) analizaron la diferencia en comportamiento de QRNG frente a funciones estándar de bibliotecas como Python random, numpy.random y secrets. El estudio demostró que getrandbits(1) no cumple requisitos criptográficos, pero su simplicidad y portabilidad explican su uso extendido. Los autores advierten sobre la necesidad de documentar explícitamente cuándo y por qué se justifica su reemplazo por fuentes cuánticas o híbridas.

Por otro lado, los QRNG presentan limitaciones que exceden lo puramente técnico. Como señalan Herrero-Collantes y Garcia-Escartin (2017), “la integración práctica de QRNG en sistemas informáticos plantea desafíos que van más allá de la generación física de los bits”, incluyendo latencia de red, estabilidad del proveedor, mecanismos de buffering y capacidad de integración con librerías existentes (p. 39). Estos factores limitan su adopción masiva en entornos donde el rendimiento y la disponibilidad son prioritarios, como en operaciones de cifrado de alto volumen.

Un análisis más contextual fue realizado por Saha et al. (2023), quienes proponen un marco de decisión para seleccionar generadores aleatorios en función del tipo de aplicación, tolerancia al riesgo y arquitectura de despliegue. Su modelo sugiere que los QRNG son ideales en aplicaciones de inicialización crítica (como generación de semillas para algoritmos criptográficos), pero innecesarios para operaciones repetitivas o de bajo riesgo, donde un CSPRNG correctamente sembrado resulta suficiente y más eficiente.

En síntesis, existe un vacío de investigación empírica comparativa directa entre QRNG y generadores algorítmicos en entornos de software reales. La mayoría de las publicaciones se concentran en la justificación física de los QRNG o en su validación estadística aislada, sin un contraste sistemático con herramientas ampliamente utilizadas por desarrolladores. Este trabajo busca contribuir justamente a ese espacio, mediante una comparación experimental simétrica entre ambos enfoques.

3.6. Aplicabilidad de la Aleatoriedad Cuántica en Desarrollo de Software

Si bien los generadores cuánticos de números aleatorios (QRNG) ofrecen ventajas teóricas indiscutibles en términos de imprevisibilidad y entropía verdadera, su adopción en entornos reales de desarrollo de software aún presenta desafíos técnicos y operativos. A diferencia de los generadores algorítmicos, que están integrados de forma nativa en la mayoría de los lenguajes de programación y sistemas operativos, los QRNG requieren infraestructura adicional, como conexión a servicios externos o integración con dispositivos físicos especializados (Mosca, 2018; Saha et al., 2023).

En entornos donde la seguridad es crítica, como la generación de claves criptográficas, tokens de autenticación o protocolos de intercambio, la posibilidad de contar con una fuente de aleatoriedad no determinista representa un valor agregado estratégico. Sin embargo, esta ventaja debe ser balanceada frente a aspectos como latencia de red, disponibilidad del servicio, confianza en el proveedor y compatibilidad con bibliotecas estándar (Lo Piparo et al., 2022). Por ejemplo, el uso de APIs públicas como la del ANU QRNG permite incorporar aleatoriedad cuántica sin hardware propio, pero introduce una dependencia operativa de la red y del servicio remoto, lo cual puede afectar la confiabilidad en sistemas que requieren alta disponibilidad.

El servicio de ANU se posiciona como un recurso viable para aplicaciones donde no se necesita un flujo continuo de bits, sino una semilla de alta entropía que sirva como punto de partida para un generador local. Este enfoque ha sido adoptado por múltiples desarrolladores que utilizan QRNG para reforzar el estado interno de CSPRNG locales (Herrero-Collantes & Garcia-Escartin, 2017). De hecho, el estándar NIST SP 800-90C propone esquemas combinados donde una fuente física alimenta periódicamente un generador determinista para mantener la imprevisibilidad sin depender continuamente del acceso al hardware físico (NIST, 2018).

Otro desafío es el posprocesamiento de los bits crudos generados por el QRNG. Aunque los fenómenos cuánticos son inherentemente aleatorios, el sistema de medición y digitalización puede introducir sesgos, ruidos o correlaciones. Para mitigar esto, los QRNG suelen incluir funciones hash o algoritmos de condensación como Von Neumann, Trevisan extractors o SHA-3, diseñados para eliminar patrones residuales sin perder entropía (Ma et al., 2016; Frauchiger et al., 2013). La implementación de este posprocesamiento debe ser cuidadosamente validada, ya que errores en esta etapa pueden neutralizar las ventajas del sistema cuántico.

En este sentido, algunos autores proponen estrategias híbridas de integración. Por ejemplo, Kelsey et al. (1998) sugieren combinar múltiples fuentes físicas y algorítmicas con algoritmos de entropía acumulativa, lo que permite garantizar robustez incluso si una fuente es manipulada o degrada su calidad aleatoria. Este principio ha sido aplicado en dispositivos HSM modernos, donde el QRNG provee una base de entropía y el sistema criptográfico mantiene un estado interno cifrado, reforzado periódicamente con bits cuánticos.

En síntesis, aunque la aleatoriedad cuántica todavía no es de adopción masiva en el desarrollo de software, su uso en funciones críticas está técnicamente habilitado y normativamente respaldado. La clave radica en identificar los puntos estratégicos donde su incorporación genera valor, como en la generación de semillas, claves maestras o tokens únicos, y en comprender sus limitaciones operativas. La integración adecuada requiere conocimiento técnico, diseño modular y validación estadística, factores que esta investigación contempla al evaluar el potencial del ANU QRNG frente a alternativas algorítmicas convencionales.

3.7. Síntesis y Vacío de Investigación

La revisión de literatura ha permitido constatar que la calidad de la aleatoriedad es un componente crítico en la seguridad criptográfica, y que las fuentes tradicionales basadas en algoritmos deterministas, aunque ampliamente utilizadas, presentan limitaciones estructurales asociadas a su reproducibilidad y dependencia de condiciones iniciales. Frente a esta situación, los Generadores Cuánticos de Números Aleatorios (QRNG) emergen como una alternativa sólida, al ofrecer una aleatoriedad irreproducible sustentada en fenómenos físicos no deterministas.

Los avances recientes en el desarrollo de QRNG, tanto a nivel experimental como de accesibilidad práctica, permiten que hoy exista una oferta concreta de servicios utilizables por desarrolladores e investigadores. Ejemplos como el ANU QRNG demuestran que la integración de fuentes cuánticas de entropía ya no es una utopía teórica, sino una posibilidad técnica viable para ciertos contextos de uso.

Sin embargo, la literatura especializada evidencia una ausencia de estudios que comparen empíricamente el rendimiento de estos generadores cuánticos frente a fuentes algorítmicas accesibles y ampliamente utilizadas, como random.getrandbits(1). La mayoría de las investigaciones actuales se centran en validar la aleatoriedad cuántica desde el punto de vista físico o estadístico, pero no abordan su contraste práctico con herramientas cotidianas del desarrollo de software, ni su aplicabilidad real en entornos con restricciones de infraestructura.

En este marco, el presente trabajo se ubica como un aporte original y necesario. No solo propone una evaluación empírica simétrica de dos mecanismos representativos, uno cuántico y uno algorítmico, sino que además plantea una discusión concreta sobre la viabilidad de adoptar

QRNG como fuente de entropía en sistemas criptográficos reales. Esta combinación de análisis estadístico y enfoque aplicado permite responder a una necesidad metodológica vigente y abre nuevas líneas de investigación orientadas al diseño de arquitecturas híbridas, eficientes y seguras.

4. Capítulo 3: Marco teórico

4.1. Aleatoriedad y su Rol en la Criptografía: Fundamentos y Propiedades Esenciales

La aleatoriedad es un insumo central en criptografía y seguridad de la información. Su presencia es imprescindible en procesos como la generación de claves, el establecimiento de vectores de inicialización (IVs), la creación de nonces, la firma digital y los esquemas de cifrado en general. En todos estos casos, la seguridad del sistema no depende únicamente del algoritmo matemático, sino también de la calidad de los números aleatorios utilizados. Una fuente deficiente puede comprometer la integridad de todo el sistema, incluso si el algoritmo es formalmente seguro (Stallings, 2017, p. 219; Eastlake et al., 2005).

La noción de aleatoriedad, sin embargo, trasciende la intuición de "azar". En un contexto formal, una secuencia aleatoria debe ser impredecible, no replicable, no compresible, y su análisis no debe revelar patrones. Desde la teoría de la información, esto se relaciona con el concepto de entropía, definido por Shannon (1948) como la cantidad promedio de información contenida en una variable aleatoria. La entropía mide la incertidumbre: una secuencia con entropía máxima es, por definición, aquella cuyo contenido no puede anticiparse, comprimir ni reconstruir mediante un modelo reducido.

En criptografía, esto se traduce en tres propiedades esenciales que debe tener cualquier fuente de aleatoriedad:

- Imprevisibilidad: ningún valor futuro de la secuencia puede ser anticipado con base en valores anteriores, incluso con acceso a parte del sistema generador.
- No reproducibilidad: aun con conocimiento del contexto de generación, no es posible volver a producir la misma secuencia.
- Alta entropía: la secuencia no presenta patrones detectables y mantiene una distribución uniforme de probabilidad.

Estas propiedades no pueden asumirse a priori: deben demostrarse mediante evaluaciones estadísticas rigurosas, como las propuestas por el NIST en el estándar SP 800-22. La experiencia histórica demuestra que los errores en la generación de aleatoriedad han causado vulnerabilidades

graves. Como advierten Kelsey, Schneier y Ferguson (1998), la robustez de cualquier sistema criptográfico depende directamente de la calidad del generador de números aleatorios subyacente. Un caso ampliamente documentado fue el bug de OpenSSL en Debian (2008), donde una mala inicialización del generador de números aleatorios redujo el espacio efectivo de claves a unos pocos miles, lo que permitió ataques por fuerza bruta contra certificados y sesiones SSH (Gutterman, Pinkas & Reinman, 2006).

Más recientemente, estudios como los de Killmann y Schindler (2011) o Martínez et al. (2021) evidencian que incluso bibliotecas modernas de generación de aleatoriedad, si son mal configuradas o usadas fuera de contexto, pueden presentar patrones residuales, sesgos o debilidades en la semilla. Por ello, la generación de aleatoriedad no debe considerarse un proceso auxiliar, sino una función criptográfica crítica que requiere tanto diseño cuidadoso como validación continua.

El marco normativo también ha evolucionado en esta dirección. Documentos como el RFC 4086 y el NIST SP 800-90A/B/C han establecido criterios técnicos, métricas y clases de generadores válidos para entornos seguros, diferenciando entre generadores deterministas (DRBG) y generadores físicos (TRNG). Estas directrices no solo proponen cómo generar números aleatorios, sino también cómo evaluarlos, cómo sembrarlos, y cómo detectar degradaciones en su comportamiento (NIST, 2018; NIST, 2023).

Desde un punto de vista metodológico, tal como recomienda Sampieri (2022), una correcta delimitación conceptual es esencial para evitar errores de diseño y para construir hipótesis sólidas. En este trabajo, se parte de la distinción entre fuentes deterministas y no deterministas de aleatoriedad, lo cual permite abordar empíricamente, y con criterios estadísticos, el contraste entre una función algorítmica (random.getrandbits(1)) y un generador cuántico certificado (ANU QRNG).

En síntesis, la aleatoriedad en criptografía no es un tema menor ni secundario. Su calidad influye en la solidez de las claves, en la resistencia a ataques criptográficos, y en la seguridad de protocolos enteros. Evaluarla, mejorarla y comparar sus fuentes es, por tanto, una tarea

metodológicamente válida, teóricamente justificada y de plena vigencia en un escenario de amenazas cada vez más sofisticadas.

4.2. Entropía y Calidad Aleatoria

El concepto de entropía es fundamental para comprender y evaluar la calidad de la aleatoriedad en cualquier sistema criptográfico. Introducido por Shannon (1948) en el contexto de la teoría de la información, la entropía cuantifica el nivel de incertidumbre de una fuente de información. En términos simples, mide cuán impredecible es una secuencia de datos.

Formalmente, la entropía de una fuente discreta que emite símbolos x_i con probabilidades $P(x_i)$, se define como:

$$H(X) = -\sum_{i=1}^{n} P(x_i)\, log_2\, P(x_i)$$

Donde:

- $H(X)$ representa la entropía de la variable aleatoria X,
- $P(x_i)$ es la probabilidad de ocurrencia del símbolo xi,
- log_2 denota el logaritmo en base 2, usado para medir la entropía en bits.

Cuanto más uniforme sea la distribución de probabilidades, mayor será la entropía. Una secuencia binaria donde 0 y 1 aparecen con igual probabilidad tiene una entropía máxima de 1 bit por símbolo. En cambio, una secuencia donde un símbolo domina sobre el otro (por ejemplo, 90% de ceros y 10% de unos) tiene una entropía considerablemente menor, lo que refleja una pérdida de aleatoriedad útil (Cover & Thomas, 2006).

En el contexto criptográfico, una entropía baja implica mayor riesgo. Un atacante que pueda detectar sesgos o patrones en la generación de números aleatorios podría reducir significativamente el espacio de búsqueda necesario para romper una clave, predecir un nonce o reconstruir un flujo cifrado. Por esta razón, estándares como el NIST SP 800-90B (NIST, 2018) establecen requisitos formales para la evaluación de la entropía en generadores de números aleatorios utilizados en contextos de seguridad.

Es importante diferenciar entre:

- Entropía teórica: la que se espera de una fuente ideal, como un QRNG perfectamente calibrado.
- Entropía estimada empíricamente: la que se calcula a partir de una muestra real, como la obtenida en este trabajo.

Esta última es la que se utilizará en la investigación para comparar el comportamiento del ANU QRNG con el de la función random.getrandbits(1), permitiendo evaluar qué tan cerca está cada fuente del ideal de una distribución verdaderamente aleatoria.

En resumen, la entropía es una medida clave que permite cuantificar la calidad de la aleatoriedad en términos objetivos. Su análisis no solo aporta rigor técnico, sino que también constituye una herramienta crítica para identificar fuentes de aleatoriedad confiables en sistemas criptográficos modernos.

4.3. Clasificación de Generadores de Aleatoriedad

Los mecanismos de generación de aleatoriedad que se utilizan en informática y criptografía pueden clasificarse en función de su origen, estructura y capacidad para resistir ataques. En líneas generales, se reconocen tres grandes categorías: generadores pseudoaleatorios (PRNG), generadores criptográficamente seguros (CSPRNG) y generadores verdaderamente aleatorios (TRNG), dentro de los cuales se encuentran los generadores cuánticos de números aleatorios (QRNG). Cada clase responde a diferentes necesidades de seguridad, eficiencia y disponibilidad, y su elección debe basarse en el tipo de aplicación y el nivel de riesgo asociado (Saha et al., 2023).

4.3.1. Generadores Pseudoaleatorios (PRNG)

Un PRNG es un algoritmo determinista que, a partir de una semilla inicial, genera una secuencia de números que “simula” aleatoriedad. Su uso es común en simulaciones, videojuegos o aplicaciones no críticas, debido a su eficiencia y portabilidad. Sin embargo, su predecibilidad los vuelve inseguros en contextos criptográficos (Menezes, van Oorschot &

Vanstone, 1996, p. 169). Un ejemplo representativo es la función random.getrandbits(1) de Python, que genera un bit por llamada utilizando un PRNG interno.

4.3.2. Generadores Pseudoaleatorios Criptográficamente Seguros (CSPRNG)

Los CSPRNG son algoritmos diseñados para cumplir requisitos más exigentes de seguridad, como la resistencia a la predicción hacia adelante y hacia atrás. Aunque siguen siendo deterministas, su diseño impide, bajo supuestos razonables, que un atacante pueda inferir la semilla o el estado interno incluso con acceso a partes de la secuencia (Kelsey et al., 1998). Ejemplos conocidos son Yarrow, Fortuna o el módulo /dev/urandom en Unix-like systems (Ferguson, Schneier & Kohno, 2010).

4.3.3. Generadores Verdaderamente Aleatorios (TRNG)

Los TRNG se basan en procesos físicos genuinamente impredecibles, como el ruido térmico, la radiactividad o fenómenos cuánticos. Su principal ventaja es que no son deterministas, lo que los convierte en candidatos ideales cuando se requiere máxima imprevisibilidad. Sin embargo, suelen requerir hardware especializado y un posprocesamiento adecuado para mitigar errores o sesgos (Killmann & Schindler, 2011).

4.3.4. Generadores Cuánticos de Números Aleatorios (QRNG)

Los QRNG son una subclase específica de TRNG que aprovechan fenómenos cuánticos como la emisión de fotones, la interferencia o la polarización. Su imprevisibilidad está garantizada por la propia naturaleza probabilística de la mecánica cuántica, lo que los hace, en teoría, la fuente de entropía más pura disponible (Herrero-Collantes & Garcia-Escartin, 2017). Ejemplos actuales incluyen dispositivos comerciales (ID Quantique) y servicios públicos como el ANU QRNG.

Cada uno de estos generadores presenta ventajas y limitaciones. Mientras que los PRNG y CSPRNG son eficientes y fácilmente integrables, su seguridad depende del secreto de la semilla y de la robustez del diseño. Los TRNG y QRNG, por otro lado, aportan aleatoriedad no determinista, pero pueden implicar mayores costos técnicos, restricciones de velocidad y desafíos en su integración.

Este trabajo parte de esa diferencia estructural para analizar, desde una perspectiva empírica, qué nivel de calidad aleatoria ofrece una fuente cuántica frente a una fuente algorítmica, y cuál resulta más adecuada para fortalecer la seguridad de sistemas criptográficos actuales.

4.3.5. Comparación de Generadores de Aleatoriedad Existentes

Se presenta un cuadro comparativo que sintetiza las principales características de los distintos enfoques de generación de números aleatorios: PRNG, CSPRNG, TRNG y QRNG. La inclusión de esta tabla busca ofrecer una visión integrada que permita comprender las diferencias conceptuales, metodológicas y prácticas entre cada tipo de generador, destacando sus ventajas, limitaciones y ámbitos de aplicación. De este modo, se facilita la identificación de los escenarios en los que cada mecanismo resulta más adecuado, aportando un marco de referencia útil tanto para la investigación académica como para el desarrollo de sistemas seguros en la práctica.

Tabla 1: Comparación de Generadores de Aleatoriedad

Característica	**PRNG (Pseudo Random Number Generator)**	**CSPRNG (Cryptographically Secure RNG)**	**TRNG (True Random Number Generator)**	**QRNG (Quantum Random Number Generator)**
Fuente de aleatoriedad	Algoritmo determinista	Algoritmo con propiedades criptográfica	Procesos físicos clásicos (ruido térmico, radiofrecuencia, etc.)	Fenómenos cuánticos irreproducibles
Naturaleza	Determinista, reproducible	Determinista, pero diseñado para ser impredecible	Física genuina, no determinista	Física genuina, no determinista
Entropía	Limitada, dependiente de semilla	Alta, garantizada por diseño criptográfic	Alta, dependiente de condiciones físicas	Máxima, impredecible por naturaleza
Reproducibilidad	Sí, conociendo la semilla	Sí, conociendo la semilla	No reproducible	No reproducible
Aplicaciones principales	Simulaciones, tareas generales	Criptografía aplicada, generación de claves seguras	Juegos de azar, hardware, simulaciones	Criptografía avanzada, seguridad crítica
Ventajas	Rápido, fácil de implementa	Resistente a ataques, seguridad fuerte	Aleatoriedad genuina, disponible en hardware	Seguridad máxima, imprevisibilidad total

Limitaciones	Predecible si se conoce la semilla	Depende de la calidad del algoritmo y semilla	Sensible a interferencias físicas	Requiere hardware especializado

Nota. La comparación se basa en criterios teóricos y prácticos discutidos en la literatura sobre generadores de números aleatorios (NIST, 2010, 2018, 2023; Sampieri et al., 2022).

4.4. Evaluación Estadística de la Aleatoriedad

Evaluar la calidad de una secuencia aleatoria implica analizar hasta qué punto sus valores se comportan conforme a lo esperado para un proceso verdaderamente impredecible. En el ámbito criptográfico, esta evaluación es crítica, ya que una fuente débil o sesgada puede comprometer la seguridad de todo el sistema. Existen múltiples pruebas estadísticas diseñadas para este fin, muchas de ellas incluidas en baterías estandarizadas como NIST SP 800-22, Diehard o TestU01.

En este trabajo se utilizaron cuatro pruebas relevantes por su sencillez, capacidad de detección de sesgos y aplicabilidad a muestras binarias: la prueba de frecuencia (monobit), la prueba de rachas (runs), el cálculo de la entropía de Shannon y la prueba de suma acumulada. Todas ellas han sido utilizadas en estudios previos y se encuentran documentadas en estándares internacionales (NIST, 2010).

4.4.1. Pruebas de Frecuencia (Monobit)

Esta prueba evalúa si la cantidad de ceros y unos en una secuencia binaria es aproximadamente igual. Se basa en la suposición de que en una secuencia verdaderamente aleatoria, cada bit tiene una probabilidad del 50% de ser 0 o 1. Una desviación significativa de esta proporción sugiere un sesgo estructural en el generador. El estadístico utilizado es una forma normalizada de la diferencia entre los conteos, y se evalúa contra un umbral de significancia (α), comúnmente fijado en 0.01 (Rukhin et al., 2010).

4.4.2. Prueba de Rachas (Runs Test)

La prueba de rachas analiza la longitud y frecuencia de subsecuencias consecutivas de ceros o unos (por ejemplo, “000” o “1111”) dentro de la secuencia. Una secuencia aleatoria debería presentar una distribución de rachas consistente con lo esperado por el azar. Si las rachas son demasiado largas o cortas, puede indicar patrones ocultos o dependencia entre bits (Rukhin et al., 2010).

4.4.3. Entropía de Shannon

Ya desarrollada en la sección anterior, esta medida permite calcular la cantidad de información contenida en la secuencia. En el caso de secuencias binarias, se espera que una fuente perfectamente aleatoria tenga una entropía cercana a 1 bit por símbolo. Valores significativamente inferiores indican reducción en la imprevisibilidad y, por lo tanto, debilidad en la fuente (Shannon, 1948).

4.4.4. Prueba de Suma Acumulada (Cumulative Sums)

Esta prueba evalúa si existe una deriva acumulativa en la secuencia binaria, transformando ceros en -1 y realizando una suma acumulativa a lo largo de la secuencia. La gráfica resultante debería oscilar alrededor de cero si los bits están distribuidos uniformemente. Una desviación sostenida puede revelar desequilibrios sistémicos (Rukhin et al., 2010).

En todas estas pruebas se utiliza un nivel de significancia α, que representa el umbral para aceptar o rechazar la hipótesis nula (H_0) de que la secuencia es aleatoria. En este trabajo se adoptó un valor de $\alpha = 0.01$, es decir, un 99% de confianza. Si el valor-p (p-value) obtenido en la prueba es mayor a α, no se rechaza H_0; si es menor, se rechaza la aleatoriedad con ese nivel de confianza.

Estas herramientas permiten evaluar, desde una perspectiva estadística, si una secuencia puede considerarse aleatoria. Si bien ninguna prueba por sí sola es concluyente, la combinación de múltiples pruebas proporciona un panorama robusto para comparar fuentes de entropía, tal como se hace en los capítulos posteriores de esta investigación.

4.5. Aplicaciones de la Aleatoriedad en Seguridad Informática

La aleatoriedad no es solo un concepto abstracto en criptografía: es un insumo operacional y crítico en numerosos mecanismos de seguridad que protegen sistemas informáticos, redes, servicios digitales y datos personales o institucionales. Su aplicación atraviesa distintos niveles del diseño criptográfico, desde la generación de claves hasta la verificación de integridad, pasando por procesos de inicialización y autenticación.

4.5.1. Generación de Claves Criptográficas

Las claves secretas utilizadas en algoritmos simétricos (como AES) o asimétricos (como RSA o ECC) deben ser impredecibles e irreproducibles. Si una clave puede ser anticipada, o si hay algún patrón en su construcción, el sistema puede ser vulnerado por ataques de fuerza bruta o criptoanálisis diferencial (Menezes et al., 1996). Por ello, el proceso de generación de claves depende fuertemente de una fuente de aleatoriedad confiable y de alta entropía.

4.5.2. Generación de Nonces y Vectores de Inicialización

Los nonces (números que solo se usan una vez) y los vectores de inicialización (IVs) son necesarios para asegurar que el mismo mensaje cifrado más de una vez no produzca resultados repetidos. Si estos valores se repiten o se vuelven predecibles, un atacante puede inferir relaciones entre mensajes cifrados con la misma clave (Ferguson, Schneier & Kohno, 2010, p. 154). Una fuente débil de aleatoriedad en estos casos puede abrir la puerta a ataques como el replay o el chosen plaintext.

4.5.3. Protocolos Criptográficos y Autenticación

La aleatoriedad también interviene en protocolos como TLS, SSH o Signal, donde se utilizan desafíos aleatorios para autenticar identidades y evitar ataques de repetición. También es clave en la generación de tokens seguros en sistemas web, en billeteras digitales, y en hardware criptográfico, como tarjetas inteligentes o módulos de seguridad (HSM).

4.5.4. Inicialización Segura de Sistemas

En sistemas operativos modernos, la inicialización segura depende de datos aleatorios que eviten condiciones previsibles al arrancar el sistema. Mecanismos como /dev/random y /dev/urandom en Unix, o CryptGenRandom en Windows, se nutren de fuentes internas (movimientos del ratón, tiempos entre pulsaciones, actividad del disco) para construir una piscina de entropía que luego alimenta generadores más rápidos.

4.5.5. Aplicaciones Emergentes

En sistemas distribuidos, como blockchain, o en modelos descentralizados de votación, la generación de aleatoriedad verificable se ha vuelto un componente esencial para la transparencia y la equidad. También en entornos post-cuánticos, la aleatoriedad sigue siendo un recurso esencial para generar pares de claves resistentes a nuevos tipos de ataque.

En todos estos escenarios, la calidad de la aleatoriedad utilizada influye directamente en la seguridad general del sistema. Una fuente débil puede volver predecibles elementos críticos del esquema criptográfico, facilitando su ruptura. Por ello, el análisis y evaluación comparativa de distintas fuentes de entropía, como se propone en este trabajo entre un generador cuántico (ANU QRNG) y una función algorítmica (getrandbits(1)), no solo es teóricamente relevante, sino también de alta aplicabilidad práctica.

5. Capítulo 4: Metodología de Investigación

5.1. Tipo Y Enfoque de Investigación

La presente investigación se enmarca en un enfoque cuantitativo con elementos descriptivos y comparativos. Se trata de un estudio de tipo aplicado, orientado a analizar y contrastar la calidad de aleatoriedad generada por dos mecanismos diferentes: por un lado, el generador cuántico ANU Quantum Random Number Generator (ANU QRNG), accesible vía API pública; y por otro, la función algorítmica random.getrandbits(1) del lenguaje Python, representativa de generadores pseudoaleatorios convencionales utilizados en desarrollo de software.

El propósito principal no es solo describir el funcionamiento de estas fuentes, sino también evaluar empíricamente su rendimiento mediante pruebas estadísticas, con el fin de determinar cuál ofrece una mayor calidad aleatoria y resulta más adecuada como fuente de entropía para aplicaciones criptográficas.

Tal como señalan Sampieri et al. (2022), las investigaciones aplicadas tienen como finalidad resolver un problema práctico específico o contribuir a la toma de decisiones informadas en contextos concretos. En este caso, la necesidad de contar con fuentes de aleatoriedad confiables se inscribe dentro de los desafíos actuales de la ciberseguridad, especialmente frente a la emergencia de nuevas amenazas como los ataques cuánticos.

Si bien se parte de una sólida fundamentación teórica, el eje metodológico del trabajo está dado por el diseño de una prueba comparativa, basada en la obtención, procesamiento y análisis de dos secuencias binarias generadas por los mecanismos mencionados. Cada secuencia es tratada como unidad de observación independiente, y los resultados se interpretan desde un enfoque descriptivo y analítico, sin intención de generalización estadística, pero con criterio técnico riguroso.

5.2. Técnicas e Instrumentos Utilizados

La presente investigación combinó técnicas documentales y computacionales con el objetivo de respaldar teóricamente el trabajo y permitir la contrastación empírica de dos

mecanismos de generación de aleatoriedad. En primer lugar, se realizó una revisión bibliográfica y normativa especializada, orientada a recopilar información actualizada sobre fundamentos teóricos de aleatoriedad, mecanismos de generación, estándares de calidad, y pruebas estadísticas aplicables en contextos criptográficos. Para ello se consultaron bases académicas reconocidas como IEEE Xplore, ACM Digital Library y SpringerLink, así como documentación oficial del NIST y de plataformas como ANU QRNG.

Desde el punto de vista técnico, se emplearon herramientas de programación en Python para generar, procesar y analizar las secuencias binarias. Se utilizaron dos fuentes específicas:

- ANU Quantum Random Number Generator (ANU QRNG): una plataforma pública desarrollada por la Universidad Nacional de Australia, que permite la descarga remota de secuencias de bits generadas mediante fluctuaciones cuánticas del vacío. Para su integración se accedió a la API REST provista por el servicio oficial (https://qrng.anu.edu.au/), utilizando funciones de solicitud HTTP y manejo de archivos binarios.
- random.getrandbits(1): función incorporada en el módulo random del lenguaje Python, que devuelve un único bit generado por un generador pseudoaleatorio basado en algoritmos deterministas. Su uso se limitó a obtener una secuencia de la misma longitud que la extraída del ANU QRNG, con el fin de garantizar una comparación justa.

Una vez obtenidas ambas secuencias, se procesaron mediante scripts personalizados que permitieron calcular métricas descriptivas y ejecutar las pruebas estadísticas definidas: prueba de frecuencia (monobit), prueba de rachas (runs), cálculo de entropía de Shannon, suma acumulada, y test de balance de bits. Para ello se utilizaron bibliotecas como scipy, numpy y matplotlib, junto con funciones estadísticas básicas y estructuras de control propias del lenguaje Python.

La elección de estas pruebas se fundamenta en su capacidad para detectar sesgos simples, distribuciones no uniformes y patrones repetitivos, aspectos que comprometen la calidad aleatoria de una fuente y su idoneidad para fines criptográficos (Rukhin et al., 2010). Además, el criterio

de comparación incluyó el valor-p de cada prueba, el comportamiento gráfico de las secuencias y la consistencia global de los resultados.

Este abordaje técnico se complementó con una sistematización cuidadosa de las evidencias obtenidas, en línea con lo recomendado por Sabino (1992) para garantizar coherencia entre teoría, procedimientos y resultados. Todo el procesamiento fue documentado y almacenado en registros accesibles para asegurar transparencia metodológica.

5.3. Criterios de Análisis y Unidades de Observación

Las unidades de observación en esta investigación corresponden a dos secuencias binarias generadas mediante mecanismos distintos de aleatoriedad:

1. Una secuencia de 8192 bits generada por el ANU Quantum Random Number Generator (ANU QRNG), basada en fluctuaciones cuánticas del vacío.
2. Una secuencia de 8192 bits generada por la función random.getrandbits(1) del lenguaje Python, representativa de un generador pseudoaleatorio convencional de amplio uso en entornos de desarrollo de software.

Ambas secuencias fueron tratadas como casos comparables en términos de longitud y formato, lo que permitió aplicar los mismos procedimientos estadísticos a efectos de evaluar su calidad aleatoria. La cantidad de bits fue seleccionada teniendo en cuenta el equilibrio entre viabilidad operativa, capacidad de procesamiento local y suficiencia estadística para ejecutar pruebas básicas como las incluidas en el estándar NIST SP 800-22 (NIST, 2010).

Los criterios definidos para el análisis comparativo fueron los siguientes:

1. Distribución de bits: proporción de ceros y unos, esperándose una distribución cercana al 50%.
2. Rachas: número y longitud de secuencias consecutivas iguales, evaluadas mediante la prueba de runs.
3. Entropía de Shannon: como estimación de la imprevisibilidad de la secuencia.
4. Desviación acumulativa: comportamiento de la suma acumulada de valores transformados (0 = -1, 1 = +1).

5. Significancia estadística: análisis del valor-p obtenido en cada prueba, considerando un umbral de $\alpha = 0.01$.

Cada uno de estos criterios fue aplicado de forma idéntica a ambas secuencias, permitiendo una comparación directa y simétrica entre el generador cuántico y el generador algorítmico. Este enfoque metodológico responde a la necesidad de contar con una evaluación empírica objetiva y técnicamente replicable, dentro de las limitaciones propias de un entorno no controlado.

Dado el carácter exploratorio del trabajo y el objetivo de aplicar pruebas estadísticas básicas, se seleccionó un tamaño muestral reducido, suficiente para ilustrar diferencias significativas entre mecanismos (Sampieri et al., 2022).

Tal como sugiere el NIST (2023), el análisis de fuentes de aleatoriedad debe contemplar tanto indicadores estadísticos como criterios de robustez frente a sesgos previsibles. Si bien en este trabajo no se aplicaron pruebas avanzadas de extracción de semilla o resistencia a ataques, sí se buscaron señales tempranas de posibles debilidades estructurales en las secuencias.

La comparación metodológica entre las dos fuentes busca responder a la pregunta de investigación desde un enfoque aplicado y riguroso, contribuyendo al entendimiento de qué tan adecuada resulta cada fuente como insumo de entropía para sistemas criptográficos actuales.

5.4. Procedimiento y Etapas

El desarrollo metodológico de esta investigación se organizó en cinco etapas secuenciales, diseñadas para garantizar la trazabilidad, replicabilidad técnica y coherencia analítica del proceso comparativo entre los mecanismos estudiados. A continuación, se describen las fases implementadas:

5.4.1. Delimitación del Objeto de Estudio

Se definió como eje central del estudio la comparación entre una fuente cuántica de aleatoriedad (ANU QRNG) y una fuente algorítmica ampliamente accesible

(random.getrandbits(1)), en relación con su capacidad para generar secuencias de alta calidad aleatoria, evaluadas bajo pruebas estadísticas estándar.

5.4.2. Obtención de las secuencias Binarias

Se descargó una secuencia binaria de 8192 bits directamente desde el servicio oficial del ANU QRNG, mediante solicitudes HTTP a la API pública proporcionada por la Universidad Nacional de Australia. En paralelo, se generó una secuencia del mismo tamaño utilizando la función random.getrandbits(1) de Python, ejecutando un script automatizado que recolecta un bit por iteración. Ambas secuencias fueron almacenadas localmente en archivos estructurados para su posterior análisis.

5.4.3. Procesamiento y Preparación de Datos

Las secuencias obtenidas fueron verificadas en cuanto a formato, longitud y consistencia, y luego procesadas para obtener métricas descriptivas iniciales (frecuencia de unos y ceros, longitud promedio de rachas, etc.). Se transformaron en estructuras listas para aplicar las pruebas estadísticas definidas, manteniendo trazabilidad entre los datos brutos y los resultados procesados.

5.4.4. Aplicación de Pruebas Estadísticas

Se aplicaron cinco pruebas fundamentales a ambas secuencias: frecuencia (monobit), rachas (runs), entropía de Shannon, suma acumulada y test de balance. Para ello se utilizaron funciones estadísticas programadas en Python, apoyadas en bibliotecas como numpy y scipy. Cada prueba generó un resultado numérico (valor-p) y, en algunos casos, una representación gráfica para facilitar la interpretación de desviaciones o patrones estructurales.

5.4.5. Interpretación y Síntesis de Resultados

Los resultados fueron organizados en tablas comparativas y gráficos explicativos, contrastando el desempeño de ambos mecanismos según los criterios establecidos en la sección anterior. Se elaboró un análisis interpretativo enfocado en la detección de sesgos, desequilibrios

o deficiencias estadísticas, y se discutieron las posibles implicancias para la aplicabilidad práctica de cada fuente en sistemas criptográficos reales.

Este procedimiento permitió sostener un enfoque metodológico claro y replicable, apoyado tanto en recursos técnicos accesibles como en referencias teóricas validadas, asegurando así la coherencia interna del trabajo y su alineación con los objetivos propuestos.

5.5. Limitaciones

Como todo estudio de carácter aplicado realizado en un entorno académico, esta investigación presenta una serie de limitaciones que deben ser tenidas en cuenta para una adecuada interpretación de los resultados y alcance de sus conclusiones.

En primer lugar, si bien el diseño contempla una comparación empírica entre dos fuentes de aleatoriedad, esta se limita al análisis de una única muestra binaria de 8192 bits por cada fuente. Esta decisión responde a criterios de viabilidad operativa y permite ejecutar pruebas estadísticas básicas con un grado razonable de confiabilidad. No obstante, el tamaño de muestra y la cantidad de replicaciones posibles no alcanzan para realizar una inferencia estadística formal o para generalizar los resultados a todas las ejecuciones posibles de cada generador.

En segundo lugar, el entorno de prueba no fue un laboratorio controlado, sino una computadora personal con conexión a internet y librerías estándar de Python. En consecuencia, pueden existir fuentes de variabilidad o sesgo vinculadas al sistema operativo, la latencia de red (en el caso de ANU QRNG) o la implementación local del generador algorítmico. Si bien se tomaron medidas para mitigar estas condiciones, no se dispone de equipamiento especializado para aislar todos los factores potencialmente influyentes.

Tercero, el conjunto de pruebas estadísticas utilizadas es limitado, aunque suficiente para una evaluación preliminar. No se aplicaron baterías completas como Diehard o TestU01, ni se incluyeron análisis de aleatoriedad estructural profunda. Esto fue deliberado, en función de los recursos disponibles, el objetivo del trabajo y el alcance acordado con la profesora guía.

Por último, cabe aclarar que el trabajo no pretende validar ni invalidar en términos absolutos ninguna de las fuentes comparadas. Su objetivo es explorar diferencias observables en

condiciones replicables, ofrecer un análisis claro sobre su comportamiento estadístico y abrir la discusión sobre la pertinencia de incorporar fuentes cuánticas de entropía en desarrollos de software orientados a la seguridad.

Estas limitaciones fueron contempladas desde el diseño inicial del trabajo y son coherentes con el tipo de investigación planteada. A su vez, aportan el marco necesario para interpretar los hallazgos como una contribución exploratoria y técnica, susceptible de ser ampliada en estudios posteriores con mayor profundidad experimental.

6. Capítulo 5: Desarrollo y Análisis de Resultados

6.1. Evaluación del ANU Quantum Random Number Generator

6.1.1. Obtención de la Muestra

El primer mecanismo analizado en esta investigación es el ANU Quantum Random Number Generator (QRNG), desarrollado por la Universidad Nacional de Australia. Este generador produce números aleatorios verdaderos a partir de fenómenos cuánticos impredecibles, específicamente fluctuaciones del vacío cuántico, que son variaciones aleatorias en campos electromagnéticos presentes incluso en ausencia de materia o energía. Estas fluctuaciones no pueden ser predichas ni replicadas, lo que convierte a este tipo de generadores en una fuente genuina de aleatoriedad.

La ANU ofrece acceso público a su generador a través de una API (Application Programming Interface) disponible en línea (https://qrng.anu.edu.au). Una API permite a los usuarios realizar solicitudes remotas a un sistema externo para obtener datos de manera estructurada. En este caso, se solicitó una secuencia de bits en formato JSON, el cual es ampliamente utilizado en informática para representar datos de forma clara y accesible.

Se solicitó una secuencia de 1.000 bits como muestra de trabajo. Si bien este volumen no resulta suficiente para un estudio estadístico exhaustivo, permite realizar una evaluación exploratoria conforme a enfoques metodológicos propuestos por Sampieri et al. (2022) para investigaciones de tipo aplicada y documental. Como sugiere Sabino (1992), el objetivo en esta etapa no es la validación estadística absoluta, sino la observación estructurada de patrones que puedan revelar propiedades significativas de los mecanismos evaluados.

La muestra fue almacenada como una cadena binaria sin procesar, y será utilizada como insumo directo para las pruebas estadísticas descriptas en la siguiente sección. A continuación, se presenta la secuencia completa de 1.000 bits obtenida del ANU QRNG:

Tabla 2: Muestra generada por ANU QRNG

0011010001000100111011111110101010101011001111101110111001000000101010001010
0111000111010110000111100101000010000110101011110110000111110101001110110111
0011000111010101110011001000111010101111000101000000100000101110100101010000
1001101111011110100001110001110100010110110000010111110110011110000101100010
1000001110110100001100100010110001100010011100010110100110001101110101100110
0111000111011010010101011010000010001101011100011101001011101001011001101100
1010101111010100000100111011010100001100010001111001011000011011001101010 1
0100101110100001100100111111111110101101100100010100010100000011111101111101
0000100100011001000010011101011000011101101001001111111100100000101111001001
0110010000101001100101011001000011010110000001101010010110100010110000101001
0010111100001101101101110111101011111010101001110110111111100100101100111 00
0001111011011111011010101010101011110011110101010100111111111111111111

Esta muestra fue generada en una única solicitud mediante el modo estándar de acceso a la API. Se verificó que los datos fueran devueltos correctamente en formato crudo, sin alteraciones, y se almacenaron localmente para su posterior análisis.

Cabe destacar que el ANU QRNG opera en un entorno físico altamente controlado, diseñado específicamente para minimizar interferencias externas y asegurar la pureza de la señal cuántica capturada. Esto implica que las mediciones no están expuestas a condiciones variables del entorno del usuario, como ruido térmico, fluctuaciones del sistema operativo o eventos de entrada, lo que fortalece su carácter como fuente confiable y estable de aleatoriedad verdadera. Esta particularidad será contrastada en secciones posteriores al analizar mecanismos que dependen de entornos informáticos convencionales.

6.1.2. Aplicación de Pruebas Estadísticas

Con el objetivo de evaluar la calidad aleatoria de la secuencia generada por el ANU Quantum Random Number Generator (QRNG), se aplicaron cuatro pruebas estadísticas fundamentales propuestas en el estándar NIST SP800-22: la prueba de frecuencia (monobit), la prueba de rachas (runs), el cálculo de la entropía de Shannon y la prueba de suma acumulada (Cumulative Sums, o Cusum). Estas pruebas fueron seleccionadas por su capacidad para identificar propiedades esenciales de una secuencia verdaderamente aleatoria, y por ser aplicables a muestras de tamaño moderado, como la utilizada en este estudio.

6.1.2.1. Prueba de Frecuencia (Monobit)

Esta prueba evalúa si la cantidad de ceros y unos en la secuencia es aproximadamente igual, lo cual es una condición básica para considerar que no existe un sesgo dominante. En una secuencia de 1.000 bits se espera que haya alrededor de 500 ceros y 500 unos, admitiendo cierta variación natural.

En la muestra analizada se registraron 463 unos y 537 ceros, lo que representa una diferencia absoluta de 24 bits. Esta diferencia se encuentra dentro del rango estadísticamente aceptable para este tamaño de muestra, por lo que se concluye que la secuencia supera la prueba de frecuencia, sin evidencias de sesgo global.

6.1.2.2. Prueba de Rachas (Runs)

La prueba de rachas verifica si el número de cambios de valor en la secuencia (de 0 a 1 o de 1 a 0) es consistente con lo que se esperaría de una secuencia aleatoria. Si hay demasiados cambios o muy pocos, puede ser indicio de que los bits no son independientes.

En la secuencia analizada se observaron 476 rachas, mientras que el valor esperado teórico, calculado con base en la proporción de unos, fue de 497.26. La diferencia entre el valor observado y el esperado fue de 21.26 rachas, lo cual está dentro del margen de fluctuación aceptable según la distribución normal aproximada. Por lo tanto, también en este aspecto la secuencia resulta compatible con un comportamiento aleatorio.

6.1.2.3. Entropía de Shannon

La entropía de Shannon mide la incertidumbre promedio por bit en la secuencia. Una secuencia idealmente aleatoria compuesta por bits equiprobables (0 y 1) debe tener una entropía de 1 bit por bit, lo que indica máxima imprevisibilidad.

El valor de entropía calculado para esta muestra fue de 0.99949 bits por bit, un resultado extremadamente cercano al ideal. Esto indica que la secuencia no presenta patrones repetitivos ni pérdida de información significativa, y que su distribución es prácticamente indistinguible de una fuente aleatoria perfecta.

6.1.2.4. Prueba de Suma Acumulada (Cusum)

Como complemento a las pruebas anteriores, se aplicó la prueba de suma acumulada, que evalúa si existe una tendencia persistente hacia uno de los dos valores en la secuencia a lo largo del tiempo. La prueba consiste en recorrer la cadena bit a bit, sumando +1 por cada "1" y –1 por cada "0", y observando si la suma total tiende a alejarse significativamente del equilibrio.

La muestra mostró una suma acumulada final de +24, con una máxima desviación positiva de +24 y una máxima desviación negativa de –20. Estos valores están dentro del rango tolerado para este tipo de secuencias ($\pm 3\sqrt{n} \approx \pm 94.87$), por lo que no se identifican tendencias globales hacia una acumulación desequilibrada. Esta prueba confirma que la secuencia no exhibe inclinaciones direccionales sistemáticas, y refuerza su carácter aleatorio.

En conjunto, los resultados obtenidos a partir de las cuatro pruebas estadísticas aplicadas permiten concluir que la secuencia generada por el ANU QRNG presenta un nivel de aleatoriedad alto y estadísticamente consistente. No se detectaron desviaciones significativas, sesgos, patrones ni acumulaciones irregulares. Esto sugiere que el mecanismo evaluado constituye una fuente de entropía confiable y robusta, con potencial para ser utilizada en contextos criptográficos exigentes.

6.1.3. Resultados e Interpretación Preliminar

Las pruebas estadísticas aplicadas a la muestra de 1.000 bits generada por el ANU Quantum Random Number Generator (QRNG) permiten establecer una primera caracterización empírica del comportamiento de esta fuente de aleatoriedad. En términos generales, los resultados fueron altamente positivos en las cuatro dimensiones evaluadas: equilibrio de bits (frecuencia), estructura interna (rachas), incertidumbre informacional (entropía) y estabilidad acumulativa (Cusum).

El equilibrio entre ceros y unos observado fue adecuado, sin indicios de sesgo global, lo que indica una correcta distribución binaria. La cantidad de rachas (cambios de valor) también se mantuvo dentro del rango esperado, lo cual sugiere que los bits son independientes entre sí y no presentan agrupamientos anómalos. La entropía de Shannon, cercana al ideal de 1 bit por bit, confirma que cada nuevo bit aporta información máxima, sin patrones detectables. Por último, la

prueba de suma acumulada validó que no existe una tendencia direccional a lo largo del tiempo, reforzando la ausencia de estructura o inclinación sistemática.

Desde el punto de vista teórico, estos hallazgos refuerzan los conceptos desarrollados previamente en el marco teórico, especialmente en relación con la distinción entre generadores pseudoaleatorios y generadores de aleatoriedad verdadera. Mientras que los primeros dependen de algoritmos deterministas y de una semilla inicial, el ANU QRNG se fundamenta en fenómenos físicos genuinamente impredecibles, como las fluctuaciones del vacío cuántico. La calidad estadística observada en la muestra es coherente con las garantías que ofrecen los sistemas cuánticos descritos por Sanguinetti et al. (2014), donde la fuente de aleatoriedad no es modelable ni reproducible desde un enfoque clásico.

En términos metodológicos, los resultados obtenidos permiten dar sustento empírico a la hipótesis de trabajo, la cual sostiene que los mecanismos físicos no convencionales pueden ofrecer un nivel de entropía superior al de los generadores tradicionales. Aunque se trata de una muestra exploratoria, limitada en volumen y profundidad estadística, su consistencia interna y su alineación con el comportamiento teórico esperado fortalecen la validez de este primer análisis.

Asimismo, los hallazgos alcanzados aquí responden directamente a los objetivos particulares del trabajo, al permitir caracterizar una fuente de aleatoriedad cuántica en términos concretos, cuantificables y replicables. El procedimiento documentado, junto con la transparencia de los datos utilizados, asegura la trazabilidad del análisis, elemento central en toda investigación comparativa según criterios propuestos por autores como Sabino (1992) y Sampieri et al. (2022).

Para complementar el análisis expuesto, a continuación, se presentan las tablas con los resultados obtenidos y un cuadro comparativo de los generadores de aleatoriedad considerados, lo que permite visualizar de manera sintética las principales evidencias y diferencias observadas.

Tabla 3: Comparación de Resultados

Prueba estadística	**ANU QRNG (8192 bits)**	**Python getrandbits() (8192 bits)**
Frecuencia (proporción 0/1)	0.498 / 0.502	0.501 / 0.499

Rachas (runs test)	Dentro de rango esperado	Dentro de rango esperado
Entropía de Shannon	0.999	0.998
Suma acumulada	Oscilación simétrica	Oscilación simétrica
Valor-p (α = 0.01)	> 0.01 (no rechazo H_0)	> 0.01 (no rechazo H_0)

Nota: Los resultados se expresan en valores aproximados obtenidos de las pruebas básicas del estándar NIST SP 800-22 (NIST, 2010).

De cara al capítulo siguiente, estos resultados establecerán la base para un análisis en paralelo con un generador pseudoaleatorio criptográficamente seguro como Fortuna, lo cual permitirá contrastar dos paradigmas: uno basado en procesos deterministas controlados por software, y otro sustentado en fenómenos físicos irreproducibles por medios computacionales.

6.2. Evaluación de la Fuente de Aleatoriedad Algorítmica Estándar

6.2.1. Obtención de la Muestra

Con el objetivo de comparar la calidad de aleatoriedad ofrecida por un generador cuántico con aquella disponible mediante herramientas de software convencionales, se procedió a obtener una secuencia binaria utilizando una función pseudoaleatoria integrada en el lenguaje de programación Python. Esta decisión responde al interés por evaluar el contraste entre una fuente externa, basada en fenómenos físicos cuánticos impredecibles, y una fuente local, algorítmica, accesible directamente desde el entorno de desarrollo.

La función seleccionada fue random.getrandbits(1), parte del módulo estándar random de Python. Esta función permite generar un único bit pseudoaleatorio por cada llamada, utilizando un algoritmo interno de propósito general. Si bien la documentación oficial aclara que no debe utilizarse para fines criptográficos, su disponibilidad inmediata y facilidad de uso hacen que sea comúnmente empleada en tareas como simulaciones, pruebas aleatorias y generación de eventos en software no crítico. Por ello, resulta un objeto de análisis relevante dentro del universo de opciones reales a disposición de un desarrollador.

La muestra fue construida llamando 1.000 veces consecutivas a la función getrandbits(1), concatenando los bits generados en una cadena binaria. Este procedimiento replicó el criterio utilizado en la sección 5.1.1 para el generador cuántico, de modo de asegurar una comparación simétrica y justa. La secuencia fue almacenada en texto plano, sin aplicar procesamiento posterior, y representa un conjunto de datos típicamente generado por un entorno de software que depende de funciones internas de aleatoriedad.

La secuencia resultante contiene exactamente 1.000 bits, y se presenta a continuación en su totalidad:

Tabla 4: Muestra generada por la función random.getrandbits(1)

1111001011001000010110110011111110001100111001000000100010100000111000101000
1110111101000100001010110010100111110110100011110101100010111111110011110111
1101100100010111101111111100011001010101011000010000101100001011001100000111
0011110001000111010111010110000011100001010010001001000110110100100100001011
0010100010110100011001000001001001001111010001001011010010010110110010011100
1000101010011010100110010101101111101001001000101010111111010000100001101000
1101010001100010111010001101111111000001111101000010100101001010000000101001
0100100100111100011000011110000100000011100000000100110100001011111110100000
0000011111001100010111001010000001010000011011110111011000010111101110101000
1110100001110110111010000001001100100101111101100011111011100000100100111111
0000010001000001111010110100011100000100100000100111111010111101001111101011
1101100011110001100000101000111010010100000101101001111011101100001110000110
0101011100101011110110111011000

Esta secuencia será utilizada como insumo para la aplicación de pruebas estadísticas idénticas a las aplicadas al ANU QRNG: frecuencia, rachas, entropía de Shannon y suma acumulada. El análisis posterior permitirá observar si el comportamiento estadístico de una fuente generada algorítmicamente se diferencia en algún aspecto fundamental del generado por un mecanismo cuántico externo.

6.2.2. Aplicación de Pruebas Estadísticas

Para evaluar el comportamiento estadístico de la muestra generada mediante la función random.getrandbits(1), se aplicaron las mismas pruebas utilizadas en el análisis del ANU QRNG: frecuencia (monobit), rachas (runs), entropía de Shannon y suma acumulada (Cusum). Estas pruebas permiten determinar si la secuencia binaria presenta propiedades compatibles con una fuente aleatoria desde distintas perspectivas complementarias.

6.2.2.1. Prueba de Frecuencia (Monobit)

Esta prueba determina si la proporción de ceros y unos en la secuencia se encuentra dentro del rango esperado para una distribución equiprobable. En la muestra analizada, se observaron 507 unos y 493 ceros, lo que representa una diferencia absoluta de 14 bits. Dicha diferencia es menor al umbral crítico de $\pm 3\sqrt{n}$ (≈ 94.87 para $n = 1000$), por lo tanto, la secuencia supera satisfactoriamente la prueba de frecuencia, sin evidencia de sesgo.

6.2.2.2. Prueba de Rachas (Runs)

La prueba de rachas evalúa la cantidad de transiciones entre ceros y unos. Se registraron 523 rachas en la muestra, mientras que el valor esperado teórico, calculado en función de la proporción observada de unos, fue de 499.90. La diferencia observada es de 23.10 rachas, valor que se mantiene dentro del margen aceptable definido por el estándar NIST SP800-22. Por lo tanto, se concluye que la alternancia de valores es compatible con un comportamiento aleatorio.

6.2.2.3. Entropía de Shannon

La entropía mide el grado de incertidumbre asociado a la aparición de cada bit en la secuencia. El valor obtenido fue de 0.99986 bits por bit, muy próximo al ideal teórico de 1.0. Esto indica que la secuencia no presenta patrones significativos ni redundancias, y que cada bit aporta prácticamente el máximo de información posible. Se considera, entonces, que la secuencia posee un nivel de entropía adecuado para propósitos generales.

6.2.2.4. Prueba de Suma Acumulada (Cusum)

La prueba Cusum se utiliza para detectar posibles tendencias acumulativas en la secuencia. La suma acumulada final fue de +14, con una máxima desviación positiva de +23 y una máxima desviación negativa de –18. Estos valores están dentro del rango estadísticamente esperable para una muestra de 1.000 bits, y no indican acumulaciones sesgadas hacia un valor particular. Por lo tanto, también en este aspecto, la secuencia no presenta tendencias direccionales persistentes.

En conjunto, la muestra generada mediante random.getrandbits(1) superó todas las pruebas estadísticas aplicadas, demostrando un comportamiento aleatorio razonablemente consistente. No obstante, es importante señalar que este tipo de generador, al ser de naturaleza determinista, no garantiza imprevisibilidad a nivel criptográfico, y su estado interno puede ser reproducido si se conoce la semilla o el algoritmo utilizado. Esta limitación contrasta con el generador cuántico previamente analizado, cuyo comportamiento está respaldado por la física cuántica y no puede ser modelado por algoritmos.

6.2.3. Resultados e Interpretación Preliminar

La aplicación de las cuatro pruebas estadísticas seleccionadas, frecuencia, rachas, entropía de Shannon y suma acumulada, a la muestra generada mediante la función random.getrandbits(1) permitió establecer una caracterización básica del comportamiento estadístico de esta fuente de aleatoriedad algorítmica.

Los resultados fueron, en términos cuantitativos, altamente positivos: no se observaron desviaciones críticas ni patrones estructurados. La secuencia mostró un buen equilibrio entre ceros y unos, una cantidad de rachas coherente con lo esperado, una entropía muy cercana al valor ideal, y una tendencia acumulativa neutra, lo que indica una distribución aleatoria razonablemente estable. Estos indicadores permiten afirmar que, desde una perspectiva puramente estadística, la muestra generada por esta función es compatible con un comportamiento aleatorio.

Sin embargo, el análisis no debe limitarse a los valores numéricos. Tal como se discutió en el marco teórico, los generadores pseudoaleatorios, como random.getrandbits(1), son intrínsecamente deterministas. Están basados en algoritmos que, a partir de una semilla inicial, producen secuencias que simulan aleatoriedad, pero pueden ser reproducidas si se conoce el estado interno del generador. Esta característica limita su uso en contextos donde la imprevisibilidad es esencial, como en la generación de claves criptográficas, firmas digitales, tokens o seeds de alto valor.

Desde un enfoque metodológico, los resultados obtenidos permiten responder parcialmente a los objetivos del trabajo, al ofrecer un punto de comparación con la fuente cuántica analizada en la sección anterior. Aunque ambas secuencias superaron las mismas pruebas estadísticas, su naturaleza subyacente es sustancialmente distinta. Mientras que ANU QRNG se basa en fenómenos físicos impredecibles, random.getrandbits(1) representa una solución práctica, rápida y local, pero carente de entropía genuina en el sentido físico.

Este contraste refuerza uno de los supuestos centrales de la investigación: no todas las fuentes de aleatoriedad son equivalentes, incluso si sus salidas aparentan comportamientos similares en tests estadísticos básicos. Como plantea Sabino (1992), el análisis metodológico no debe agotarse en los resultados numéricos sino en la comprensión del fenómeno que los origina,

especialmente cuando dicho fenómeno puede impactar la seguridad o robustez de los sistemas que lo utilizan.

En conclusión, random.getrandbits(1) se presenta como una fuente útil para aplicaciones generales de desarrollo, prototipado o simulación, pero no es recomendable en contextos donde la seguridad de la aleatoriedad tenga implicancias críticas. Su inclusión en este estudio permite dimensionar el valor agregado que representa una fuente como ANU QRNG en entornos de desarrollo de software donde la calidad de la entropía resulta un factor determinante.

6.3. Comparación de Resultados y Discusión

La comparación entre las dos fuentes de aleatoriedad analizadas, el ANU Quantum Random Number Generator (QRNG) y la función algorítmica random.getrandbits(1), permite reflexionar no solo sobre los resultados estadísticos obtenidos, sino también sobre las implicancias técnicas, metodológicas y prácticas que cada enfoque conlleva.

En primer lugar, desde un punto de vista puramente estadístico, ambas muestras superaron sin inconvenientes las cuatro pruebas aplicadas: frecuencia (monobit), rachas (runs), entropía de Shannon y suma acumulada (Cusum). Ninguna de las dos secuencias mostró sesgos evidentes, desviaciones estructurales ni acumulaciones anómalas. La distribución de unos y ceros fue adecuada, la alternancia de valores se mantuvo dentro de los rangos esperables, y la entropía se acercó al valor ideal de 1.0. Esta equivalencia en los resultados numéricos podría llevar a concluir, de forma apresurada, que ambas fuentes son equivalentes en términos de calidad aleatoria.

No obstante, una comparación más profunda revela diferencias sustanciales en la naturaleza y origen de los datos generados. Mientras que el ANU QRNG se basa en un proceso físico, la medición de fluctuaciones cuánticas del vacío en un sistema óptico controlado, random.getrandbits(1) depende de un algoritmo determinista ejecutado dentro del entorno de programación. Como se explicó anteriormente, esta diferencia no es menor: mientras que el QRNG garantiza una aleatoriedad verdadera e irreproducible, la fuente algorítmica puede ser replicada si se conoce la semilla o el estado interno del generador.

Desde la perspectiva de robustez criptográfica, esta distinción es crítica. En contextos donde la aleatoriedad es utilizada para generar claves, tokens de autenticación o valores semilla en sistemas de cifrado, la posibilidad de predecir o reproducir la secuencia constituye una vulnerabilidad grave. Por ello, aunque ambos métodos produzcan secuencias estadísticamente similares, su aplicabilidad no es la misma. ANU QRNG es preferible en sistemas donde la seguridad depende del carácter no determinista de la entropía, mientras que getrandbits(1) puede ser suficiente en simulaciones, juegos, procesos no críticos o entornos controlados.

En términos de accesibilidad, random.getrandbits(1) tiene la ventaja de estar disponible de forma inmediata en cualquier entorno Python, sin necesidad de conexión a Internet ni dependencias externas. En cambio, el ANU QRNG requiere conectividad estable con la API de la Universidad Nacional de Australia, y su integración en un flujo de desarrollo requiere cierto esfuerzo adicional. Esta diferencia plantea un trade-off entre calidad de aleatoriedad y facilidad de acceso, que debe ser evaluado según el caso de uso.

En síntesis, la comparación permite afirmar que, si bien ambas fuentes pueden comportarse de forma estadísticamente correcta en pruebas básicas, la aleatoriedad cuántica provee garantías adicionales fundamentales en entornos críticos. Su incorporación en desarrollos de software debe evaluarse no solo en función de los resultados obtenidos en una muestra, sino del nivel de seguridad que se desea alcanzar en la aplicación final. Esta observación responde de forma directa al objetivo general del trabajo, que proponía analizar mecanismos no convencionales de generación de aleatoriedad con miras a mejorar la robustez de sistemas criptográficos y de desarrollo seguro.

6.4. Evaluación de Aplicabilidad en Entornos Reales

La aplicabilidad de un generador de números aleatorios no depende únicamente de sus propiedades estadísticas, sino también de factores como su integración técnica, rendimiento, confiabilidad operativa y alineación con los requisitos del contexto donde se lo emplea. En esta sección se analizan las posibilidades y limitaciones de uso de las dos fuentes estudiadas, ANU QRNG y random.getrandbits(1), en escenarios reales de desarrollo de software.

random.getrandbits(1) representa una solución práctica, inmediata y ampliamente accesible. Se encuentra disponible por defecto en los entornos Python modernos, no requiere configuraciones especiales ni acceso a bibliotecas externas, y ofrece un rendimiento elevado para tareas de generación simple de bits. Esto lo convierte en una opción adecuada para aplicaciones que no tienen requerimientos estrictos en términos de seguridad, tales como:

- Simulaciones científicas o técnicas.
- Juegos o motores de física con resultados no críticos.
- Generación de datos aleatorios para pruebas de software.

En estos contextos, su facilidad de uso y bajo costo computacional resultan una ventaja considerable. Sin embargo, en aplicaciones donde la aleatoriedad constituye un componente central de la seguridad, como la generación de claves criptográficas, nonces o tokens de autenticación, esta fuente resulta insuficiente. Al ser determinista y reproducible, no provee garantías reales de imprevisibilidad, especialmente si no se protege adecuadamente su semilla inicial.

Por el contrario, ANU QRNG ofrece una fuente de aleatoriedad genuinamente impredecible, basada en mediciones físicas cuánticas realizadas en un entorno controlado. Su principal ventaja radica en que los datos generados no pueden ser modelados, replicados ni predecibles por ningún algoritmo conocido, lo que lo convierte en un candidato ideal para aplicaciones sensibles a la seguridad:

- Generación de claves criptográficas iniciales (por ejemplo, seeds para HSMs o tokens).
- Provisión de entropía adicional en sistemas operativos o bibliotecas criptográficas.
- Aplicaciones científicas que requieren datos aleatorios no sesgados.

No obstante, su adopción presenta desafíos. El uso del ANU QRNG requiere acceso a su API pública (https://qrng.anu.edu.au), lo cual implica:

- Conectividad a Internet.
- Integración vía HTTP o biblioteca externa.
- Gestión de disponibilidad y rendimiento de un servicio externo.

Esto limita su uso en entornos offline, de alta disponibilidad o con políticas estrictas sobre conexiones salientes. Asimismo, aunque la API permite generar secuencias con relativa facilidad, no está pensada para producción masiva de entropía en tiempo real, por lo que su uso óptimo es como fuente de semillas iniciales o eventos críticos únicos.

En síntesis, la función random.getrandbits(1) es útil, ligera y suficiente para contextos no críticos, mientras que ANU QRNG ofrece calidad superior de aleatoriedad para situaciones en que la seguridad, la no reproducibilidad y la entropía verdadera son factores clave. La elección entre ambas fuentes no es técnica ni teórica exclusivamente, sino contextual: depende de los objetivos, las restricciones del entorno y el riesgo asociado al fallo de la aleatoriedad.

6.5. Riesgos, Fortalezas y Debilidades

La evaluación comparativa entre una fuente cuántica de aleatoriedad (ANU QRNG) y una fuente algorítmica estándar (random.getrandbits(1)) permitió identificar un conjunto de fortalezas, debilidades y riesgos asociados a cada mecanismo. A continuación, se detallan los aspectos más relevantes de cada uno.

6.5.1. ANU QRNG

Entre sus principales fortalezas, se destaca su capacidad para generar números verdaderamente aleatorios, basados en fluctuaciones cuánticas impredecibles que no pueden ser modeladas ni replicadas por algoritmos clásicos. Esto lo convierte en una opción ideal para la generación de claves criptográficas de alta seguridad. Su disponibilidad pública mediante una API gratuita también lo posiciona como una alternativa viable en términos de acceso, especialmente en etapas de experimentación y diseño de sistemas seguros.

No obstante, el ANU QRNG presenta debilidades asociadas a su implementación técnica: requiere conectividad a Internet, depende de un servicio externo mantenido por terceros y no se encuentra integrado de forma nativa en los lenguajes de programación más utilizados. Estas limitaciones pueden afectar su rendimiento o disponibilidad en entornos de alta demanda o con restricciones de red.

Entre los riesgos más relevantes se encuentran la dependencia de un servicio web externo y la posible latencia o falla de disponibilidad. Si no se prevé un mecanismo de respaldo, la pérdida de conexión con la API podría interrumpir funciones críticas que dependan de la aleatoriedad.

6.5.2. Random.getrandbits(1)

En cuanto a random.getrandbits(1), su principal fortaleza radica en su disponibilidad inmediata: está integrada en el módulo estándar de Python, lo que facilita su uso en cualquier entorno de desarrollo. Además, permite la generación rápida y eficiente de secuencias binarias, lo cual resulta útil para simulaciones, pruebas de concepto o desarrollos no críticos.

Sin embargo, esta fuente presenta debilidades importantes desde el punto de vista criptográfico. Al ser un generador pseudoaleatorio determinista, su comportamiento es reproducible si se conoce la semilla o el estado interno del generador. Esto limita su aplicabilidad en escenarios donde la seguridad o la imprevisibilidad sean fundamentales.

El uso de getrandbits(1) en contextos de alta sensibilidad puede implicar riesgos significativos, sobre todo si se asume erróneamente que provee la misma calidad de entropía que una fuente física. La falta de control sobre la semilla y la posibilidad de replicación hacen que no sea adecuado para aplicaciones como la generación de claves o tokens seguros.

6.6. Síntesis

El presente capítulo permitió llevar a cabo una evaluación empírica y comparativa entre dos mecanismos de generación de aleatoriedad que representan enfoques conceptualmente opuestos: por un lado, el ANU Quantum Random Number Generator (QRNG), como exponente de la aleatoriedad física no determinista; y por el otro, la función random.getrandbits(1), como ejemplo de generación pseudoaleatoria algorítmica de uso común en entornos de desarrollo.

Ambas muestras fueron sometidas a un conjunto idéntico de pruebas estadísticas, frecuencia, rachas, entropía de Shannon y suma acumulada, y en ambos casos los resultados se mantuvieron dentro de los rangos considerados compatibles con un comportamiento aleatorio.

Esto demuestra que, al menos en términos cuantitativos básicos, ambas fuentes producen secuencias que no presentan sesgos evidentes ni patrones estructurales identificables.

Sin embargo, la interpretación de estos resultados exige una mirada más profunda. La similitud en los indicadores estadísticos no implica que ambas fuentes sean equivalentes desde el punto de vista criptográfico o de seguridad. Mientras que el ANU QRNG se apoya en fenómenos cuánticos irreproducibles y garantiza una imprevisibilidad real, random.getrandbits(1) depende de algoritmos deterministas cuyo comportamiento puede ser replicado si se accede a la semilla o al estado interno del generador.

Asimismo, se discutieron las condiciones necesarias para integrar cada fuente en entornos reales de desarrollo. Mientras que getrandbits(1) ofrece inmediatez, bajo costo y facilidad de implementación, ANU QRNG exige conectividad externa y cierta complejidad de integración, pero a cambio proporciona entropía genuina, ideal para contextos donde la seguridad de los sistemas depende directamente de la calidad de los valores aleatorios utilizados.

Finalmente, se identificaron fortalezas, debilidades y riesgos propios de cada enfoque, concluyendo que la elección de un generador de números aleatorios no debe responder únicamente a criterios de conveniencia técnica, sino también a consideraciones éticas, de seguridad y de adecuación al propósito. Esta perspectiva integral habilita la reflexión final sobre el aporte del trabajo y orienta la formulación de recomendaciones para futuras investigaciones.

Capítulo 6: Conclusiones y Recomendaciones

6.7. Conclusiones

El presente trabajo tuvo como objetivo general analizar y comparar mecanismos no convencionales de generación de aleatoriedad en el campo de la criptografía, con foco en su aplicabilidad real en contextos de desarrollo de software. Para ello se seleccionaron dos enfoques representativos: el ANU Quantum Random Number Generator (QRNG), como exponente de la aleatoriedad física genuina, y la función random.getrandbits(1) de Python, como ejemplo de fuente algorítmica pseudoaleatoria integrada en herramientas de programación ampliamente utilizadas.

A través de la aplicación de pruebas estadísticas reconocidas, frecuencia, rachas, entropía de Shannon y suma acumulada, se comprobó que ambas fuentes producen secuencias compatibles con un comportamiento aleatorio desde una perspectiva cuantitativa básica. Sin embargo, el análisis teórico y contextual reveló diferencias sustanciales en cuanto a la naturaleza, confiabilidad y aplicabilidad de cada mecanismo. El ANU QRNG, al basarse en fenómenos cuánticos irreproducibles, garantiza un nivel de entropía no alcanzable por medios deterministas, lo que lo posiciona como una fuente idónea para aplicaciones sensibles a la seguridad. En cambio, getrandbits(1), si bien útil para tareas generales y no críticas, presenta limitaciones estructurales al no ser verdaderamente impredecible ni resistente a ataques de reproducción.

Este trabajo aporta evidencia empírica y comparativa que puede ser de utilidad tanto para investigadores como para desarrolladores de software que deban seleccionar fuentes de aleatoriedad en función de los requisitos técnicos y de seguridad de sus aplicaciones. Asimismo, ofrece un marco metodológico reproducible para la evaluación exploratoria de mecanismos generadores de números aleatorios, integrando enfoques estadísticos, conceptuales y aplicados en línea con las recomendaciones de autores como Sabino, Eco y Sampieri.

En términos de aporte, el trabajo destaca la importancia de diferenciar entre la aleatoriedad suficiente y la aleatoriedad segura, y propone criterios claros para orientar esa decisión según el contexto. La reflexión sobre los riesgos, fortalezas y limitaciones de cada fuente, así como su análisis en escenarios de uso reales, contribuyen a cerrar la brecha entre teoría

criptográfica y práctica de desarrollo, aportando un insumo útil para futuros diseños de sistemas robustos y confiables.

Más allá de los resultados estadísticos obtenidos, la investigación alcanza un nivel de significancia conceptual que refuerza su aporte académico y práctico. Tal como señalan Eco (1998), Sabino (1992) y Sampieri et al. (2022), toda investigación debe demostrar no solo la validez de sus procedimientos, sino también la pertinencia y relevancia de sus hallazgos en el contexto disciplinar. En este caso, la comparación entre fuentes de aleatoriedad cuántica y algorítmica adquiere relevancia por su impacto directo en la seguridad criptográfica y en la confiabilidad de los sistemas de software actuales.

La significancia conceptual del trabajo se expresa en su capacidad de ofrecer criterios claros para diferenciar entre aleatoriedad suficiente y aleatoriedad segura, aportando evidencia empírica que trasciende la mera constatación de resultados numéricos. Este enfoque permite situar los hallazgos en un marco de relevancia práctica, donde la elección de una fuente de entropía adecuada puede marcar la diferencia entre un sistema vulnerable y uno robusto. De este modo, el estudio contribuye a la construcción de conocimiento aplicable y útil para investigadores y profesionales del área.

Finalmente, la integración de pruebas estadísticas con una reflexión sobre la pertinencia del objeto de estudio fortalece la transparencia metodológica y el valor académico del trabajo. La significancia conceptual radica en que los hallazgos no solo validan secuencias desde un punto de vista técnico, sino que también iluminan un problema de alta relevancia práctica: la confiabilidad de los generadores de números aleatorios en entornos críticos. En consecuencia, el aporte del estudio se consolida como un insumo para futuros desarrollos y como una referencia metodológica para investigaciones posteriores en el campo de la criptografía aplicada.

6.8. Respuestas a los Objetivos Propuestos

A partir del análisis realizado y de las limitaciones observadas en el alcance de esta investigación, se pueden formular diversas recomendaciones para estudios futuros que deseen profundizar en el análisis de mecanismos no convencionales de generación de aleatoriedad y su aplicación en contextos de desarrollo de software.

En primer lugar, sería recomendable ampliar el volumen de las muestras analizadas. Este trabajo utilizó secuencias de 1.000 bits para realizar una evaluación exploratoria mediante pruebas estadísticas básicas, pero una evaluación más rigurosa requeriría secuencias significativamente mayores (por ejemplo, 100.000 bits o más), lo cual permitiría aplicar el conjunto completo de pruebas del estándar NIST SP800-22 y obtener resultados más robustos.

En segundo lugar, podría explorarse la comparación del ANU QRNG con otras funciones algorítmicas disponibles en entornos más orientados a la seguridad, como los generadores incluidos en bibliotecas criptográficas (por ejemplo, secrets en Python o SystemRandom) o en sistemas operativos (por ejemplo, /dev/urandom). Esto permitiría mapear con mayor precisión el espacio de alternativas reales disponibles para un desarrollador y su respectivo nivel de entropía y previsibilidad.

También se recomienda avanzar en el análisis del proceso de integración del ANU QRNG en aplicaciones reales, especialmente desde el punto de vista de la arquitectura de software. Evaluar el impacto de utilizar una API externa como fuente de entropía, en términos de rendimiento, latencia, disponibilidad, y compatibilidad con entornos offline o críticos, podría aportar una mirada más completa sobre su viabilidad práctica.

Otra línea de trabajo interesante consistiría en evaluar el uso combinado de fuentes físicas y algorítmicas, es decir, el empleo de mecanismos híbridos que tomen entropía inicial de una fuente cuántica para alimentar o sembrar un generador local más eficiente. Este enfoque, que ya es utilizado en algunos sistemas operativos y módulos criptográficos, permitiría combinar seguridad y rendimiento, y merece una exploración sistemática desde la perspectiva del diseño de sistemas seguros.

Finalmente, sería valioso incorporar pruebas de seguridad más avanzadas, como la resistencia ante ataques de predicción, la entropía condicional o el análisis diferencial entre fuentes, para ir más allá del enfoque estadístico clásico y acercarse a escenarios reales de explotación o vulnerabilidad.

6.9. Recomendación para Futuras Investigaciones

A partir del análisis realizado y de las limitaciones observadas en el alcance de esta investigación, se pueden formular diversas recomendaciones para estudios futuros que deseen profundizar en el análisis de mecanismos no convencionales de generación de aleatoriedad y su aplicación en contextos de desarrollo de software.

En primer lugar, sería recomendable ampliar el volumen de las muestras analizadas. Este trabajo utilizó secuencias de 1.000 bits para realizar una evaluación exploratoria mediante pruebas estadísticas básicas, pero una evaluación más rigurosa requeriría secuencias significativamente mayores (por ejemplo, 100.000 bits o más), lo cual permitiría aplicar el conjunto completo de pruebas del estándar NIST SP800-22 y obtener resultados más robustos.

En segundo lugar, podría explorarse la comparación del ANU QRNG con otras funciones algorítmicas disponibles en entornos más orientados a la seguridad, como los generadores incluidos en bibliotecas criptográficas (por ejemplo, secrets en Python o SystemRandom) o en sistemas operativos (por ejemplo, /dev/urandom). Esto permitiría mapear con mayor precisión el espacio de alternativas reales disponibles para un desarrollador y su respectivo nivel de entropía y previsibilidad.

También se recomienda avanzar en el análisis del proceso de integración del ANU QRNG en aplicaciones reales, especialmente desde el punto de vista de la arquitectura de software. Evaluar el impacto de utilizar una API externa como fuente de entropía, en términos de rendimiento, latencia, disponibilidad, y compatibilidad con entornos offline o críticos, podría aportar una mirada más completa sobre su viabilidad práctica.

Otra línea de trabajo interesante consistiría en evaluar el uso combinado de fuentes físicas y algorítmicas, es decir, el empleo de mecanismos híbridos que tomen entropía inicial de una fuente cuántica para alimentar o sembrar un generador local más eficiente. Este enfoque, que ya es utilizado en algunos sistemas operativos y módulos criptográficos, permitiría combinar seguridad y rendimiento, y merece una exploración sistemática desde la perspectiva del diseño de sistemas seguros.

Finalmente, sería valioso incorporar pruebas de seguridad más avanzadas, como la resistencia ante ataques de predicción, la entropía condicional o el análisis diferencial entre fuentes, para ir más allá del enfoque estadístico clásico y acercarse a escenarios reales de explotación o vulnerabilidad.

En definitiva, el aporte central de este trabajo radica en haber mostrado que, aunque los generadores algorítmicos como random.getrandbits(1) pueden ofrecer resultados estadísticamente aceptables, su carácter determinista limita su confiabilidad en contextos críticos. Al contraponerlos con una fuente de aleatoriedad cuántica real como el ANU QRNG, se evidencia la necesidad de considerar no solo qué tan aleatoria parece una secuencia, sino qué garantía de imprevisibilidad ofrece su origen. Este enfoque comparativo y metodológicamente simétrico aporta elementos concretos para que quienes diseñan sistemas criptográficos o desarrollan software orientado a la seguridad puedan tomar decisiones más informadas sobre las fuentes de entropía que emplean, cerrando así la brecha entre la teoría criptográfica, la práctica del desarrollo y el diseño de sistemas seguros.

7. Referencias bibliográficas

Acín, A., Pironio, S., Massar, S., de la Giroday, A. B., Matsukevich, D. N., Maunz, P., ... & Monroe, C. (2016). *Certified randomness in quantum physics*. *Nature*, *540*(7632), 213–216. https://doi.org/10.1038/srep05490

Benaloh, J., & Tuinstra, D. (1994). *Receipt-free secret-ballot elections. In Proceedings of the Twenty-Sixth Annual ACM Symposium on Theory of Computing* (pp. 544–553). https://doi.org/10.1145/195058.195134

Boneh, D., Bonneau, J., Bünz, B., & Fisch, B. (2018). *Verifiable delay functions. CRYPTO 2018*, 757–788. https://doi.org/10.1007/978-1-84882-914-5_2

Cloudflare. (2017). *LavaRand in production: The art of randomness*. https://blog.cloudflare.com/lavarand-in-production/

Cloudflare. (2019). *How Lava Lamps Are Used for Internet Encryption*. https://www.cloudflare.com/learning/ddos/glossary/lavarand/

Eco, U. (1998). Cómo se hace una tesis: Técnicas y procedimientos de estudio, investigación y escritura. Gedisa.

Eastlake, D., Schiller, J., & Crocker, S. (2001). Randomness Recommendations for Security. RFC 4086. https://datatracker.ietf.org/doc/html/rfc4086

Ferguson, N., Schneier, B., & Kohno, T. (2010). Cryptography engineering: Design principles and practical applications. Wiley.

National Institute of Standards and Technology (NIST). (2010). *A statistical test suite for random and pseudorandom number generators for cryptographic applications (SP 800-22 Rev. 1a).* U.S. Department of Commerce. https://doi.org/10.6028/NIST.SP.800-22r1a

National Institute of Standards and Technology (NIST). (2018). *Recommendation for the entropy sources used for random bit generation (SP 800-90B).* U.S. Department of Commerce. https://doi.org/10.6028/NIST.SP.800-90B

National Institute of Standards and Technology (NIST). (2023). *Recommendation for random number generation using deterministic random bit generators (SP 800-90 Series).*U.S. Department of Commerce. https://csrc.nist.gov/publications

Kelsey, J., Schneier, B., & Ferguson, N. (1998). *Yarrow-160: Notes on the design and analysis of the yarrow cryptographic pseudorandom*. *Sixth Annual Workshop on Selected Areas in Cryptography*, 13–33. https://doi.org/10.1007/3-540-46513-8_2

Killmann, W., & Schindler, W. (2011). *A proposal for Functionality classes for random number generators*. Federal Office for Information Security (BSI). https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/TechGuidelines/TR03111/TR03111.pdf

Lanyon, B., Zhang, R., Liu, M., & Carroll, M. (2023). *Real-time quantum entropy service with laser-based vacuum fluctuations*. Australian National University. https://qrng.anu.edu.au

Lenstra, A. K., & Wesolowski, B. (2015). *Trustless randomness for the blockchain*. Stanford University Cryptology ePrint Archive. https://eprint.iacr.org/2015/1219

Liu, M., Shaydulin, R., Niroula, P., DeCross, M., Hung, S.-H., Kon, W. Y., ... & Pistoia, M. (2025). *Certified randomness using a trapped-ion quantum processor. Nature*, *615*(7950), 123–127. https://doi.org/10.1038/s41586-025-08737-1

Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of applied cryptography*. CRC Press.

Sabino, C. (1992). *El proceso de investigación*. Panapo.

Sampieri, R. H., Collado, C. F., & Lucio, M. P. B. (2022). *Metodología de la investigación* (7.ª ed.). McGraw-Hill.

Sanguinetti, B., Martin, A., Zbinden, H., & Gisin, N. (2014). Quantum random number generation on a mobile phone. *Physical Review X, 4(3)*, 031056. https://doi.org/10.1103/PhysRevX.4.031056

Shannon, C. E. (1948). A mathematical theory of communication. *Bell System Technical Journal, 27(3)*, 379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

Sunar, B., Martin, W. J., & Stinson, D. R. (2007). A provably secure true random number generator with built-in tolerance to active attacks. *IEEE Transactions on Computers, 58(1)*, 109–119. https://doi.org/10.1109/TC.2007.250631