Instituto Tecnológico de Estudios Superiores de Monterrey
Campus Monterrey
Ingeniería en Ciencias de Datos y Matemáticas

Andrea Bravo Avila A01028579
Renata Vargas Caballero A01025281
Natalia Monserrat González De León A00831090
Fernando González Rosas A01253694
Alberto Lozano Cárdenas A01067141
Fernanda Sherlin Calderón López A01275430
Gil Herzberg Alperon A00827347

24 de Febero, 2023

Implementación de criptografía de clave pública para la protección de comunicaciones y almacenamiento de datos en IoT en entorno de monitoreo y consumo de energía

Uso de Álgebras Modernas para Seguridad y Criptografía
Grupo 602

Alberto F. Martínez
Daniel Otero Fadul
OSF: LiCore

Índice

1. Absatract **2**

2. Introducción **2**

3. Marco Teórico **3**
3.1. Internet of Things . . . 3
3.1.1. Seguridad en IoT . . . 4
3.1.2. Plataformas IoT . . . 4
3.2. Esquemas de clave publica . . . 5
3.3. Protocolos de comunicación . . . 5
3.4. Arquitecturas de Red . . . 6

4. Estado del Arte **6**
4.1. Librerías . . . 8
4.2. Arquitecturas de red . . . 9
4.3. Recursos de Hardware . . . 10

5. Propuesta de solución del reto **12**
5.1. Firma digital con RSA . . . 13
5.2. AWS . . . 14
5.3. EC2 y RDS . . . 14
5.4. Arquitectura Sugerida . . . 15
5.5. Componentes de la propuesta . . . 15

6. Implementación de la propuesta **16**
6.1. Configuración de la base de datos AWS RDS . . . 16
6.2. Creación de la tabla MeterReadings . . . 16
6.3. Configuración del servidor AWS EC2 . . . 17
6.4. Creación del código de la aplicación Flask . . . 17
6.4.1. Pruebas de la aplicación Flask en la instancia EC2 . . . 18
6.5. Generacion de claves (Firma Digital) . . . 18
6.6. Transmisión de Datos . . . 19
6.7. Envío de Datos desde la Raspberry Pi . . . 19
6.8. Preprocesado de los Datos . . . 19
6.9. Interacción Segura con la Base de Datos Remota . . . 20

7. Referencias **21**

8. Tabla de Recursos **23**

1. Absatract

Debido a la necesidad de medidas de seguridad en los canales de comunicación entre distintos dispositivos inteligentes se propone, haciendo uso de herramientas computacionales como Python, Flask, OpenSSL, AWS, etc. la implementación de un cifrado de llave publica para salvaguardar la confidencialidad, autenticidad e integridad de la información recolectada por celdas solares inteligentes que mediante auditores será enviada por el internet a un centro de control en el cual se almacenaran.

2. Introducción

Los avances tecnológicos logrados en el último siglo han provocado grandes cambios en la manera en la que la sociedad se estructura, especialmente, en el ámbito de las telecomunicaciones, pues, en la actualidad el intercambio de información por medios inalámbricos que se realiza día a día, no solo por empresas e instituciones de gran magnitud o por personas individuales sino también mediante los llamados *dispositivos inteligentes* los cuales son capaces de recolectar información de manera autómata para su posterior uso y análisis, creando así el llamado "Internet de las cosas" o IoT por sus siglas en ingles [33] .

Pero esta conectividad entre dispositivos que nos ha permitido mejorar nuestros medios de comunicación entre distintos tipos de agentes no viene sin su contra parte de riesgos, ya que la transmisión de grandes cantidades de información, especialmente si esta es información sensible, es un gran imán de cibercriminales que buscaran vulnerar la integridad del medio de comunicación para ganar acceso a su contenido, permitiéndoles conocer información privilegiada con la que podrían realizar un ataque a cualquiera de las partes involucradas. Con esto en mente, es lógico ver a la criptografía como herramienta para proteger nuestra información, proveyendo **confidencialidad**, **integridad** y **autenticidad** a los mensajes enviados por los medios distintos medios de comunicación [33].

Tomando en cuenta esto se vuelve evidente la necesidad de generar una solución para la protección de los datos de los clientes que usan las herramientas de LiCore, una empresa que en la búsqueda de una solución para la crisis energética y la necesidad de hacer un cambio hacia energías sustentables desarrolla herramientas que con el uso de tecnologías inteligentes permite que medidores inteligentes recopilen información sobre el uso de energía eléctrica, la cual es enviada por el internet a un centro de control donde es analizada haciendo uso una red como la mostrada en la figura 1. Por lo que en este trabajo se busca encontrar un esquema de red el cual permita proteger las información intercambiada entre los dispositivos inteligentes y el centro de control, haciendo uso de un esquema de clave publica con la cual encriptar dicha información. [33]

Específicamente para el caso de LiCore debemos de tomar en cuenta que la

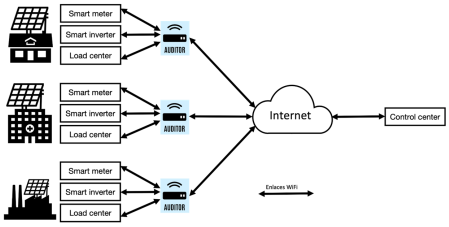


Figura 1: Esquema de intercambio de datos a ser protegidos (Canvas).

arquitectura de red consiste de medidores inteligentes los cuales están conectados a un auditor digital el cual se comunica por un canal bidireccional por el internet con un centro de control como se muestra en la figura 2

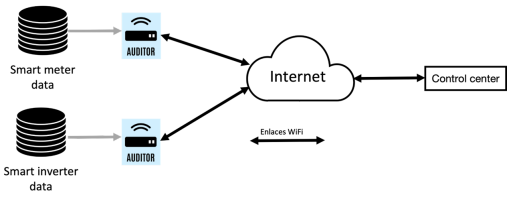


Figura 2: Arquitectura de red de LiCore (Canvas).

3. Marco Teórico

3.1. Internet of Things

Internet of Things (IoT) es un nuevo paradigma que combina aspectos y tecnologías provenientes de diferentes enfoques. La computación ubicua, la informática ubicua, el protocolo de Internet, las tecnologías de detección, las tecnologías de comunicación y los dispositivos integrados se fusionan para formar un sistema en el que los mundos real y digital se encuentran y están continuamente en interacción simbiótica. El objeto inteligente es el componente básico de la visión IoT. Al colocar inteligencia en los objetos cotidianos, se convierten en objetos inteligentes capaces no solo de recopilar información del entorno e interactuar/controlar el mundo físico, sino también de estar interconectados entre sí a través de Internet para intercambiar datos e información.([29])

Hoy en día se ha vuelto cotidiano escucha los terminos *Internet of Things* traducido al español como **internet de las cosas**, la cual se considera la si-

guiente gran evolución del internet, además de ser el origen de los *wearables devices*, es decir, las tecnologías vestibles [33].

En 1999, Neil Gershenfeld en su libro "Cuando las cosas empiezan a pensar" planteó la idea del concepto, pero fue Kevin Ashton, cofundador del Auto-ID Center (MIT), la primera persona en acuñar el término *Internet of Things* en una conferencia para Procter Gamble en 1999, en la cual argumenta que, gracias a la asociación de los objetos físicos con etiquetas RFID, se podría dar identidad a cada objeto para generar datos sobre ellos. ([20])([4]) Tal y como señalan Alexandres Fernández, Rodríguez Morcillo García y Muñoz Frías (2006, p.46): "El objetivo de esta tecnología RFID (Radio Frequency IDentification) es el de identificar objetos de una forma rápida, con poca transmisión de información y en un entorno cercano". ([1])

El gran avance y la versatilidad de esta tecnología han permitido su implementación en casi cualquier área, algunas de estas son: Sector salud, Ganadería, Agricultura, Monitoreo del uso y manejo de recursos energéticos, Smart Cities y Smart Homes, entre otras [33].

3.1.1. Seguridad en IoT

La seguridad en IoT se ha convertido en un tema importante en la actualidad debido al creciente número de dispositivos conectados a la red y la gran cantidad de datos que se manejan en ellos. La seguridad es un aspecto crítico que debe considerarse en todos los dispositivos IoT, ya que pueden estar expuestos a diversos riesgos, como el robo de datos, el control remoto no autorizado, el sabotaje y la interrupción de los servicios. Por lo tanto, es esencial implementar medidas de seguridad adecuadas para proteger los dispositivos IoT y los datos que manejan [33].

Uno de los mayores desafíos en la seguridad de IoT es la gran cantidad de dispositivos y redes involucrados. Cada dispositivo IoT puede tener diferentes sistemas operativos y protocolos de comunicación, lo que hace difícil implementar medidas de seguridad universalmente aceptadas. Además, muchos dispositivos IoT tienen limitaciones en cuanto a la memoria y la capacidad de procesamiento, lo que dificulta la implementación de soluciones de seguridad más avanzadas. Por lo tanto, es importante que las medidas de seguridad sean escalables y adaptables a diferentes dispositivos y redes. ([34])

3.1.2. Plataformas IoT

Las soluciones de IoT se fundamentan en el uso de múltiples dispositivos, conocidos como nodos, los cuales contienen los componentes fundamentales de un ordenador; como CPU, RAM, módulos de comunicación, entre otros; integrados en un dispositivo pequeño y con un rango corto de comunicación [33].

3.2. Esquemas de clave publica

Los esquemas de clave pública son un tipo de criptografía que utiliza un par de claves diferentes para cifrar y descifrar información. En lugar de utilizar una única clave compartida entre las partes que se comunican, en los esquemas de clave pública se utilizan dos claves matemáticamente relacionadas: una clave pública y una clave privada. ([23])

La clave pública es compartida con todas las partes que quieran enviar información cifrada al dueño de la clave privada. La clave privada, en cambio, es mantenida en secreto por el dueño de la clave y se utiliza para descifrar la información recibida cifrada con la clave pública. ([18])

La seguridad de estos esquemas se basa en la dificultad de resolver ciertos problemas matemáticos complejos, como el factorizar grandes números o el problema del logaritmo discreto, que son utilizados para generar las claves y para cifrar y descifrar la información [19].

La elección del esquema de clave pública más adecuado dependerá de las características de la aplicación, como la cantidad de información a cifrar, la velocidad de procesamiento necesaria y las restricciones de recursos de los dispositivos implicados. Además, es importante tener en cuenta la seguridad y la resistencia del esquema a posibles ataques, así como la facilidad de implementación y uso. ([28])

3.3. Protocolos de comunicación

Los protocolos de comunicación son esenciales en la criptografía para establecer canales seguros de comunicación entre dos o más entidades. Un protocolo de comunicación se refiere a un conjunto de reglas y procedimientos que rigen la comunicación entre dos o más dispositivos en una red. Los protocolos de comunicación se utilizan en la criptografía para garantizar la seguridad de la información que se intercambia entre las entidades [19].

Existen varios protocolos de comunicación utilizados en criptografía, algunos de los más comunes son:

- SSL/TLS: Secure Sockets Layer (SSL) y su sucesor, Transport Layer Security (TLS) son protocolos criptográficos ampliamente utilizados para establecer canales seguros de comunicación en Internet. SSL/TLS utiliza certificados digitales para autenticar los servidores y los clientes y utiliza cifrado simétrico y asimétrico para proteger la información que se intercambia entre las entidades.([12])
- SSH: Secure Shell (SSH) es un protocolo criptográfico utilizado para acceder de forma segura a un servidor remoto. SSH utiliza técnicas crip-

tográficas para cifrar el tráfico entre el cliente y el servidor, así como para autenticar la identidad del servidor y del usuario.([25])

- IPsec: Internet Protocol Security (IPsec) es un conjunto de protocolos criptográficos utilizados para proteger el tráfico de red en Internet. IPsec utiliza autenticación, cifrado y encapsulamiento para garantizar la seguridad de la información que se intercambia entre dos o más dispositivos.([37])
- PGP: Pretty Good Privacy (PGP) es un protocolo criptográfico utilizado para cifrar y firmar correos electrónicos y archivos. PGP utiliza cifrado asimétrico y autenticación de clave pública para garantizar la confidencialidad, integridad y autenticidad de la información que se intercambia.([21])
- Kerberos: Kerberos es un protocolo criptográfico utilizado para la autenticación de usuarios en redes de computadoras. Kerberos utiliza un sistema de tickets para autenticar la identidad de los usuarios y garantizar la seguridad de la información que se intercambia entre los usuarios y los servidores.([7])

3.4. Arquitecturas de Red

La arquitectura de red se refiere a la estructura y organización de los dispositivos y servicios de red utilizados para garantizar la seguridad de la comunicación. La seguridad en la red es fundamental para la criptografía, ya que los algoritmos y protocolos de cifrado se utilizan para proteger los datos mientras se transmiten a través de la red. En este artículo, se explorarán algunas arquitecturas de red comúnmente utilizadas en criptografía [19].

4. Estado del Arte

Debido a la creciente importancia que tiene el IoT en el mundo moderno, debemos de prestar gran atención y recursos a la creación e implementación de medidas de seguridad mediante las cuales poder asegurar un medio seguro por el cual los dispositivos inteligentes puedan comunicarse sin exponer la información a agentes externos o maliciosos. Entre las medidas actuales con las cuales se afronta esto podemos encontrar comúnmente algoritmos criptograficos RSA, ECC y AES. Además, otra medida de seguridad importante es el control de acceso, asegurándose de que solo los usuarios y dispositivos autorizados tengan acceso a los recursos de IoT. Esto puede lograrse mediante la implementación de autenticación y autorización adecuadas. La autenticación permite verificar la identidad de los usuarios y dispositivos, mientras que la autorización permite determinar qué recursos pueden acceder. La gestión de claves también es una parte importante de la seguridad en IoT. La gestión adecuada de las claves permite proteger la información confidencial, como las claves de cifrado y las contraseñas. Es esencial asegurarse de que las claves se generen de manera segura y se almacenen de forma segura [33].

Hay diferentes tipos de clave que se pueden utilizar, pero uno de los esquemas más utilizados son aquellos de clave pública, llamados PKI. Estos sistemas permiten el intercambio de información de manera segura, a partir de una clave pública [30]. Los sistemas de PKI incluyen el hardware, software y la criptografía, manejando claves y certificados que permiten la autentificación, confidencialidad e integridad de procesos dentro del IoT. Los esquemas de clave pública protegen información brindando confidencialidad, y por medio de las firmas asegura que los datos no hayan sido manipulados, para garantizar su autenticidad. [29] Entre los esquemas de clave pública más utilizados se encuentran:

- RSA: es uno de los esquemas de clave pública más populares. Se basa en la dificultad de factorizar números grandes en sus factores primos. Fue inventado en 1977 por Ron Rivest, Adi Shamir y Leonard Adleman.[22]
- Diffie-Hellman: es un esquema de intercambio de claves que permite a dos partes establecer una clave compartida utilizando una clave pública compartida. Fue inventado por Whitfield Diffie y Martin Hellman en 1976.[24]
- ElGamal: es un esquema de cifrado de clave pública basado en el problema del logaritmo discreto. Fue inventado por Taher Elgamal en 1985. [14]
- Curvas elípticas: son un tipo de criptografía de clave pública que se basan en la dificultad del problema del logaritmo discreto en una curva elíptica. Son utilizadas en aplicaciones con restricciones de recursos, como dispositivos móviles o sensores IoT. [34]

Pros y Cons de los algoritmos considerados:

- **RSA:**
  - **Pros:** Es ampliamente utilizado en la industria y en la criptografía de clave pública en general debido a su seguridad probada y robustez. Además, es fácil de implementar y entender.
  - **Contras:** A medida que los números clave aumentan en tamaño, el proceso de cifrado y descifrado se vuelve más lento. Además, su seguridad depende en gran medida de la dificultad de factorizar números grandes en sus factores primos, lo que significa que si se descubre un algoritmo que pueda factorizar números grandes con facilidad, la seguridad de RSA se verá comprometida.
- **Diffie-Hellman:**
  - **Pros:** Es ampliamente utilizado en la criptografía de clave pública y se considera un protocolo seguro para el intercambio de claves. Además, su implementación es relativamente sencilla.
  - **Contras:** No ofrece cifrado, solo permite a dos partes establecer una clave compartida, lo que significa que se necesita un algoritmo adicional para el cifrado de datos. Además, se sabe que es vulnerable a los ataques de intermediarios.

- **ElGamal:**
  - **Pros:** Ofrece cifrado y firma digital y se considera un protocolo seguro. Además, permite la generación de claves de forma distribuida, lo que significa que no es necesario que una sola entidad genere todas las claves.
  - **Contras:** El proceso de cifrado y descifrado es más lento en comparación con otros algoritmos de clave pública. Además, es vulnerable a los ataques de clave no válida.
- **Curvas elípticas:**
  - **Pros:** Ofrece una seguridad comparable a la de RSA y otros algoritmos de clave pública, pero utiliza claves más cortas, lo que lo hace adecuado para dispositivos móviles y aplicaciones IoT con recursos limitados. Además, el proceso de cifrado y descifrado es más rápido en comparación con otros algoritmos de clave pública [19].
  - **Contras:** Su implementación y comprensión son más difíciles en comparación con otros algoritmos de clave pública, y no es tan ampliamente utilizado en la industria. Además, es vulnerable a los ataques basados en la elección de curvas inseguras o el uso de parámetros mal elegidos [19].

4.1. Librerías

En cuanto a la implementación de algoritmos criptográficos, es necesario conocer las herramientas que nos permitirán desarrollar este trabajo, específicamente, es necesario conocer los recursos presentes en los distintos lenguajes de programación como lo son las bibliotecas criptográficas disponibles que permiten la implementación de esquemas de clave pública de manera rápida y eficiente. A continuación, se presentan algunas de las bibliotecas más populares:

- OpenSSL: Es una biblioteca criptográfica de código abierto escrita en C que proporciona implementaciones de algoritmos criptográficos de clave pública y simétrica, así como funciones para la creación y gestión de certificados digitales. OpenSSL es ampliamente utilizado en aplicaciones de seguridad en línea, incluyendo servidores web, navegadores y correo electrónico.[31]
- Bouncy Castle: Es una biblioteca de criptografía de clave pública de código abierto escrita en Java que proporciona implementaciones de algoritmos criptográficos de clave pública, incluyendo RSA, DSA, Diffie-Hellman, Elliptic Curve Cryptography (ECC) y más. También ofrece implementaciones de algoritmos de cifrado simétrico y funciones de hash criptográficas.[36]

- PyCrypto: Es una biblioteca criptográfica de código abierto escrita en Python que proporciona implementaciones de algoritmos criptográficos de clave pública y simétrica, incluyendo RSA, DSA, Diffie-Hellman, AES y más. PyCrypto también ofrece funciones para la creación y gestión de certificados digitales.[27]

- Libgcrypt: escrita en C que proporciona implementaciones de algoritmos criptográficos de clave pública y simétrica, incluyendo RSA, DSA, Diffie-Hellman, AES y más. Libgcrypt también ofrece funciones para la creación y gestión de claves criptográficas y la generación de números aleatorios.[26]

- Crypto++: escrita en C++ que proporciona implementaciones de algoritmos criptográficos de clave pública y simétrica, incluyendo RSA, DSA, Diffie-Hellman, AES y más. Crypto++ también ofrece funciones para la creación y gestión de certificados digitales y la generación de números aleatorios.[10]

4.2. Arquitecturas de red

Además de la implementación de algoritmos criptograficos, es importante establecer una arquitectura de red la cual se adecue a las necesidades de nuestro sistema para asi asegurarnos de conocer las distintas formas de acceso por la cuales se podria recibir un ataque o fuga de información [19]. Entra las arquitecturas de red más comunes podemos encontrar:

- Redes de Área Local (LAN): las redes de área local (LAN) son sistemas de red que conectan dispositivos en una ubicación geográfica cercana, como una oficina, un edificio o una institución educativa. Los dispositivos de red se comunican a través de un medio físico, como un cable de Ethernet o una conexión inalámbrica. Para proteger la seguridad de la red, se pueden utilizar tecnologías como el control de acceso a la red (NAC), la autenticación y la encriptación de datos.[11]

- Redes de Área Amplia (WAN): las redes de área amplia (WAN) son sistemas de red que conectan dispositivos a través de una gran área geográfica, como una ciudad, un país o incluso el mundo entero. La seguridad de las WAN puede ser más difícil de proteger debido a la gran cantidad de dispositivos y puntos de conexión que pueden existir en la red. Para proteger la seguridad de la WAN, se pueden utilizar tecnologías como los firewalls, los túneles VPN y los servicios de autenticación de usuario.[15]

- Redes Privadas Virtuales (VPN): las redes privadas virtuales (VPN) son una forma de conectarse a una red privada a través de una red pública, como Internet. La VPN utiliza protocolos de seguridad y cifrado para proteger los datos transmitidos a través de la red pública. Las VPN se utilizan a menudo para permitir a los empleados acceder a la red de la empresa desde ubicaciones remotas de forma segura.[8]

- Redes Inalámbricas: las redes inalámbricas permiten a los dispositivos conectarse a la red sin necesidad de cables físicos. Aunque son convenientes, las redes inalámbricas son más vulnerables a los ataques de seguridad que las redes cableadas. Para proteger la seguridad de las redes inalámbricas, se pueden utilizar tecnologías como la autenticación de usuario, la encriptación de datos y el control de acceso a la red.[32]
- Redes de Acceso a Internet: las redes de acceso a Internet permiten a los dispositivos conectarse a Internet a través de un proveedor de servicios de Internet (ISP). La seguridad en estas redes puede ser difícil de proteger debido a la gran cantidad de dispositivos y usuarios que pueden conectarse a la red. Para proteger la seguridad de las redes de acceso a Internet, se pueden utilizar tecnologías como los firewalls, los servicios de autenticación de usuario y la encriptación de datos.[38]

4.3. Recursos de Hardware

- **Arduino Uno:** es una placa de desarrollo electrónica de código abierto basada en el microcontrolador ATmega328P. Tiene una velocidad de reloj de 16 MHz y una memoria interna de 32KB. La RAM es de 2KB y se puede programar en lenguajes como Java, MicroPython, C/C++.[3]
- **Esp8266:** es un sistema en chip (SoC) de bajo costo con conectividad Wi-Fi. Utiliza un procesador Tensilica Xtensa LX106 con una velocidad de 80 MHz (160 MHz en overclocking). Tiene una memoria interna de 4MB EEPROM y una RAM de 64KB. Se puede programar en lenguajes como Arduino, Lua, MicroPython, C/C++ y Scratch.[35]
- **Raspberry Pi 3:** es una computadora de placa única (SBC) con un procesador Quad Core 1.2GHz Broadcom BCM2837 de 64 bits. No tiene memoria interna, pero tiene un puerto SD para integrar. La RAM es de 1GB y se puede programar en lenguajes como Python y C/C++.[16]
- **Raspberry Pi 4:** es una computadora de placa única (SBC) de alta calidad y rendimiento. Tiene un procesador System on Chip (SoC) Broadcom BCM2711 compuesto por un procesador ARMv8 de cuatro núcleos de 64 bits de 1.5GHz. La memoria interna depende del modelo, y puede ser de 1GB, 2GB, 4GB o 8GB LPDDR4-3200 SDRAM. También tiene un puerto SD para integrar. Se puede programar en lenguajes como Python y C/C++.[17]
- **Nvidia Jetson Nano (Developer Kit):** es un kit de desarrollo para la inteligencia artificial (IA) y el aprendizaje automático (ML) que utiliza un procesador Quad-core ARM® A57. No tiene memoria interna, pero tiene un puerto SD para integrar. La RAM es de

4GB LPDDR4 de 64 bits y se puede programar en lenguajes como Python.[9]

Pros y Contras de cada Hardware considerado:
Arduino Uno

- Pros:
  - Tiene soporte para criptografía a través de librerías de software.
  - Es adecuado para proyectos que requieren una baja complejidad criptográfica.
  - Puede ser utilizado para construir proyectos que requieren alta seguridad física.
- Contras:
  - Limitado por su potencia de procesamiento y memoria, lo que puede hacer que la criptografía sea lenta o imposible en proyectos más complejos.
  - No tiene hardware dedicado para acelerar operaciones criptográficas.
  - No es adecuado para proyectos que requieren una alta complejidad criptográfica o grandes cantidades de datos a cifrar o descifrar.

[9]

Esp8266

- Pros:
  - Tiene soporte para criptografía a través de librerías de software.
  - Puede ser utilizado para proyectos que requieren conectividad inalámbrica y seguridad.
  - Es adecuado para proyectos que requieren una baja complejidad criptográfica.
- Contras:
  - Limitado por su potencia de procesamiento y memoria, lo que puede hacer que la criptografía sea lenta o imposible en proyectos más complejos.
  - No tiene hardware dedicado para acelerar operaciones criptográficas.
  - No es adecuado para proyectos que requieren una alta complejidad criptográfica o grandes cantidades de datos a cifrar o descifrar.

Raspberry Pi 3 y 4

- Pros:

- Tienen soporte para criptografía a través de hardware dedicado para acelerar operaciones criptográficas, como AES.
- Tienen mayor potencia de procesamiento y memoria en comparación con los microcontroladores, lo que los hace adecuados para proyectos que requieren una alta complejidad criptográfica o grandes cantidades de datos a cifrar o descifrar.
- Son adecuados para proyectos que requieren conectividad inalámbrica y seguridad.

- Contras:
  - Pueden ser más costosos en comparación con los microcontroladores.
  - La seguridad física puede ser un problema, ya que son más grandes y menos portátiles que los microcontroladores.

Nvidia Jetson Nano

- Pros:
  - Tiene soporte para criptografía a través de hardware dedicado para acelerar operaciones criptográficas, como AES.
  - Tiene mayor potencia de procesamiento y memoria en comparación con los microcontroladores, lo que lo hace adecuado para proyectos que requieren una alta complejidad criptográfica o grandes cantidades de datos a cifrar o descifrar.
  - Es adecuado para proyectos que requieren inteligencia artificial y aprendizaje automático en combinación con la criptografía.

- Contras:
  - Puede ser más costoso en comparación con los microcontroladores y algunos modelos de Raspberry Pi.
  - La seguridad física puede ser un problema, ya que es más grande y menos portátil que los microcontroladores.
  - Puede ser más difícil de programar y configurar en comparación con los microcontroladores y algunos modelos de Raspberry Pi.

5. Propuesta de solución del reto

La solución que propondremos al socio formador se basará en los siguientes puntos. Primero que nada, la constante actualización de software/firmware de todo el sistema, la utilización de contraseñas seguras y una propuesta de cambio de las mismas cada cierto tiempo, medidas de autenticación con diversos factores cuando se acceda remotamente, limitación del acceso físico, la monitorización del sistema, la realización de auditorías de seguridad cada cierto tiempo y, por

último, asegurar que las políticas de los proveedores se alinean con las nuestras [19].

Siguiendo lo anteriormente expuesto, es crucial destacar la importancia de la información reciente y selecta relacionada con la vulnerabilidad en el firmware, la cual ha ido en aumento cada año según los datos del N.V.D.(National Vulnerability Database). Por tanto, resulta una buena práctica prestar atención a este aspecto, no solo por nuestra seguridad personal, sino también por la responsabilidad que implica proteger a quienes nos rodean.([13])

El equipo decidió utilizar el algoritmo RSA para la criptografía de clave pública por varias razones. En primer lugar, se evaluaron diversos algoritmos de clave pública, incluyendo Diffie-Hellman, ElGamal y curvas elípticas, pero se encontró que RSA ofrecía un equilibrio adecuado entre seguridad y eficiencia [19].

Además, el equipo consideró la naturaleza de las aplicaciones en las que se utilizaría la criptografía de clave pública, y encontró que RSA era especialmente adecuado para dispositivos con restricciones de recursos, como dispositivos móviles o sensores IoT, debido a su amplia adopción y soporte en diversas plataformas [33].

Junto a esto podemos implementar otras soluciones de seguridad como firewalls, sistemas de detección de intrusiones y monitorización de la red, pues la monitorización de la red puede ayudar a detectar posibles amenazas de seguridad y a tomar medidas preventivas para proteger los dispositivos y los datos presentes en nuestra red [19].

Con base en las investigaciones revisadas, podemos llegar a dicha arquitectura ya que fue creada con los diferentes ejemplos que vimos, ya que estos tenían diferentes procesos de encriptación, pero su arquitectura es muy similar, en este caso se tienen a los auditores al principio, que se quiere suponer que ya están cargados de la información, y posterior se sube al internet donde es encriptada, de esta se pasa al centro de control donde se desencripta y llega a las bases de datos. Se decidió como principal opción el uso de una Raspberry debido a su valoración de pros y contras, la clave pública se utilizará con RSA para la firma [19].

5.1. Firma digital con RSA

El Algoritmo de RSA permite el acceso a personas ajenas sin comprometer la seguridad del sistema, su funcionamiento se basa en el producto, conocido, de dos números primos grandes elegidos al azar y mantenidos en secreto.

El proceso de firma digital con RSA puede ser descrito de la siguiente manera:

1. Se seleccionan dos números primos p y q de longitud mayor a 1024 bits y se calcula el producto: $n = pq$
2. Se selecciona la llave para encriptación e, donde e es primo relativo a $(p-1)(q-1)$

3. Calculamos la llave de desencriptado d usando el algoritmo euclidiano de tal manera que d satisfaga

$$ed \equiv 1((p-1)(q-1))$$

Entonces $d = e^{-}1((p-1)(q-1))$, donde d tambien es primo relativo a n

4. e y n son las llaves publicas, mientras que d es la llave privada y por lo tanto debe ser secreta. La llave e es usada para encriptar texto cuando se desee enviarse a otros. Si es destinatario puede desencriptar esta información con la llave d, entonces se prueba que la información proviene de una fuente confiable.[6]

5.2. AWS

Los Amazon Web Services (AWS) es nuble que cuenta con muchos servicios como lo son infraestructuras, almacenamiento, base de datos, tecnologías emergentes, análisis e internet de las cosas entre otras. Tiene la característica de ser flexible y seguro, cuenta con los requerimientos de seguridad del ejército, bancos internacionales acerca de la confidencialidad, igualmente tiene un respaldo de seguridad en la nube, así mismo es compatible con 98 estándares de seguridad y certificaciones de conformidad y 117 servidores que almacenan datos de los cliente. [2]

5.3. EC2 y RDS

Amazon EC2 es un servicio de Amazon Web Services que permite a los usuarios alquilar computadoras virtuales en las que pueden ejecutar sus propias aplicaciones. EC2 es flexible y puede adaptarse a las necesidades de computación, lo que significa que puedes añadir más poder de procesamiento o memoria según lo necesites.

En nuestro proyecto, utilizamos EC2 como la pieza central que facilita el intercambio de datos entre la Raspberry Pi y RDS. Amazon RDS es un servicio de Amazon Web Services que proporciona bases de datos relacionales en la nube. Permite a los usuarios guardar y recuperar cualquier cantidad de datos en cualquier momento y desde cualquier lugar. RDS puede soportar varios motores de bases de datos, incluyendo MySQL, PostgreSQL, MariaDB, Oracle, y SQL Server.

En nuestro proyecto, RDS desempeña el papel de almacén seguro para los datos. RDS es esencialmente donde guardamos nuestros datos decodificados. Una de las ventajas de usar RDS es que ofrece la capacidad de cifrar los datos en reposo, lo que añade una capa adicional de seguridad. Esto significa que los datos están cifrados cuando se almacenan en el disco en su instancia de base de datos. Utilizando AWS Key Management Service (KMS), RDS hace que sea muy fácil crear y controlar las claves utilizadas para cifrar los datos, lo que mejora la seguridad y cumple con los requisitos de cumplimiento.

5.4. Arquitectura Sugerida

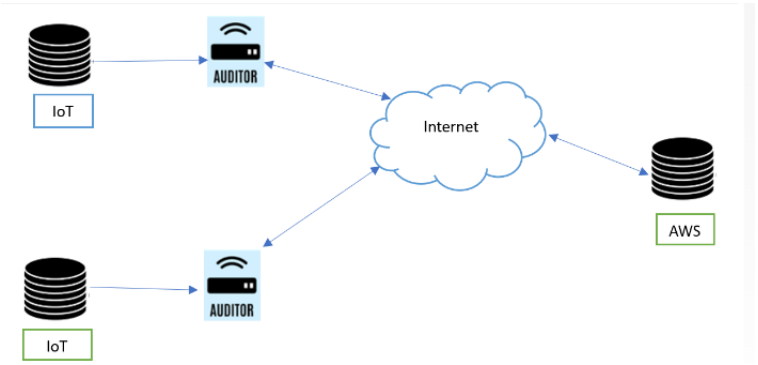


Figura 3: Arquitectura sugerida

Con base a las investigaciones revisada, podemos llegar a la arquitectura mostrada en la Figura 3 ya que fue creada con los diferentes ejemplos que vimos, debido que estos tenían diferentes procesos de encriptación, pero su arquitectura es muy similar, en este caso se especifica el uso de AWS (Amazon Web Services). Se decidió como principal opción el uso de una Raspberry debido a su valoración de pros y contras, la clave púbica se utilizará con las curvas elípticas y para encriptar se utilizará el algoritmo de AES, por su simplicidad en implementación y gran seguridad probada por NIST [19].

5.5. Componentes de la propuesta

Componente	**Herramienta a utilizar**	**Función**
Algoritmo de intercambio de claves	RSA	Algoritmo de clave pública que se utiliza para encriptar los datos y proporcionar autenticidad.
Algoritmo de encriptado	TLS/SSL (a través de HTTPS)	Protocolos de seguridad que proporcionan comunicaciones seguras a través de una red, utilizados para encriptar los datos transmitidos entre el cliente y el servidor.
Hardware	Rasberry Pi 3 y 4	Equipo donde se programará el algoritmo.
Conexión	AWS EC2	Permite alquilar computadoras virtuales en las que pueden ejecutar sus propias aplicaciones.

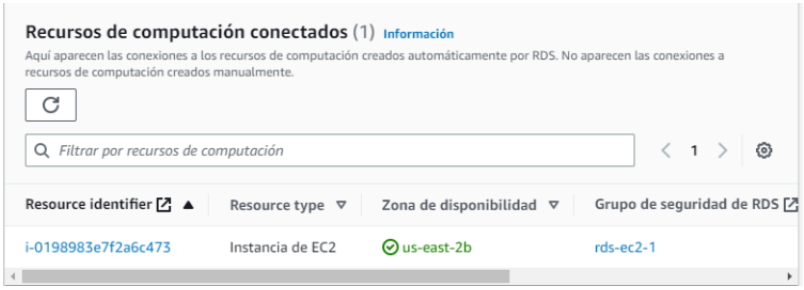
Base de datos	AWS RDS	Servicio de AWS que proporciona bases de datos relacionales en la nube, escalables y con varias opciones de motor de base de datos.

Cuadro 1: Descripción de los componentes a usar en la propuesta, así como las herramientas a usar y su función.

6. Implementación de la propuesta

6.1. Configuración de la base de datos AWS RDS

En esta parte todo es a través de la interfaz gráfica de usuario en la consola de administración de WS, en donde primero se tuvo que seleccionar un motor de base de datos, nosotros escogimos MySQL y se tuvo que configura el tipo e instancia, la capacidad de almacenamiento y establecer las credenciales del administrador.



Se puede observar la consola de AWS RDS con la base de datos ya creada, de puede ver el nombre de la base de datos, la ubicación y el estado de ella.

6.2. Creación de la tabla MeterReadings

La creación de la tabla `MeterReadings` en la base de datos es un paso esencial en nuestro proceso de implementación. Esta tabla almacena las lecturas de los medidores, que son enviadas desde los dispositivos IoT a través de la Raspberry Pi. La tabla se diseñó con los siguientes campos:

- `EntryID`: Un identificador único para cada lectura.
- `ID`: Un identificador para el medidor que tomó la lectura.
- `ConsumptionOrProduction`: Este campo indica si la lectura es un consumo (0) o una producción (1).

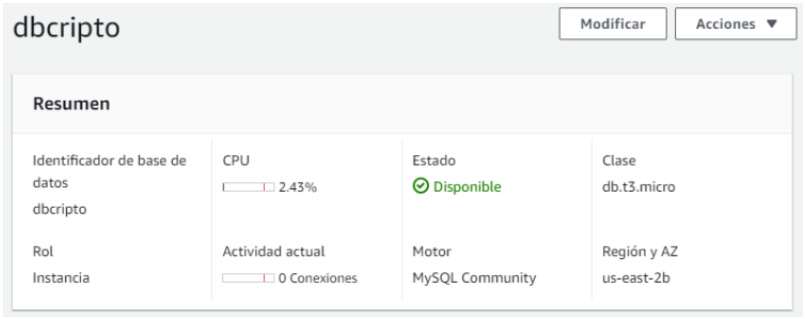


Figura 4:

- `Day`, `Month`, `Year`: Estos campos representan la fecha en la que se tomó la lectura.
- `Time`: Este campo representa la hora del día en la que se tomó la lectura.
- `Reading`: Este campo representa la lectura real tomada por el medidor.

Para crear la tabla, utilizamos la biblioteca `pymysql` en Python, que proporciona una interfaz para conectarse a una base de datos MySQL desde una aplicación Python. Creamos una función `db_connection` que establece una conexión con la base de datos y devuelve un objeto de conexión. Esta función se utiliza para interactuar con la base de datos, ya sea para crear la tabla `MeterReadings` o para insertar datos en ella.

Una vez que la tabla está creada, podemos empezar a enviar datos a ella. Para ello, utilizamos una función `receive_data` que se activa cuando se realiza una petición POST a la ruta '/receive_data'. Esta función recibe los datos en formato JSON, verifica la firma y el hash de los datos, y luego los inserta en la tabla `MeterReadings`.

Es importante destacar que todas las operaciones de base de datos están envueltas en un bloque try/finally para asegurar que la conexión a la base de datos se cierre correctamente, incluso si ocurre un error durante la operación.

En resumen, la creación de la tabla `MeterReadings` y el envío de datos a ella es un proceso que implica la configuración de una conexión a la base de datos, la creación de la tabla, y la implementación de una función para recibir y procesar los datos enviados desde los dispositivos IoT. Este proceso es esencial para el funcionamiento de nuestro sistema, ya que permite almacenar y gestionar las lecturas de los medidores de manera eficiente y segura.

6.3. Configuración del servidor AWS EC2

6.4. Creación del código de la aplicación Flask

Se crea una aplicación Flask que recibe los datos de una ruta POST, la cual es endpoint en una aplicación web que está configurada para aceptar HTTP en-

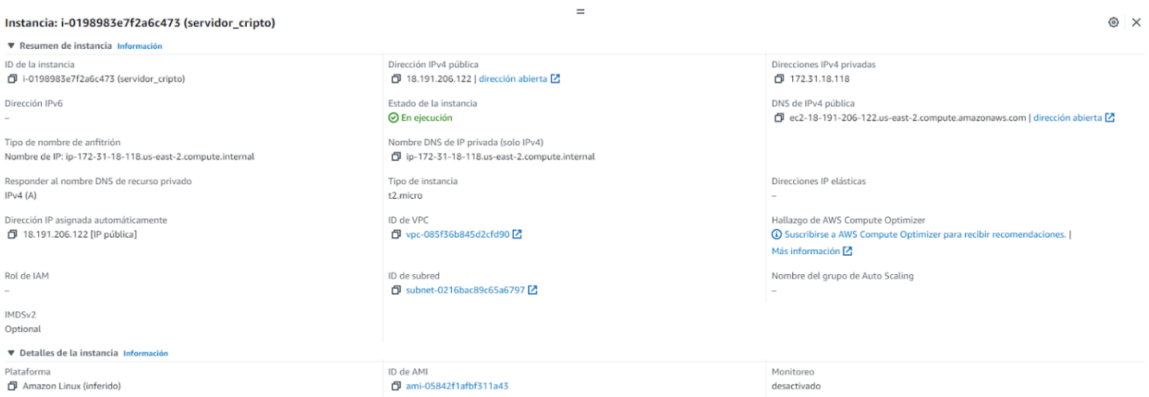


Figura 5:

viadas al servidor y los inserta en la base de datos RDS. El código crea un punto final en $/receive_data$ que acepta peticiones POST. Cada petición debe incluir un cuerpo JSON con los campos EntryID, ID, ConsumptionOrProduction, etc. Una vez recibida la petición, la aplicación se conecta a la base de datos RDS y almacena la información recibida en la tabla MeterReadings.

6.4.1. Pruebas de la aplicación Flask en la instancia EC2

Se generó el certificado SSL(Secure Sockets Layer) autofirmado para el servidor. Este certificado es necesario para permitir conexiones seguras HTTPS a tu aplicación Flask. Durante el proceso, el sistema te pide que introduzcas información que será incorporada en el certificado. Luego, se copio el certificado a la máquina local utilizando SCP. En la máquina local, se puede encontrar el certificado en la ubicación que especifica al ejecutar el comando SCP.

6.5. Generacion de claves (Firma Digital)

La generación de claves es el primer paso en la implementación de la seguridad de los datos. Utilizamos el algoritmo RSA, un algoritmo de criptografía asimétrica, para generar un par de claves: una clave privada y una clave pública. La clave privada se mantiene en secreto y se utiliza para firmar los datos, mientras que la clave pública se comparte y se utiliza para verificar la firma.

El proceso de generación de claves comienza con la creación de una nueva clave privada utilizando el algoritmo RSA. Esta clave privada se guarda en un archivo para su uso posterior. A continuación, obtenemos la clave pública correspondiente a la clave privada y la guardamos en un archivo. Esta clave pública se utilizará en el servidor para verificar la firma de los datos recibidos.

6.6. Transmisión de Datos

Una vez generadas las claves, el siguiente paso es la transmisión segura de los datos. Para ello, utilizamos un servidor basado en Flask que recibe los datos, verifica su integridad y autenticidad, y luego los almacena en una base de datos.

El servidor Flask se configura para recibir datos a través de una ruta específica. Cuando se recibe una solicitud en esta ruta, el servidor extrae los datos de la solicitud. Estos datos incluyen la información real que se está transmitiendo, así como una firma y un hash de los datos.

El servidor verifica primero la integridad de los datos comparando el hash recibido con un hash calculado de los datos recibidos. Si los hashes no coinciden, se rechaza la solicitud, ya que esto indica que los datos pueden haber sido alterados durante la transmisión.

A continuación, el servidor verifica la autenticidad de los datos comprobando la firma. Utiliza la clave pública que se generó anteriormente para verificar la firma de los datos. Si la firma no se verifica, se rechaza la solicitud, ya que esto indica que los datos pueden no haber sido enviados por la entidad que afirma haberlos enviado.

Si tanto la integridad como la autenticidad de los datos se verifican, los datos se consideran seguros y se procede a almacenarlos en la base de datos.

6.7. Envío de Datos desde la Raspberry Pi

En el lado del cliente, utilizamos una Raspberry Pi para enviar los datos al servidor. La Raspberry Pi lee los datos de un archivo CSV, los firma con la clave privada y luego los envía al servidor registro por registro a través de una solicitud HTTPS.

Para firmar los datos, la Raspberry Pi utiliza la clave privada que se generó anteriormente. Crea un hash de los datos y luego firma este hash. La firma y el hash se incluyen en los datos que se envían al servidor.

Este proceso garantiza que los datos enviados al servidor sean tanto íntegros como auténticos. La integridad se garantiza a través del uso del hash, que cambia si los datos se alteran. La autenticidad se garantiza a través del uso de la firma, que sólo puede ser verificada con la clave pública correspondiente a la clave privada utilizada para firmar los datos.

6.8. Preprocesado de los Datos

En el marco de este proyecto, se simula que un dispositivo Raspberry Pi actúa como un medidor IoT. Los datos recogidos por este medidor se almacenan en un archivo CSV. Cada fila de este archivo representa una lectura tomada por el medidor en un momento específico, y cada columna representa un atributo diferente de esa lectura.

El archivo CSV original contenía 96 mediciones por día. Sin embargo, para facilitar la simulación y el envío de datos a la instancia EC2, se transformó el formato de los datos para que cada registro represente una lectura tomada cada

15 minutos. Esto se hizo para que los datos pudieran enviarse línea por línea a la instancia EC2, en lugar de enviar todo el archivo de una vez.

El DataFrame resultante tiene la siguiente estructura:

- **EntryID**: Un identificador único para cada lectura.
- **ID**: Un identificador para el medidor que tomó la lectura.
- **Consumo (0) / Producción (1)**: Esta columna indica si la lectura es un consumo (0) o una producción (1).
- **Dia, Mes, Año**: Estas columnas representan la fecha en la que se tomó la lectura.
- **Hora**: Esta columna representa la hora del día en la que se tomó la lectura.
- **Lectura**: Esta columna representa la lectura real tomada por el medidor.

Este cambio en la estructura de los datos permite una simulación más precisa del funcionamiento del medidor IoT y facilita el envío de los datos a la instancia EC2 para su posterior análisis y procesamiento.

6.9. Interacción Segura con la Base de Datos Remota

Este sistema está diseñado para interactuar de forma segura con una base de datos remota. Con el fin de garantizar la integridad y seguridad de la información, se establece una conexión segura utilizando protocolos de comunicación cifrados.

El primer paso de este proceso consiste en establecer una conexión segura con un servidor remoto. Esto se realiza utilizando un protocolo de red seguro que garantiza que los datos transmitidos entre el servidor y el cliente se mantienen privados y confidenciales.

Una vez que la conexión está establecida, el sistema puede interactuar con la base de datos del servidor. Esta interacción puede incluir una variedad de acciones, como ejecutar consultas en la base de datos para recuperar información. La consulta específica que se ejecuta puede ser personalizada para adaptarse a las necesidades del proyecto.

Los resultados de estas consultas se devuelven al cliente. Estos resultados son entonces procesados y almacenados de manera segura en un formato fácilmente accesible para su análisis posterior. Este proceso permite al sistema interactuar de forma segura con los datos almacenados en el servidor, sin comprometer la privacidad o seguridad de los datos.

Una vez completada la interacción, se cierra la conexión para mantener la integridad del sistema y proteger los datos almacenados en el servidor.

Esta secuencia de acciones forma la base del sistema de interacción segura con la base de datos, que puede adaptarse y ampliarse para satisfacer una variedad de necesidades y escenarios.

7. Referencias

Referencia del github: [5]

Referencias

Alexandres Fernández, S., Rodriguez Morcillo García, C., & Muñoz Frías, J. (2006). RDIF: La tecnología de identificación por radiofrecuencia. *Anales de Mecánica y Electricidad*, *83*(1), 47-52. https://www.iit.upcomillas.es/docs/IIT-05-035A.pdf

Amazon. (2023). ¿Qué es Amazon Web Services? https://aws.amazon.com/es/what-is-aws/

Arduino. (2014). Arduino Uno Rev3. *Arduino*. https://store.arduino.cc/usa/arduino-uno-rev3

Ashton, K. (2009). That 'internet of things' thing. *RFiD*, *22*(7), 97-114. https://www.rfidjournal.com/articles/view?4986

Bravo, A., Vargas, R., González, N. M., González, F., Calderón, F. S., & Herzberg, G. (2023). https://github.com/nath084/MA2006B-E01

Cao Y., F. C. (2008). An Efficient Implementation of RSA Digital Signature Algorithm.

Ciberseguridad.com. (2020). ¿Qué es Kerberos y cómo funciona? *Ciberseguridad.com — Noticias sobre ciberseguridad e informática*. https://ciberseguridad.com/guias/prevencion-proteccion/kerberos/

Cisco. (2023). ¿Qué es una VPN? - Red privada virtual. https://www.cisco.com/c/es_mx/products/security/vpn-endpoint-security-clients/what-is-vpn.html

Corporation, N. (2020). NVIDIA Jetson Nano Developer Kit User Guide. *Nvidia Corporation*. https://developer.nvidia.com/embedded/dlc/jetson-nano-dev-kit-user-guide

Dai, W. (2019). Crypto++ Library 8.2 – a Free C++ Class Library of Cryptographic Schemes. *Crypto++*. https://www.cryptopp.com/

Diana, H. (2021). Red de área local o LAN. https://www.computerweekly.com/es/definicion/Red-de-area-local-o-LAN

DigiCert. (n.d.). ¿Qué es SSL, TLS y HTTPS? *Web Security DigiCert*. https://www.websecurity.digicert.com/es/es/security-topics/what-is-ssl-tls-https

, E. (2020, 2 de abril). *Enterprise Best Practices for Firmware Updates - Eclypsium*. https://eclypsium.com/2020/04/02/enterprise-best-practices-for-firmware-updates/

ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE transactions on information theory*, *31*(4), 469-472.

European Knowledge Center for Information Technology. (2018). Red de área amplia (WAN). https://www.ticportal.es/glosario-tic/wan-red-area-amplia

Foundation, R. P. (2016). Raspberry Pi 3 Model B. *Raspberry Pi Foundation*. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

Foundation, R. P. (2019). Raspberry Pi 4 Model B. *Raspberry Pi Foundation*. https://www.raspberrypi.org/products/raspberry-pi-4-model-b/

Gómez, J. S. (2004). Criptografia de clave pública: el sistema RSA. *Sigma: revista de matemáticas= matematika aldizkaria*, (25), 149-165.

Gómez, W. (2019). ¿Qué es el algoritmo de firma ECDSA? *Academy Bit2me*.

Haller, S. (2010). The internet of things. *Poster at the Internet of Things Conference (IoT 2010), Tokyo*. https://iot-conference.org/iot2010/

HardZone. (2019). Qué es PGP: definición, características y utilidades. *HardZone : Hardware Reviews y Noticias de Hardware en Español*. https://hardzone.es/reportajes/que-es/pgp-pretty-good-privacy/

Hernández Encinas, L. (2005). El Criptosistema RSA. http://hdl.handle.net/10261/15961

IBM. (2022). Criptografía de clave pública.

KeepCoding. (2022). ¿Qué es el algoritmo Diffie-Hellman? https://keepcoding.io/blog/que-es-el-algoritmo-diffie-hellman/

Kinsta. (2021). TLS vs SSL: ¿Cuál es la diferencia? ¿Cuál debería usar? *Kinsta® Hosting WordPress Administrado Premium y Soluciones de Velocidad para WordPress*. https://kinsta.com/es/base-de-conocimiento/tls-vs-ssl/

Koch, W. (2020). Libgcrypt – General purpose cryptographic library based on the code from GnuPG. *GnuPG*. https://gnupg.org/software/libgcrypt/index.html

Litzenberger, D. C. (2013). PyCrypto – The Python Cryptography Toolkit. *PyPI*. https://pypi.org/project/pycrypto/

López Garzón, J. R. (2015). Infraestructura de clave pública (PKI) para una PYME.

Mauricio, A. M. C., et al. (s.f.). EL ESTADO DEL ARTE SOBRE EL INTERNET DE LAS COSAS. AMENAZAS Y VULNERABILIDADES DE SEGURIDAD INFORMÁTICA EVIDENCIADAS DESDE LA DOMOTICA.

Morales, Y. U., & Garcıa, A. O. (2020). PKI-based trust scheme for the exchange of electronic clinical information in the XAVIA HIS system. *Revista Cubana de Informática Médica*, *12*(2).

OpenSSL Project. (2021). OpenSSL: The open source toolkit for SSL/TLS. *OpenSSL*. https://www.openssl.org/

Roberto, R. (2021). Medios de Transmisión para redes inalámbricas. *Universidad Autónoma de Coahuila*, 1-3.

SALAZAR, J., & SILVESTRE, S. (2017). INTERNET DE LAS COSAS. *České vysoké učení technické v Praze Fakulta elektrotechnická*.

Sánchez Martın, J. Á. (2021). Solución de cifrado para dispositivos de bajas capacidades.

Systems, E. (2021). ESP8266 Series of SoCs. *Espressif Systems*. https://www.espressif.com/en/products/socs/esp8266

The Legion of the Bouncy Castle Inc. (2021). Bouncy Castle Crypto APIs. *Bouncy Castle*. https://www.bouncycastle.org/

Universidad Nacional Autónoma de México. (n.d.). El Cifrado Web (SSL/TLS). *Revista .Seguridad - UNAM*. https://revista.seguridad.unam.mx/numero-10/el-cifrado-web-ssltls

Vintró, J. (2005). Redes y acceso a internet. *ELSEVIER*. https://www.elsevier.es/es-revista-offarm-4-articulo-redes-acceso-internet-13078590

8. Tabla de Recursos

	Nombres	Autores	Fecha	Keywords
1	Criptografía de clave pública: El sistema RSA	Josu Sangroniz Gómez	Noviembre 2004	-
2	Sustainable Energy, Grids and Networks(Standardization of the Distributed Ledger Technology cybersecurity stack for power and energy applications)	Sri Nikhil Gupta Gourisetti, Ümit Cali,Kim-Kwang Raymond Choo	2021	Distributed Ledger Technology (DLT) ,Blockchain, Power systems, Distributed energy resource (DER), Cybersecurity, Smart grid
3	Cybersecurity for Distributed Energy Resources and Smart Inverts	Junjian Qi , Adam Hahn , Xiaonan Lu, Jianhui Wang,Chen-Ching Liu	2016	Energy Systems Division, Argonne National Laboratory
4	El Estado del arte sobre el internet de las cosas. Amenazas y vulnerabilidades de seguridad informática evidencias desde la domotica	Cesar Mauricio Acosta Molina	2019	LoT,Actualida, Futuro
5	PKI-based Trust Scheme for Exchange of Electronic Clinical Information in the XAVIA HIS System	Yanssel Urquijo Morales, Arturo Orellana García	Diciembre 2020	Firma digital; PKI; Sector de salud; Seguridad de la información; XAVIA HIS
6	Infraestructura de clave pública (PKI) para una PYME	José Rodolfo López Garzón	2015	Criptografía de Datos, Control de acceso, Certificado digital, infraestructura de clave pública, pequeña empresa
7	Solución de cifrado para dispositivos de bajas capacidades	José Ángel Sánchez Martín	2021	ECC, WSN, cifrado, seguridad, capacidades, limitaciones, información, claves.

8	Propuesta y Análisis de Criptosistemas de Clave Pública Basados en Matrices Triangulares Superiores por Bloques	José Francisco Vicent Francés	2007	Criptosistemas, cifrado, intercambio,clave

Cuadro 2: Tabla mostrando las fuentes y papers utilizados