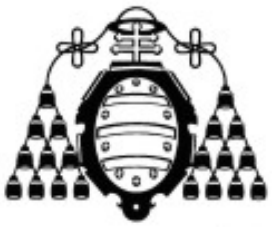
UNIVERSIDAD DE OVIEDO



MÁSTER EN INGENIERÍA WEB

CENTRO INTERNACIONAL DE POSTGRADO

TRABAJO FIN DE MÁSTER

Seguridad en la interconexión de Smart Objects en el marco de Internet of Things

AUTOR: Gonzalo Sánchez Arias
DIRECTORES: B. Cristina Pelayo García-Bustelo

Resumen

Los Smart Objects, objetos físicos con inteligencia añadida, ya forman parte de nuestra vida cotidiana, ya que algunos de ellos como los smartphones, las tablets o las Smart TVs son de uso común. Además, cada vez más objetos incluyen la capacidad de conectarse a la red, de recabar información y de compartirla, abriendo el camino hacia un mundo de objetos conectados dónde se podrán automatizar gran parte de las tareas de nuestra vida cotidiana.

Esta automatización podría realizarse en tareas tan sencillas como hacer la compra, dónde un Smartphone podría conectarse a una nevera inteligente para consultar las existencias y analizar nuestros ámbitos alimenticios. O bien, en tareas más complejas y a nivel de negocio, dónde una compañía aseguradora podría consultar los sensores de un vehículo para comprobar realmente como se conduce dicho automóvil, y dónde los pacientes con enfermedades crónicas podrían contar con dispositivos que monitorizaran su estado, permitiendo prever así empeoramientos del estado de salud y contabilizar hospitalizaciones no previstas.

Esta visión, de objetos conectados facilitando nuestras tareas cotidianas, requiere que dichos objetos compartan y envíen información personal de todo tipo, tan irrelevante como los alimentos que hay en nuestra nevera, o más confidencial como nuestro estado de salud, nuestra localización, rutas diarias o cuentas bancarias, vulnerando así nuestra privacidad.

El estudio que plantea la presente investigación, se centra en la comunicación entre Smart Objects en una plataforma de Internet of Things, proponiendo medidas de seguridad que permitan el envío de un mensaje de forma segura en un entorno no seguro. Estas medidas, extraídas de la seguridad tradicional, se centran en la criptografía, creando así comunicaciones entre nodos mediante el manejo de claves y tratando de asegurar que el mensaje sólo será leído por su destinatario.

Internet of Things es la interconexión de objetos heterogéneos y ubicuos, y debido a la heterogeneidad de estos objetos, no todos van a gozar de las mismas prestaciones y posibilidades a la hora de establecer una comunicación, ya que sus capacidades de procesamiento serán diferentes y las tareas de encriptado/desencriptado supondrán un tiempo extra de ralentización de dichas comunicaciones. Por ello, se presenta un estudio en el que, se trata de aunar las mejores soluciones en términos de costo y rendimiento para que el consumo extra sea el menor posible. Para ello se emplean tres dispositivos móviles, de gama alta, media y baja, con el fin observar las diferencias en función de las prestaciones y poder concluir el mejor resultado.

Palabras Clave

Internet of Things; Seguridad; Criptografía; Computación ubícua; Smart Objects;

Abstract

The Smart Objects, physical objects with added intelligence, are already part of our daily life, as some of them as smartphones, tablets or smart TVs are commonly used. In addition, more and more objects include the ability to connect to the network, to gather information and share it, opening the way for a connected world where they can automate many of the tasks of everyday life.

This automation could be done in such simple tasks such as shopping, where a Smartphone could be connected to an intelligent fridge to consult stocks and analyze our eating areas. Or, in more complex tasks and business level, where an insurance company could see a vehicle sensors to really see how that car is driven, and where patients with chronic diseases could have devices that monitored their status, allowing provide information about worsening health status and alert about unforeseen hospitalizations.

This vision of connected objects to facilitate our daily tasks, requires that these objects share and submit personal information of all kinds, as irrelevant as the food in our fridge, or more sensitive as our health, our location, daily routes or bank accounts, thereby violating our privacy.

The study raised in this research focuses on communication between Smart Objects on a platform of Internet of Things, proposing security measures that allow sending a message safely in an unsafe environment. These measures, taken from traditional security, focus on cryptography, creating communication between nodes by managing keys and trying to ensure that the message will only be read by its recipient.

Internet of Things is the interconnection of heterogeneous and ubiquitous objects, due to the heterogeneity of these objects, not all will enjoy the same benefits and possibilities to establish a communication, since their processing capabilities are different and the encryption/decryption tasks will overtime mean a slowdown of such communications. Therefore, we present a study in which it is to combine the best solutions in terms of cost and performance for that the extra consumption as low as possible. To do this we use three mobile devices, high, medium and low range, to observe differences in function of the performance and to conclude the best result.

Keywords

Internet of Things; Security; Cryptography; Ubiquitous Computing; Smart Objects;

Índice General

Índice de Figuras

Introducción

En la actualidad, el auge de los Smart Objects [1] ha promovido un cambio de aptitudes en nuestras vidas cotidianas, incluyendo todo tipo de objetos como los smartphones, las tables, las Smart TVs o las SmartBands, que son ya de uso cotidiano. Estos objetos abren la puerta a múltiples posibilidades, ya que poseen sensores que les permite extraer información de sí mismos y del mundo que les rodea y gozan de conexión a Internet para transmitir dicha información. Además, esta evolución e incluso el abaratamiento de los sensores [2] y su desarrollo de bajo coste [3], que han aumentado su uso [4], han hecho posibles que todo tipo de objetos, de uso tan habitual como una nevera [5], incluyan sensores y la capacidad de conectarse para comprobar su estado.

Esto, unido a la revolución en el uso de Internet, de la computación en la nube [1] y al tratamiento de grandes cantidades de información mediante Big Data, han hecho posible la evolución en Internet of Things [6]. IoT es la interconexión de objetos heterogéneos y ubicuos a través de Internet [6–8]. El objetivo de IoT es tener sensores dispersos por cualquier parte del mundo con el objetivo de recibir información, ya sea en los objetos de nuestra casa, en el trabajo, en nuestro automóvil o en el fondo del mar [7]. Esta conexión entre sensores y transmisión del a información dota a la red de ciertas capacidades como la inteligencia y la capacidad de reacción, actuando en consecuencia de unos datos recibidos, que pueden llevarse a cabo mediante los objetos. Dichos objetos, o Smart Objects, son objetos físicos capaces de procesar información y comunicarse con otros dispositivos para intercambiar información y actuar en función de parámetros determinados [9].

Esos objetos pueden explorar su entorno, comunicarse con otros objetos o con los seres humanos, ayudándoles a realizar sus tareas de nuevas formas novedosas e intuitivas [10]. Por ejemplo sensores depositados en infraestructuras pueden informar sobre condiciones ambientales o detectar individuos adentrándose en zonas restringidas, para actuar en consecuencia [4]. Otros podrían situarse en lugares más remotos, de difícil acceso o totalmente inaccesibles, permitiendo no se recuperados, con el objetivo de dar datos de especial relevancia que puedan ayudar a prevenir desastres [11] o que tengan especial relevancia en sectores peligrosos como el químico o el nuclear [2].

De este modo, se pueden conectar todo tipo de nodos entre sí, en aplicaciones de ámbito tan dispar que van desde lo deportivo a lo comercial, y que pueden realizar tareas tan generales como comprobar el nivel de un termostato o tan personales y privadas como reportar información sobre nuestra salud, nuestras rutas habituales, nuestras cuentas bancarias o nuestra localización, entre otras. Estas aplicaciones, que son sólo un pequeño reflejo de toda la variedad de posibilidades que puede depararnos el futuro en ámbitos cotidianos [6], aprovechan la tecnología disponible y tratan de facilitar nuestra vida cotidiana [6,10] siendo capaces de monitorizar datos de diversa índole y permitiéndonos actuar en consecuencia o respondiendo a las necesidades existentes de forma automática.

Esta automatización permite surgir conceptos como las Smart Home [2,5,12] o las Smart Cities [13], respaldadas por la evolución en IoT y la red de nodos heterogéneos conectados

intercambiando información que surge y que se espera del futuro [14]. Esto puede ir mucho más allá, y además de automatizar tareas para ayudarnos diariamente, pueden desempeñar funciones que nos ayuden a otro nivel, como en el aprendizaje o que estén pendientes de nuestra salud.

En [15], disponen SmartBands en las muñecas de una serie de usuarios durante la narración de una historia. Durante dicha narración van surgiendo una serie de argumentos o escenas y el usuario tiene que ir interactuando y tomando sus propias decisiones sobre la historia. El objetivo final es recopilar datos sobre las sensaciones del usuario durante cada decisión para realizar un análisis cuantitativo. La recopilación de la información se hace utilizando las SmartBands que, en función de los cambios electrodérmicos en la piel del alumno, envían dicha información a otros objetos para ser tratadas y comprobar estados anímicos como la excitación, concentración o sorpresa. Dichas pulseras podrían ser aplicadas a muchos contextos e informarnos por ejemplo de la concentración de un trabajador en su puesto de trabajo o de un alumno durante una clase teórica.

En [16] se presentan los servicios necesarios para cumplir con los requisitos de personas frágiles y de avanzada edad para mantener el contacto con cuidadores mediante la comunicación inalámbrica y el almacenamiento de datos en la nube. Otro uso podría ser el de monitorizar a personas con enfermedades crónicas [4]. No obstante, debido al ámbito de la aplicación y los datos que se hace necesario enviar, se plantean cuestiones de privacidad, lo que repercute en la necesidad de utilizar determinados sistemas de seguridad.

Motivación

Internet of Things parece una tendencia del futuro, que puede mejorar nuestra calidad de vida en varios aspectos, cuyo impacto ha ganada ya tanta relevancia que se considera una de las seis tecnologías con mayor impacto en los Estados Unidos hasta 2025 [17] y cuya importancia fue resaltada por la Organicación de las Naciones Unidas (ONU) en 2005 [6].

Este futuro de objetos heterogéneos conectados [14] cambiará el sentido de las comunicaciones, dónde ya no sólo serán de un humano a una máquina, sino que también las habrá entre diferentes máquinas (M2M). Toda esta interconexión nos va a permitir estar continuamente conectados, compartiendo información personal y relevante a las tareas que desempeñen nuestros objetos, ya sea a nivel doméstico, de negocios o salud, por ejemplo. Además, los usuarios tienen por lo general los dispositivos móviles siempre encendidos, incluso por la noche, lo que significa que numerosos sensores también están presentes con el usuario revelando información de donde está y lo que están haciendo [18].

El carácter de las aplicaciones que realizan este tipo de conexiones y los mensajes de índole personal con los que trabajan comprometen nuestra información privada y hacen imprescindible tomar medidas de seguridad que garanticen una comunicación segura sin que la intercepción de un mensaje, por un usuario malintencionado, comprometa nuestra privacidad. Estas medidas de seguridad pueden ser tomadas de muchas maneras, ya que el carácter novedoso de IoT requiere que se revise su seguridad y se tengan en cuenta puntos que antes podrían haber pasado desapercibidos [19]. Estos pueden ser vulnerabilidades tan simples como la fragilidad a un ataque físico por el carácter individual o solitario de sus nodos

permaneciendo sin vigilancia, o más técnico, basándose en la comunicación inalámbrica para espiar el canal de comunicación [6].

Finalidad del Proyecto

Debido al constante envío de información personal, en este trabajo nos vamos a centrar en la comunicación entre objetos, investigando el envío de mensajes seguros a través de un canal inseguro. Para ello buscaremos las medidas necesarias que puedan impedir que los mensajes intercambiados por los nodos en una red IoT sean vulnerados por usuarios malintencionados y así impedir que puedan leer nuestra información personal a pesar de que sean capaces de interceptarlos.

Debido a la capacidad de procesamiento de la información, de conexión a Internet e incorporación de sensores, el objetivo es garantizar la seguridad en la comunicación entre los nodos lectores de la IoT, que en este caso son los Smart Objects, y no en la seguridad en la comunicación del propio sensor o tarjeta con el dispositivo encargado de realizar su lectura, cómo son estudiados en [20–22]. Además, el objetivo es garantizar una comunicación segura, en al que sólo el destinatario pueda leer el mensaje enviado, y garantizando que ha sido enviados por un nodo seguro, en una comunicación realizada en un canal no seguro.

Para ello utilizaremos técnicas criptográficas extraídas de la seguridad tradicional, con el objetivo de encriptar el mensaje enviado en la comunicación y que sólo pueda ser leído por el destinatario del mismo, garantizando su origen y la integridad del mensaje. Por ello utilizaremos las ventajas proporcionadas por la criptografía para realizarlo, a diferencia de otras opciones basadas en capacidades como se presentan en [23,24–26] que nos obligarían a llevar un control de las capacidades para lograr la autenticación y el control de acceso y requerirían mayor capacidad de almacenamiento de los nodos tan diferentes en cada caso, descuidando además otros aspectos como la integridad del mensaje. Por ello, evaluaremos diferentes tipos de criptografía y diferentes algoritmos, a fin de conseguir la mejor combinación en relación entre la calidad y su coste.

En primer lugar las técnicas serán evaluadas dependiendo del tipo de criptografía empleada. Se analizaran la criptografía de clave secreta o privada, clave pública o asimétrica, e híbrida. Posteriormente, y dentro de cada uno de los tipos, se estudiarán los algoritmos aplicables con el objetivo de determinar la mejor combinación, ya que estás técnicas no fueron pensadas para su utilización en IoT [27]. Por esto, su procesamiento y consumo pueden ser un impedimento a la hora de usarlas en algunos dispositivos. Estas medidas de seguridad tendrán que ser aplicables al intercambio de mensajes entre los objetos interconectados en IoT y, por ello, serán probadas en la plataforma Midgar [7], que se utilizará como caso de estudio. Dicha plataforma, genera aplicaciones que interconectan objetos heterogéneos entre sí, recabando información de ellos y haciéndolos actuar en consecuencia, en un marco en el que la seguridad no es nada desdeñable.

Fijación de Objetivos

El futuro que nos parece deparar Internet of Things y la evolución paralela que sufre la tecnología y que pone a nuestra disposición cada vez más posibilidades, hará que la seguridad de IoT se vuelva fundamental. Por ello, se hace necesario buscar las mejores soluciones para evitar comprometer los datos sensibles y privados de las personas conectadas y así no exponer toda su información privada.

La escasa capacidad de procesamiento de ciertos dispositivos utilizados en IoT, que se caracterizan por bajas capacidades en términos de energía y recursos de computación [6], hace que se comprometa la seguridad en la comunicación, haciendo a menudo de esta un punto es débil [19]. Algunos dispositivos involucrados en la comunicación son componentes pasivos que no pueden implementar esquemas complejos de apoyo a la seguridad [6], como las SmartTags. Según [19], como la información se transmite a través de Internet los ataques son un problema grave a tratar ya que podría destruir la disponibilidad de la comunicación, por lo que hay que mejorar los mecanismos y centrarse en medidas preventivas [19,28].

Objetivos

Por esta razón, se hace necesario investigar un método de envío seguro de los datos para mantener la privacidad de estos y dar cierta seguridad a los Smart Objects en IoT. En este apartado hablaremos de las medidas preventivas de seguridad existentes en otros ámbitos y que pueden ser tomadas en Internet of Things para lograr el envío de mensajes seguros a través de un entorno no seguro.

Confidencialidad

La confidencialidad asegura que un mensaje no pueda ser entendido por un usuario o máquina al que no va dirigido [3], por lo que sólo el destinatario original del mensaje será capaz de leerlo, evitando así, que la intercepción del mismo por un nodo malintencionado comprometa la privacidad de los usuarios o ponga en peligro las comunicaciones.

La criptografía es la principal forma de ocultar información para lograr confidencialidad [29], ya que, mediante las distintas técnicas que ofrece, garantiza que sólo la persona que tenga la clave será capaz de leer el mensaje, por lo que si un nodo cualquier capta un mensaje que no va dirigido a él, no podrá acceder a su contenido. Si no reenvía el mensaje se interrumpirán las comunicaciones, pero la información no será filtrada. La criptografía es una de las formas existentes de garantizar que sólo el personal autorizado pueda acceder a esa información, lleva existiendo desde el renacimiento dónde fue empleada por Leonardo Da Vinci, que escribía sus anotaciones en sentido inverso para que sólo pudiesen ser leídas frente a un espejo [30], una forma sencilla pero que cumplía su cometido. En la criptografía, los mensaje se encriptan, para que sólo se pueda recuperar el mensaje original mediante el uso de la

misma clave que lo encriptó, ya que su descubrimiento por fuerza bruta es muy costoso [31] y más, conforme aumenta el tamaño de la clave empleada.

Autenticación

Otro punto importante de la seguridad es la autenticación, para poder garantizar que el envío se realiza entre nodos registrados o autorizados dentro de la red, y que se puede establecer una comunicación de forma segura, garantizando que no recibimos un mensaje enviado por un nodo malintencionado. También, debido al tipo de la red que se espera de IoT y a las conexiones entre objetos heterogéneos que crecerán en el futuro y que establecen conexiones M2M [32], sin necesidad de la intervención de personas, hace necesario que los nodos puedan autenticarse de forma sencilla y automática, por ejemplo en el envío del mensaje.

Por ello, para permitir el paso de mensajes seguros entre diferentes objetos o personas, se necesita asegurar la **autenticidad** en los sensores [19], para garantizar que están autorizados a intervenir en el tráfico de dicha red [3] y que un nodo malicioso no puede hacerse pasar por un nodo de red de confianza [29]. Así se puede lograr que los objetos conectados a la red se identifiquen de manera única e inequívoca y no puedan ser suplantados por terceros. La autenticación es difícil, ya que requiere infraestructuras de autenticación apropiadas y servidores que logran su objetivo mediante el intercambio de mensajes apropiados con otros nodos [6]. En el contexto de la IoT no sólo los usuarios, sino también los objetos autorizados pueden acceder a los datos. Por ello es necesario definir un mecanismo de control de acceso y un proceso de autenticación [33], que en este caso deberá ser sencillo y poco costoso para que los objetos heterogéneos de capacidades diferentes puedan llevarlo a cabo sin que ello suponga un consumo extra no asumible.

Como no todos tienen las mismas capacidades de almacenamiento ni procesamiento, por lo que no todos gozan de las mismas condiciones a la hora de implantar una tecnología. Los sistemas RFID tienen recursos muy limitados con lo que se hace imposible la generación y procesamiento de claves para la autenticación en ellas [6]. Estas etiquetas son principalmente pasivas, y se limitan a responder a las preguntas del objeto lector encargado de tratar con ellas. El problema está en que un atacante puede interceptar los datos enviados de una etiqueta a un lector autorizado [6] o bien leer una tarjeta que no contenga los datos encriptados. Según [6], las principales soluciones a este problema pasan por la autenticación de los lectores autorizados. Sin embargo involucran etiquetas capaces de autenticarse y escenarios de autenticación con infraestructuras apropiadas y servidores para realizar intercambio de mensajes con los nodos, dónde los costes son altos y lejanos a los escenarios que se esperan de la IoT.

Integridad

Una parte fundamental cuando se requiere transmitir un mensaje seguro es su **integridad**. La integridad es la verificación de que el mensaje es exactamente idéntico a cuando se envió, con los mismos valores y propiedades, pues es así como debe recibirlo el destinatario final. De esta manera se puede comprobar que el mensaje no haya sido modificado por ninguna tercera persona que pudiese interceptarlo y reenviarlo. Las soluciones de integridad de datos deben

garantizar que un adversario no puede modificar los datos o bien que el destinatario final pueda verificar si el mensaje original ha sido modificado para descartarlo, dicho problema ha sido ampliamente estudiado en todos los sistemas informáticos y de comunicación tradicionales [6,34]. Una forma típica y sencilla de garantizar la integridad es mediante el envío de un mensaje hash para realizar la comprobación en el destino [29].

No repudio

Tal y como se ha comentado, la autenticación es necesaria en las comunicaciones para garantizar que los mensajes han sido enviados y recibidos por nodos seguros y no por nodos malintencionados, por ello es necesario que un nodo emisor asegure que ha enviado un mensaje y que un nodo receptor no pueda negar haberlo recibido, para evitar que los mensajes falsos puedan propagarse.

Por ello, es necesario implementar el **no repudio**. Un nodo no puede negar el envío de un mensaje [3] y así, un nodo también puede garantizar que el mensaje no es falso [29]. De esta manera, se logra que no se pueda negar una comunicación que se haya realizado. El emisor no puede negar que haya enviado un mensaje y el receptor no puede negar haberlo recibido, garantizando así la autoría de un mensaje.

Privacidad

Finalmente, el otro objetivo principal de la investigación en Smart Objects es la **privacidad**, buscando adoptar mecanismos que protejan la privacidad de los seres humanos y de los objetos conectados [19]. La investigación de [19] remarca, que en un entorno tan automatizado las personas pueden no llegar a ser del todo conscientes de la información expuesta o de hasta qué punto su privacidad puede ser vulnerada debido al uso de sensores sin conocer de verdad sus limitaciones. Ya que, como se ha mencionado las personas tienen sus dispositivos móviles encendidos durante todo el día, recabando información sobre si mismos y compartiéndola en la red [18]. Además, información personal puede ser peligrosa en caso de caer en manos malintencionadas, como un número de cuenta.

Por esto, la privacidad se vuelve en uno de los problemas principales de IoT [6,19,35]. Según [35], la naturaleza de IoT y su explotación en sensores ha creado entidades “Gran Hermano” que estudian a los usuarios sin su consentimiento, y que podrían crear un entorno en el que estuviera al alcance de los usuarios la información respecto a otros e incluso lo relativo al medio ambiente y otras gestiones automatizadas. Ya que hay todo tipo de nodos, en lugares incluso inaccesibles, informando sobre características medioambientales, o información privada de empresas, que podría utilizarse de forma malintencionada.

Un entorno que interconecte objetos heterogéneos debería tratar información encriptada de un modo seguro, en el que los usuarios no autorizados no pudiesen leerla, aún en el caso de interceptarla y hacerse con ella. Nosotros en esta investigación, proponemos una posible solución.

Ámbitos de Aplicación

Los posibles ámbitos de aplicación serían todos aquellos propios de las redes IoT en los que se interconectasen objetos, por lo tanto no habría un ámbito específico y concreto. Algunos de ellos podrían clasificarse en los siguientes niveles:

- **Doméstico**: Surgen términos como el Smart Home, dónde las personas podrían interactuar con diferentes objetos de su casa, como neveras inteligentes, calefacción o ventiladores. Automatizando tareas y simplificando el uso de muchos electrodomésticos. Por ejemplo una nevera inteligente podría informarte cuando hubiese déficit de algún producto o estuviese próximo a acabarse. También podría programarse los ventiladores para que se encendiesen cuando la temperatura ambiente, leída por un termostato, superase una determinada temperatura.
- **Privado**: A nivel de negocios, dónde las empresas podrían realizar varias acciones dependiendo de su ámbito o de las necesidades requeridas. Por ejemplo una empresa podría situar lectores en puntos alejados para obtener información necesaria que, de otra forma tendría que ser revisada, del mismo modo podrían tomarse valores peligrosos o nocivos en industrias como la química. Una compañía de seguros podría acceder a los sensores de un automóvil para comprobar cómo se conduce y si entraña un peligro o si las consecuencias son mitigables. También se podría comprobar el ciclo de vida de los productos en una producción.
- **Personal**: Entre dispositivos móviles, dónde se programen notificaciones o avisos dependiendo de determinados factores y se recolecte información de carácter personal.

En estos campos, los mensajes intercambiados, deberían ser privados y accesibles sólo por los dispositivos destino, ya que de ser capturados comprometerían nuestra privacidad.

Estado Actual de los Conocimientos Científico-Técnicos

Dentro de la visión de Internet of Things, tenemos innumerables objetos conectados a Internet, transmitiendo información desde cualquier parte del mundo y conectándose entre sí. Como expresan en [10] esta visión es tan futurista como inquietante, pues todos nuestros movimientos, acciones y decisiones son grabadas continuamente por dispositivos electrónicos que se pueden encontrar en cualquier lugar de nuestra casa, como en la cocina o en el salón, o bien ser dispositivos que usemos en nuestros viajes de fin de semana por nuestros coches, como GPS o smartphones. La propia naturaleza de Internet of Things, permite que cualquier dispositivo heterogéneo, sea cual sea su función, sea capaz de recabar información sobre nuestra vida cotidiana con el fin de actuar y mejorar nuestra calidad de vida por medio de la automatización o semi-automatización de tareas. No obstante, eso tiene el problema de que gran parte de nuestra información, tan sencilla como la cantidad de comida que tenemos en la nevera o más sensible como nuestra información bancaria, pueda estar expuesta. Los objetos cotidianos se convierten en un riesgo para la seguridad de la información, ya que la IoT podría distribuir los riesgos mucho más ampliamente de lo que lo ha hecho Internet hasta la fecha [6,17].

Por ello, se hace necesario que la comunicación sea segura, independientemente del entorno en el que se desarrolle, seguro o no, y pueda garantizar que la información transmitida sea completamente privada y no pueda ser leída por usuarios no autorizados en el caso de ser sustraída, eliminado los ataques o minimizando su impacto en el caso de que se produzcan.

Internet of Things

Cada día más objetos están incluyendo sensores y la capacidad de comunicarse. Las redes que se adapten y soporten esta comunicación crearán nuevos modelos de negocio [4] y un entorno futuro en el que la información se encuentre a nuestro alcance y siempre disponible, con dispositivos ubicuos e interconectados entre sí ofreciendo nuevas posibilidades de comunicación [33] y avanzando hasta un punto en el que habrá más objetos conectados que personas [32]. Dicho futuro, realmente se construirá a medida que evolucione la tecnología y crezca la demanda popular [6]. Esto se denomina Internet of Things, y es la interconexión de objetos heterogéneos y ubicuos a través de Internet [6,7,36]. Internet of Things, que surgió por la necesidad de identificar cosas mediante etiquetas RFID [6,36] para localizar objetos y realizar determinadas tareas, enfoca su punto central en el uso de sensores para la recogida de información [32].

Además se prevé que tenga un alto impacto sobre varios aspectos de la vida cotidiana [6] y en el comportamiento de los usuarios potenciales, como la domótica, la vida asistida, e-health, la mejora del aprendizaje, la automatización de negocios y logística, entre otros. La integración de sensores en los productos alimentará este cambio en la vida cotidiana, varios ejemplos de estos son:

- Las compañías de seguros podrán utilizar los sensores del automóvil, o instalar nuevos, para saber cómo se conduce ese vehículo, por dónde y los riesgos que entraña [4].
- Prevención de accidentes mediante la recolección de datos de conducción automovilísticos emitidos por los sensores de otros vehículos.
- La automatización de ámbitos en las actividades cotidianas, por ejemplo en e-health, dónde podría informar con antelación de posibles hospitalizaciones no planificadas gracias a la monitorización de aspectos cotidianos de los usuarios, ya que se prevé que en 2025 haya nodos de IoT en todo tipo de actividades diarias [17], lo que podría reducir los costes de hospitalización y tratamiento en mil millones de dólares anualmente en los Estados Unidos [4].
- Mejorar la gestión de productos en todos los procesos de su ciclo de vida [37].
- Otros ejemplos serían sensores en lugares inaccesibles para el ser humano o en tanques y aviones militares en un campo de batalla [29].

Toda esta innovación se hará efectiva por la incorporación de una mejora electrónica en los objetos físicos cotidianos, convirtiéndolos en ''inteligentes'' y dejándolos integrarse dentro de la infraestructura global [4,33,36]. Por ello, es relevante garantizar la seguridad en estas comunicaciones y asegurar la identidad de dichos usuarios [38].

Smart Objects

Los Smart Objects son objetos con la capacidad para interactuar entre sí mismos y con el medio ambiente [37], mediante sensores y demás elementos colocados con este fin, y con inteligencia propia para tomar decisiones en función a los datos que recojan de estas comunicaciones. Ejemplos de Smart Objects ampliamente instaurados hoy en día en nuestra vida cotidiana son los smartphones, las tables y las Smart TVs, entre otros.

Según [37] los Smart Objects se pueden clasificar según su inteligencia y tipo en función de tres dimensiones que son las siguientes:

- **Nivel de inteligencia**: Se centra en la inteligencia del objeto y se divide en tres categorías:
  - *Gestión de la información*: es la capacidad para manejar la información recogida.
  - *Notificación del problema*: es la capacidad de notificar a su propietario cuando ocurre un problema o evento.
  - *Toma de decisiones*: es la capacidad de decisiones sin intervención de un control externo.
- **Localización de la inteligencia**: Esta dimensión, a su vez, está dividida en otras dos categorías que son las siguientes:
  - *Inteligencia a través de la red*: la inteligencia del objeto depende de un agente externo al propio objeto, como una red.
  - *Inteligencia en el objeto*: Los objetos realizan el trabajo por sí mismo, toda la inteligencia está en ellos.
- **Agregación del nivel de inteligencia:** Esta dimensión, a su vez, está dividida en otras dos categorías que son las siguientes:
  - *Inteligencia en el elemento*: son aquellos objetos que sólo manejar información, notificaciones y decisiones, como un sensor.

- *Inteligencia del contenedor*: son capaces de trabajar como un objeto a pesar de que se le desensamble alguna parte de él. Como por ejemplo una placa Arduino con sensores conectados, si a esta se le quita un sensor, puede seguir funcionando como contenedor.

Para esta propuesta utilizamos varios Smart Objects, con un nivel de inteligencia, gestión de la información y la notificación del problema y con la ubicación de la inteligencia a través de la red. No utilizamos las tarjetas SmartTags como Radio Frequency IDentification (RFID) o Near Field Communication (NFC) por su escasa capacidad de procesamiento.

Algunos ataques en IoT

En este apartado se resumirán algunos de los ataques que pueden comprometer nuestra seguridad en Internet of Things, el objetivo es tenerlos presentes y que nuestras medidas logren evitarlo o al menos que reduzcan la posibilidad de su aparición o mitiguen el daño que podrían producirnos en caso de darse.

Algunos de estos ataques de seguridad potencialmente peligrosos en IoT, debido a su funcionamiento a través de Internet, son los siguientes.

Ataques de replay

Ataques de denegación de servicio o de suplantación de identidad dónde un nodo se interpone en las comunicaciones y captura los mensajes enviados, parando el tráfico, volviendo a introducirlos en la red para producir sobrecargas, o suplantando la identidad de alguno de los nodos y enviandolos en su lugar.

En IoT, como hemos mencionado anteriormente, los recursos de los nodos son limitados, por lo que la capacidad contra un ataque DDOS (Distributed Denial of Service), dónde un recurso deja de estar disponible para sus usuarios legítimos, es débil [39]. Según [39] los nodos deben tener un sistema de cierre por control remoto que les ayude a defenderse de los ataques DDOS, pueden limitarlos en gran medida llegando incluso hasta paralizarlos por completo.

Algunos de los métodos de este ataque son:

- Consumo de recursos computacionales.
- Interrupción de componentes físicos de red.
- Obstrucción de medios de comunicación entre usuarios.

Algunas de las soluciones a este tipo de ataques pasan por establecer marcas temporales.

Ataque Man-in-the-middle

El ataque Man-in-the-middle es un ataque en el que se adquiere la capacidad de leer, insertar y modificar a voluntad, los mensajes entre dos partes sin que ninguna de ellas conozca que el enlace entre ellos ha sido violado.

En IoT, ninguna de las soluciones existentes puede ayudar a resolver el problema de ataque Man-in-the-Middle [6]. El ataque se realizaría colocando dos nuevos objetos, uno cercano al emisor y el otro al receptor, e interceptando estos los mensajes y haciéndoselos llegar del mismo modo al nodo destino de cada uno, de esta forma serían una especie de transmisores de los que el nodo final no podría tener constancia. Si el sistema estuviera protegido por un encriptado de clave pública, el interceptor se colocaría en el medio y, durante el intercambio de clases, sustituiría la clave pública de los nodos por la suya propia, para que la comunicación se realizase con él mismo.

El encriptado, aunque impida la lectura del mensaje, no puede evitar que sea interceptado por los nuevos objetos. Algunas de las consecuencias derivadas de este tipo de ataque son:

- Intercepción de la comunicación.
- Ataques a partir de textos cifrados.
- Ataques de sustitución.
- Ataques de denegación de servicio.

Algunas de las soluciones más frecuentes, o que adoptan los métodos criptográficos son:

- Autenticación mutua en escenarios PKI.
- Autenticación mutua más rígida, claves secretas y contraseñas.
- Comprobación de la latencia, si tarda más de lo normal en llegar de un extremo a otro puede que haya un tercero.

Criptography

En este apartado vamos a hablar de la criptografía, que es la medida que hemos decidido emplear para este caso de estudio, y de sus diferentes tipos, en función de los cuales varía el protocolo, el tamaño de clave y el tiempo empleado.

Cada vez habrá más dispositivos conectados a la red en IoT [6,17]. Los sensores repartidos en una ciudad inteligente podrán comunicarse con smartphones, por lo que este canal requiere protocolos de seguridad que conecten todos estos dispositivos a través de Internet [35]. Para realizar dicha comunicación segura en IoT, una solución posible pasa por utilizar la criptografía como medida de seguridad.

Existen varias formas de tomar medidas criptográficas para encriptar un mensaje, que hacen variar su estructura, el tipo y la cantidad de claves, su tamaño, la capacidad de procesamiento y almacenamiento necesarias para llevarlas acabo, etc. Según los conjuntos de funcionalidades que agrupan algunos de los tipos de criptografía existentes son los que siguen a continuación.

Criptografía simétrica

La **Secret Key** o **Symmetric Key Criptography**, es un método de encriptación de datos en el que el emisor y el receptor comparten la misma clave. Ambas partes han de ponerse de acuerdo en la contraseña que van a utilizar, posteriormente el usuario que ejerza de emisor deberá cifrar el mensaje con ella, el receptor simplemente tendrá que utilizar la misma clave para desencriptarlo.

Ventajas

- El punto fuerte de este tipo de encriptado es la **velocidad**, por lo que se hacen más relevantes a la hora de cifrar grandes cantidades de información.

Inconvenientes

- El principal problema es la **distribución de claves**, pues ambos deben conocerla. Si se envía por un canal no seguro y la clave es interceptada cualquier atacante podría desencriptar el mensaje y acceder al contenido.
- Otro problema vendría dado por el **almacenamiento de claves** en una red grande, ya que se necesitaría una clave por cada par de nodos que estuviesen en contacto.
- También tiene un problema de **integridad** en el intercambio de las claves, ya que de ser interceptada, un usuario malintencionado podría modificar el mensaje.
- Además, este método no garantiza el **no repudio ni la autenticación**, ya que no puedes saber quien fue el emisor original del mensaje y por lo tanto no sabes si fue un usuario registrado en la aplicación y puede ser seguro.

Ejemplo

En este apartado se muestra una ilustración para ejemplificar como funciona este tipo de criptografía.

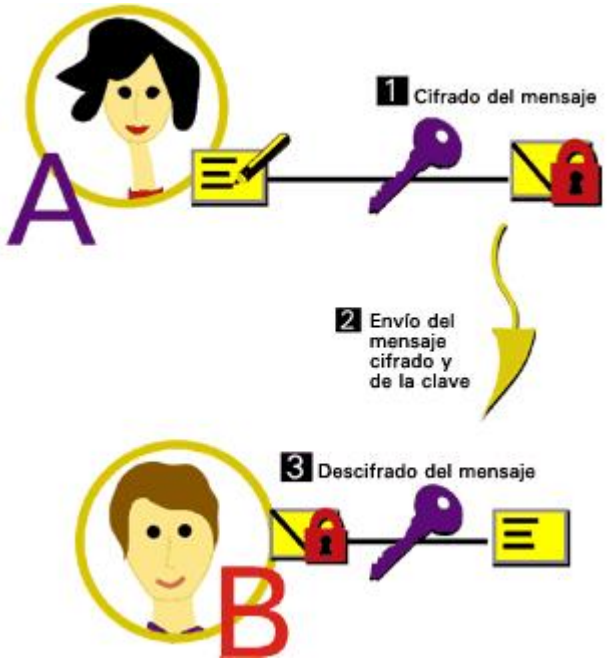


Ilustración 1. Criptografía simétrica

En la Ilustración 1 se muestra un ejemplo de criptografía simétrica que funciona del siguiente modo:

- Ambos usuarios tienen la misma clave.

- El usuario emisor, en este caso A, deberá encriptar el mensaje con dicha clave y enviarlo al usuario B.
- El usuario B se encargará de utilizar la misma clave para desencriptarlo y acceder al contenido.

Criptografía asimétrica

La **Public Key** o **Asymmetric Key Criptography**, es un algoritmo de encriptación de datos que utiliza un par de claves para el envío de mensajes: tanto emisor como receptor tienen su propia clave pública y privada, como se puede comprobar en la Ilustración 2.

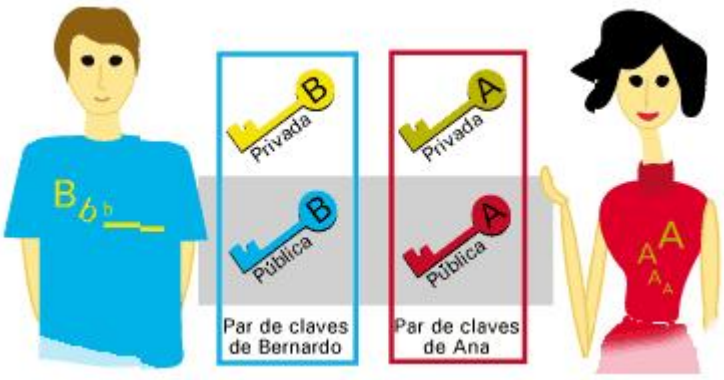


Ilustración 2. Claves criptografía asimétrica

La clave pública se puede entregar a todos los nodos con los que pretenda establecer conexión, mientras la privada deberá ser guardara y tener acceso a ella sólo el propietario. Cuando se quiera enviar un mensaje a un nodo, este ha de encriptarlo con la clave pública de ese nodo. El nodo lo desencriptará con su clave privada. Esto hace que el nodo sea el único que pueda desencriptar dicho mensaje. Si por el contrario se hace al revés, firmando con la clave privada, cualquiera podría desencriptar el mensaje porque todos los usuarios podrían tener la clave pública del emisor. De esta manera se crea la **firma digital**, ya que sólo el emisor tiene su clave privada y por lo tanto es el único capaz de encriptar los datos con ella, permitiendo a los demás verificar que fue él quien lo firmó.

Ventajas

- El punto fuerte de esta metodología, en comparación con la criptografía simétrica, es la **distribución de claves**, ya que se suprime la necesidad de intercambiar la clave redundando en una mayor seguridad.
- Otra ventaja es la **confidencialidad**, ya que al contrario que anteriormente nos podemos asegurar que sólo el destinatario del mensaje va a leerlo, ya que podrá ser cifrado con su clave pública, por lo tanto para descifrarlo es necesaria la clave privada del usuario destino, que sólo él posee.

Inconvenientes

- En contraprestación a las ventajas obtenidas respecto a la criptografía simétrica, **pierde la velocidad** que la caracterizaba, volviéndose más lento.
- Siguen presentes los **problemas de integridad, autenticación y rechazo o repudio**.

Ejemplo

En este apartado se muestra una ilustración para ejemplificar como funciona este tipo de criptografía.

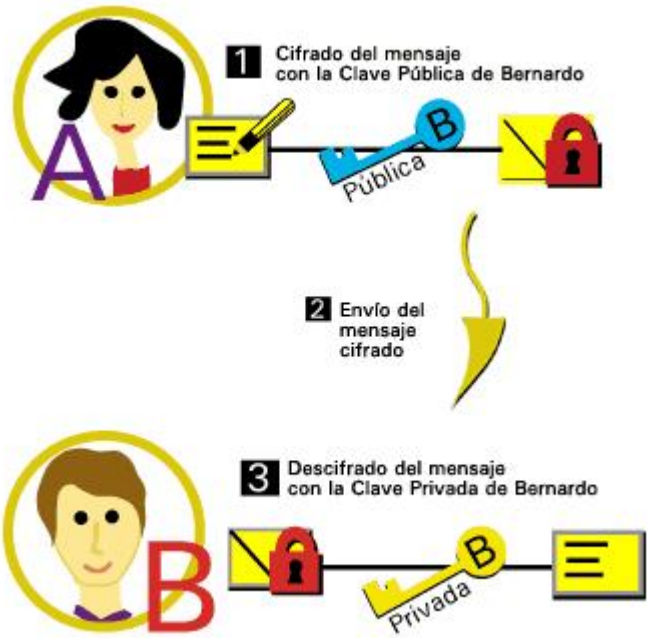


Ilustración 3. Criptografía asimétrica

En el ejemplo de la Ilustración 3, se observa un ejemplo de criptografía asimétrica, del siguiente modo:

- El usuario A encripta el mensaje con la clave pública del usuario B al que va dirigido, garantizando así que sólo el usuario B, que posee su clave privada, será capaz de desencriptar dicho mensaje, por lo que si éste fuese interceptado durante el envío no podría ser leído.
- Finalmente el usuario B recibe el mensaje y lo desencripta con su clave privada, accediendo al contenido.

Criptografía híbrida

La **Hybrid Key Criptography** es un método criptográfico que utiliza la clave simétrica y la asimétrica. Necesita una clave secreta, al igual que ocurre en la criptografía simétrica, y un par de claves, al igual que en la criptografía asimétrica. Así cuando se desea enviar un mensaje a otro usuario, se encripta el mensaje con la clave secreta y se encripta la clave secreta con la

pública del nodo al que se ha de enviar. El nodo que lo reciba podrá desencriptar la clave secreta con su clave privada, y con la secreta que obtuvo podrá desencriptar el mensaje original.

Ventajas

- Este método ofrece una **mayor velocidad** que el cifrado asimétrico pero es más lento que el cifrado simétrico. No obstante, soluciona el problema de envío de claves del cifrado simétrico.
- Al igual que en la criptografía de clave pública, mediante la criptografía híbrida logramos confidencialidad, siendo el receptor el único capaz de leer le mensaje, pero en este caso también se alcanza la **integridad**, ya que un usuario malintencionado no podría modificarlo.

Inconvenientes

- Siguen presentes como inconvenientes la **autenticación y el no repudio**, ya que debido a las medidas utilizadas en este caso aún no poseemos ningún mecanismo para garantizar la autoría.

Ejemplo

En este apartado se muestra una ilustración para ejemplificar como funciona este tipo de criptografía.

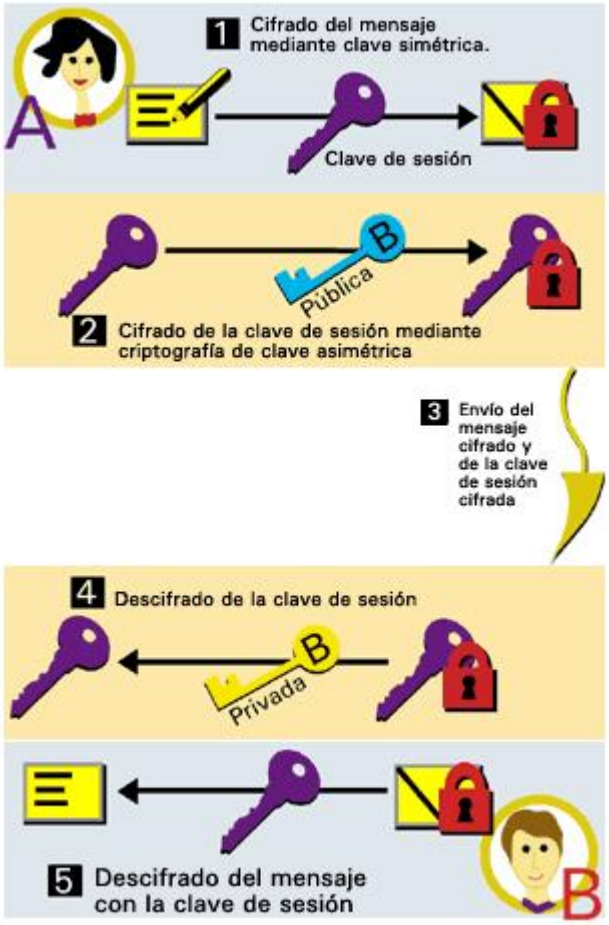


Ilustración 4. Criptografía híbrida

En la Ilustración 4 se puede ver un ejemplo de criptografía híbrida, funciona del siguiente modo:

- El usuario A encripta el mensaje original con una clave simétrica, cuyo tiempo es inferior al de cifrar con una clave asimétrica.
- Posteriormente, el usuario A encripta la clave simétrica con la que encriptó el mensaje con la clave privada del usuario B, al que va dirigido.
- Envía ambas cosas.
- El usuario B desencripta con su clave privada la clave simétrica que le ha enviado el usuario A, siendo el único capaz de hacerlo.
- Con la clave simétrica, descifra el mensaje enviado y accede al contenido original enviado por el nodo.

Firma digital

La **Digital Signature** se basa en la autenticación del emisor, como entidad emisora del mensaje, para que el receptor pueda identificar a la entidad que está realizando una transacción electrónica y que el mensaje no ha sido modificado. El funcionamiento es, en parte, igual al de criptografía asimétrica e híbrida. Emisor y receptor deben tener su par de claves pública y privada. El emisor emitirá su clave pública, para que el receptor la conozca y pueda utilizarla. El

emisor encripta con su clave privada, siendo el único que la posee y garantizando así su autoría, el mensaje y posteriormente le aplica una función hash. Envía al receptor tanto el mensaje original cifrado como la función hash, y este se encarga de desencriptar el mensaje original, con la clave pública del emisor, y aplicarle una función hash para comprobar que el resultado sea el mismo que en el hash recibido y garantizar así que no ha sido modificado durante el envío.

Ventajas

- Con este método ganamos seguridad en la **autenticación**, ya que identifica al emisor del mensaje, pues sólo el nodo poseedor de su clave privada única puede firmar con ella.
- En consecuencia ganamos el **no repudio**, ya que no podría negar haber enviado ese mensaje.
- También ganamos **integridad**, ya que el resumen del mensaje firmado podría compararse con el mensaje original y deducir así si ha sido o no modificado durante el envío.

Inconvenientes

- Por el contrario se presenta un problema similar al de la criptografía simétrica: se pierde la privacidad y la confidencialidad.

Ejemplo

En este apartado se muestra una ilustración para ejemplificar como funciona este tipo de criptografía.

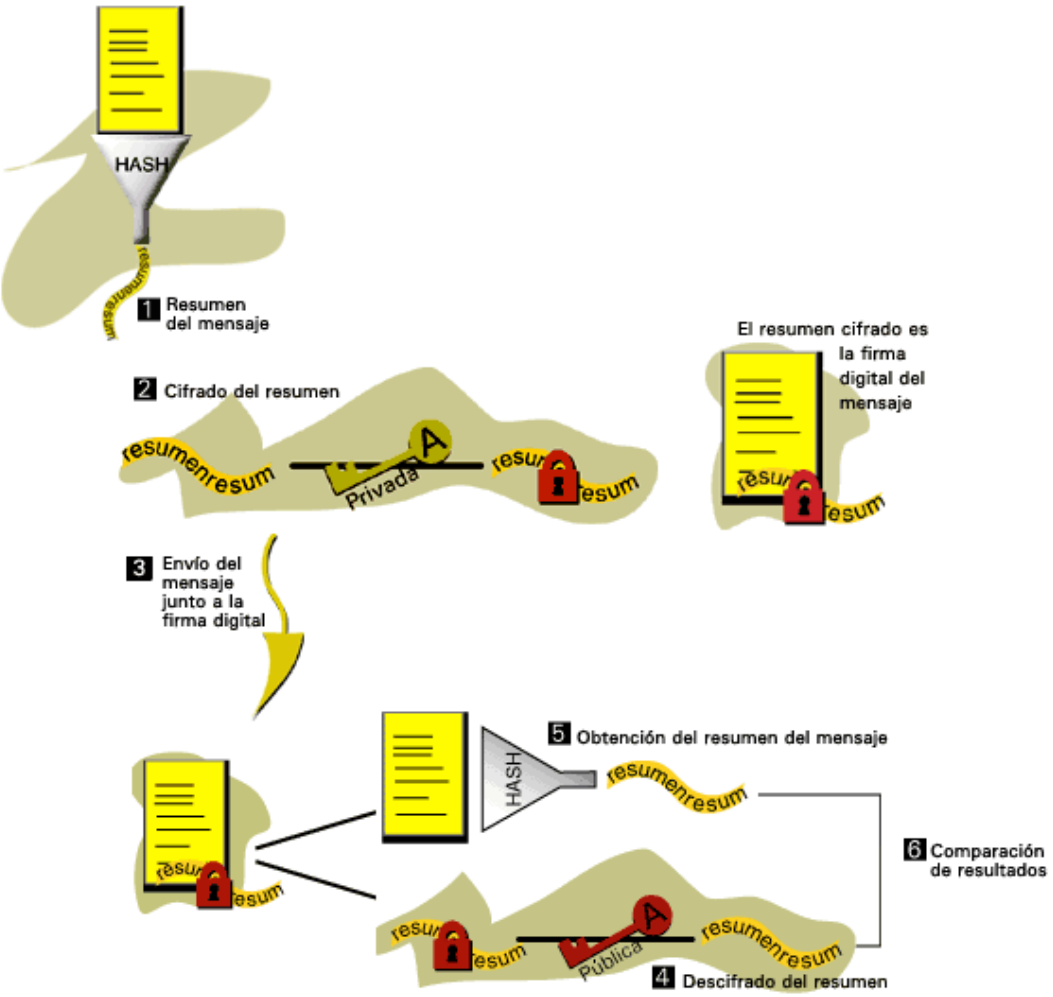


Ilustración 5. Firma digital

En la Ilustración 5 se puede observar un ejemplo de firma digital, que funciona del siguiente modo:

- Se crea un resumen del mensaje original.
- Se encripta el resumen con la clave privada del usuario, creando así una firma al autenticarse como usuario emisor, siendo el único que posee su clave privada y que habría podido firmarlo.
- Se envía al usuario destino, que se encarga de desencriptarla con la clave pública del usuario que lo ha encriptado, garantizando así su autoridad.
- El usuario destino resume el mensaje original y comprueba que coincide con el resumen que había recibido, garantizando así que no ha sido modificado durante el envío y, por lo tanto, la integridad del mismo.

Funciones hash

Una **funcione hash** es un algoritmo que se aplica sobre una cadena de texto para producir un resumen de longitud fija y que impide obtener el mensaje original a partir de este resumen. Una función Hash segura cumple las siguientes funciones: es imposible volver al mensaje original (unidireccional) desde su resumen, mensajes de longitud fija e inferior al original (compresión), debe ser fácil de calcular (facilidad) y debe ser difícil encontrar dos mensajes de entrada que produzcan la misma salida (colisión fuerte).

Así, aplicando una función Hash al mensaje original permite saber al destinatario si se produjeron modificaciones sobre el mensaje original, por lo que garantiza su integridad.

Comparativa

En la Tabla 1 mostramos los diferentes tipos de medidas criptografías que se pueden emplear y determinamos si cumplen o no las medidas de seguridad mencionadas y que consideramos necesarias, tratando así de encontrar la opción que nos proporcione mayor seguridad con menos inconvenientes.

Tipo criptografía	Distribución de la clave	Confidencialidad y privacidad	Autenticación	No repudio	Integridad
Simétrica	No	No	No	No	No
Asimétrica	Si	Si	No	No	No
Híbrida	Si	Si	No	No	Si
Intercambio de clave	Si	Si	No	No	Si
Firma digital	Si	No	Si	Si	Si

Tabla 1. Comparación de los tipos de criptografía

Como se puede observar, ninguna cumple todas las medidas de seguridad necesarias, obligando así, a realizar una combinación entre varias de ellas si se desea ofrecer las cinco medidas de seguridad. En esta comparativa quedan excluidos temas de importante relevancia, debido a la plataforma de IoT y al entorno final de aplicación del sistema, como la velocidad de la operación en función del tamaño de claves, tipo de criptografía empleada, algoritmos, etc. Serán evaluados en la sección 4.

Propuestas para la seguridad de Internet of Things en la investigación

Actualmente, la IoT se está desarrollando en su fase primaria, y la búsqueda de mecanismo de seguridad es un espacio en blanco en la práctica, por lo que tenemos un largo camino para la investigación de este dominio [19]. Cada vez habrá más nodos conectados a la red, con características particulares y diferentes y esto da como consecuencia el problema tener capacidades de computación limitadas y la heterogeneidad de los objetos. Debido al

crecimiento de la IoT y de los Smart Objects cada vez se prestará más atención en la investigación a estas dificultades en la seguridad [19].

El primero de estos problemas vienen dado por el tamaño de la contraseña, desde el punto de vista que cuanto más grande sea esta mayor seguridad ofrece, pero también mayor espacio ocupa [6] y mayores serán las necesidades de computación que necesite. El tamaño de la contraseña viene condicionado por el tipo de criptografía que se utilice. Algoritmos simétricos como Advanced Encryption Standard (AES) están limitados a contraseñas de 128-256 bits, mientras que en algoritmos asimétricos como Rivest, Shamir y Adleman (RSA) es recomendable utilizar contraseñas de mayor tamaño, por ejemplo 1024 bits para ofrecer mayor seguridad.

Según [6] los algoritmos criptográficos típicos gastan gran cantidad de recursos en términos de energía y ancho de banda, tanto en el origen y el destino por lo que estas soluciones no pueden ser aplicadas en la IoT, dado que van a incluir elementos (como las etiquetas RFID y los nodos de sensores) que están seriamente limitados en términos de capacidad de energía, comunicaciones y computación.

El intercambio de claves tradicional y los protocolos de distribución de claves basado en infraestructuras no son prácticos para una red a gran escala debido a la topología de la red desconocida antes del despliegue, las limitaciones de comunicación, funcionamiento intermitente del sensor y la dinámica de la propia red [40]. Todo ello aplicado sobre Smart Objects que cuentan, como ya hemos mencionado, con capacidades de computación y almacenamiento limitadas [41].

Debido a estos problemas deberemos buscar, entre todas las opciones posibles que hemos planteado anteriormente, la que mejor se adopte a las medidas de seguridad requeridas para la plataforma y con menor consumo y necesidades de computación debido a los objetos finales de su uso.

En [18] ven la criptografía como un mecanismo útil para utilizar sobre los sensores, ya que los consideran una parte difícil de asegurar debido a que habría numerosos nodos, y estos tendrían una capacidad de cómputo y almacenamiento limitados. Estos autores han creado un protocolo de autenticación de clave variable segura, utilizando una sola vez un método de cifrado. En este enfoque se crea una matriz de claves entre el remitente y el receptor, y se envían coordenadas de la clave en lugar de la propia clave. Los autores utilizan el ejemplo de un mando a distancia de coche en el que, si un atacante monitoriza una señal y la graba, también sería capaz de repetir la señal y abrir la puerta. Uno de los retos señalados es que la matriz de claves tendría que ser reemplazada con frecuencia para garantizar no ser descubierta. Se observa que RFID y Near Field Communication (NFC), unas de las principales tecnologías dentro de la IoT, no fueron diseñadas para llevar a cabo comunicaciones seguras.

En [38,41] buscan una autenticación mutua simple y eficiente y un establecimiento de clave segura basado en ECC. La ventaja del método propuesto es el establecimiento de clave de sesión basado en Criptografía de Curva Elíptica (ECC), con gastos de almacenamiento más pequeños. Buscan mecanismos de control y acceso, en detrimento de mecanismos criptográficos por su consumo, una vez autenticado usarán el control de acceso para poder acceder a los mensajes. Asegurando que, por gasto de consumo, el cifrado simétrico es más

adecuado para IoT. Sin embargo, esta solución no puede asegurar que un tercero que intercepte el mensaje modifique el mismo.

Al contrario que estas medidas, basadas en el cálculo de la clave para el posterior intercambio y encriptado, en [40] aseguran que la forma de distribución de claves en una red desconocida de sensores conectados pasa por la predistribución de claves. Las claves tendrían que estar guardadas en los nodos sensores para acomodar una conectividad segura entre ellos, dicho esquema se basa en la probabilidad del intercambio de claves entre los nodos de un grafo aleatorio y utiliza un protocolo de descubrimiento compartido de clave simple para la distribución de claves, revocación y nodo de volver a escribir. Esto entraña problemas de seguridad debido al intercambio de las claves, y no se garantiza la integridad del mensaje enviado.

Por el contrario, en [20] presentan un sistema criptográfico simétrico basado en el algoritmo AES para encriptar mensajes e identificar tarjetas RFID, debido a las capacidades limitadas de este tipo de tarjetas. Para buscan modificar el protocolo y proponer un sistema de challenge-response que proporcione un mecanismo criptográfico más rígido. Del mismo modo en [26], dónde defienden que las soluciones de alta seguridad se pueden usar en el entorno de RFID, sin afectar sustancialmente a los protocolos de comunicación actuales, eligiendo adecuadamente y combinándolo con algoritmos criptográficos de bajo coste. Emplean una función básica de cifrado simétrico o asimétrico, por ejemplo, AES. Ambas propuestas [20,21] buscan solución al problema de autenticación que poseen las tarjetas RFID debido a su baja capacidad de procesamiento y debido a la necesidad de autenticación porque las etiquetas son generalmente pasivas y responden a las preguntas de los lectores, independientemente de la voluntad de sus poseedores, un atacante puede interceptar la respuesta de una etiqueta a un lector autorizado [6]. En la literatura actual podemos ver otras posibles soluciones para la seguridad en la comunicación entre el lector y las tarjetas RFID, como por ejemplo [22], dónde la señal transmitida por el lector tiene la forma de un pseudo-ruido, dicha señal es modulada por las etiquetas RFID y por lo tanto, su transmisión no puede ser detectada por los lectores maliciosos. Aunque todavía se encuentran en una etapa temprana [6]. Se trata de una solución interesante para garantizar el envío seguro entre el lector y la tarjeta, pero nosotros nos centramos en la comunicación entre los lectores dentro de una red IoT.

En [42] emplean cifrado por salto, en el que cada nodo puede recibir el mensaje en texto plano, por lo que necesitan una alta credibilidad en los nodos de transmisión. Este método podría ser empleado en negocios cuyos requisitos de seguridad no sean muy altos, al contrario, para garantizar la seguridad debería emplearse cifrado de extremo a extremo [19,33].

Soluciones a más alto nivel, como en [43], requieren de un proxy que actúe por un lado con el usuario y con los servicios por el otro, con el fin de garantizar que los datos personales recogidos son utilizados únicamente para apoyar los servicios autorizados por los proveedores autorizados.

Soluciones basadas en la criptografía pública, haciendo eco de su mayor consumo, las encontramos en [44] dónde revisan su uso para ciertos tipos de sensores de capacidades limitadas dentro de una red inalámbrica. El problema es que al adaptar el sistema para las

capacidades de los sensores, el envío encriptado no puede garantizar la integridad del mensaje, la autenticación ni su resistencia a ciertos ataques.

En [45], conscientes de las amenazas en la IoT, provenientes tanto de la red inalámbrica como de la situación física, proponen una implementación del protocolo Constrained Application Protocol (COAP), sobre Datagram Transport Layer Security (DTLS). Su objetivo es utilizar una pequeña implementación de COAP, para un mayor ahorro energético, y adaptar DTLS, pensado para dispositivos de mayor capacidad, a las limitaciones de los dispositivos inteligentes. Utilizan el algoritmo de criptografía simétrica AES en las comunicaciones, debido al mayor consumo de la criptografía asimétrica. La comunicación entre los nuevos nodos a registrar, y el nodo central, requieren de un tercero que garantice la autenticidad de ambos. Esto, sumado al intercambio de claves y las claves precargadas, lo hace vulnerable a un ataque Man-in-the-Middle, que dicen no se puede proteger mediante sistemas criptográficos, pero si cambiando la limitación física de los nodos y su potencia. En nuestra opinión, el uso de criptografía híbrida y firma digital, no pueden evitar el ataque man-in-the-middle, pero si reducir sus riesgos en las comunicaciones, ya que un mensaje interceptado no podría ser leído, y de ser modificado, el receptor final del mismo lo sabría debido a la firma de su autor y al resumen del mensaje original.

Referencia al artículo	Distribución de la clave segura	Confidencialidad y privacidad	Autenticación	No repudio	Integridad
[18]	**X**				
[38]	**X**	**X**	**X**	**X**	
[41]	**X**	**X**	**X**	**X**	
[20]		**X**			
[21]		**X**			
[42]	**X**	**X**			
[44]	**X**	**X**			
[22]		**X**			**X**
[45]			**X**	**X**	

Tabla 2. Soluciones para IoT en la investigación

En la Tabla 2. Soluciones para IoT en la investigación mostramos un resumen del análisis de algunos de los artículos que ofrecen opciones de seguridad en IoT. Como se pueden comprobar algunas de las soluciones no ofrecen la autenticación, el no repudio y la integridad características de la firma digital, mientras que otros ofrecen soluciones completas que bien no pueden garantizar la integridad del mensaje o la privacidad del mismo. Las mejores soluciones de las estudiadas son la [42] y [45] pero pueden presentar problemas de ataque si el mensaje es interceptado.

Nosotros estudiaremos una solución utilizando los métodos criptográficos de las seguridad tradicional con el fin de lograr todos los objetivos presentados en la tabla.

Descripción del Sistema

La solución que proponemos es una mezcla entre la criptografía híbrida y la firma digital, aunque actualmente la mayoría de los esquemas de autenticación seguros se basan en funciones hash o criptografía de clave pública [38]. El objetivo es lograr los beneficios de ambas sin tener que incluir unos en detrimento de otros. Ya que los beneficios de la criptografía híbrida son: confidencialidad, privacidad e integridad del mensaje; mientras que las desventajas que presenta: la autoría del mensaje y el no repudio del mismo, se verían compensadas ya que son las ventajas propias de la firma digital. Del mismo modo, la firma digital perdía en privacidad, confidencialidad e integridad, que se verán compensadas con la criptografía híbrida. Aunando de la mejor manera posible los beneficios de cada una podríamos garantizar todas las medidas preventivas que indicamos habríamos de lograr para conseguir seguridad en IoT.

Arquitectura propuesta

Para llevar a cabo este prototipo implementamos la siguiente arquitectura, dividida en cuatro partes según el actor que la lleve a cabo y el momento de la comunicación.

Registrar objeto

En esta parte se modifica el funcionamiento de la aplicación original de la plataforma Midgar, en la que se registran los nuevos dispositivos que se conectan en la red IoT.

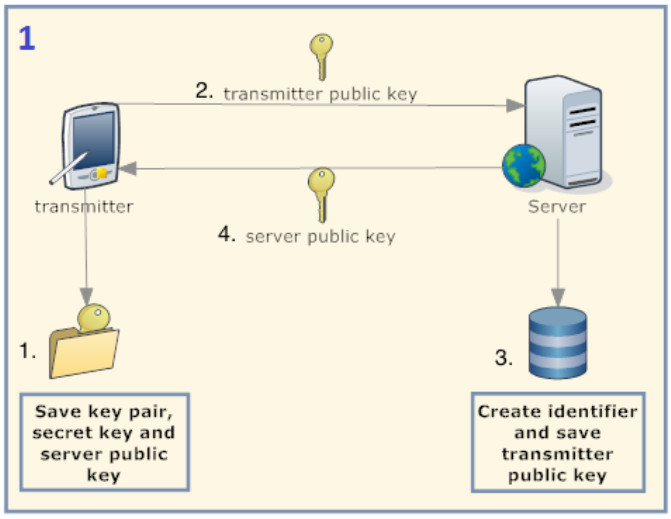


Ilustración 6. Registrar objeto

El objetivo de esta modificación es realizar en este punto, únicamente cuando el dispositivo se registra, la creación de las claves necesarias para realizar una comunicación segura. Para ello se realizarán los siguientes pasos tal y como se enumeran en la Ilustración 6.

1. El dispositivo emisor creará su clave secreta y su par de claves: privada y pública, y las almacenará en su memoria interna.
2. Cuando este dispositivo se registre enviará su clave pública al servidor.
3. El servidor almacenará la clave pública del usuario, junto con el identificador del dispositivo.
4. En respuesta, el servidor enviará al dispositivo su clave pública para mantener las futuras comunicaciones entre ambos.

Enviar mensaje al servidor

El envío del mensaje al servidor conforma el escenario principal de la comunicación entre un par de objetos, cuyo objetivo es que el objeto emisor envíe su información al servidor y este al objeto receptor, para ello se realizarán los siguientes pasos, ilustrados en la Ilustración 7.

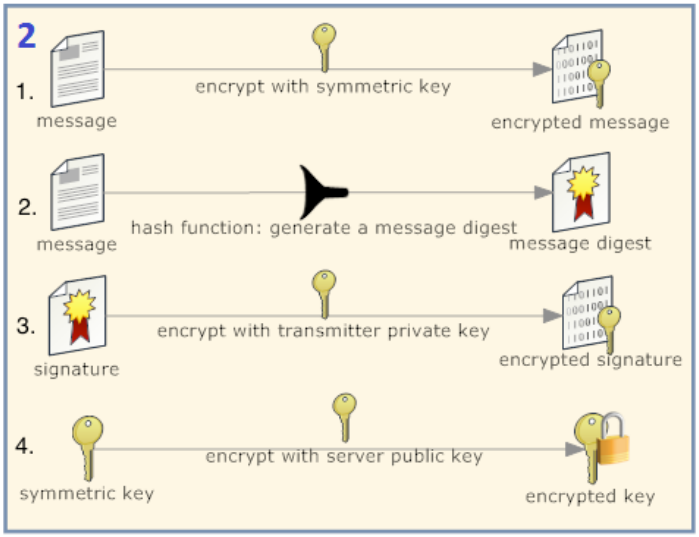


Ilustración 7. Enviar mensaje al servidor

1. El emisor encripta el mensaje a enviar con su clave simétrica, siendo esta la única con la que podrá ser desencriptado.
2. Crea un resumen del mensaje mediante una función hash, para tener una constancia resumida y con menos coste de envío, del mensaje original y poder garantizar así la integridad del mismo.
3. El emisor firma el resumen del mensaje con su clave privada, pudiendo ser desencriptado con su clave pública y garantizando así su autoría del mensaje, ya que es el único que posee su clave privada y por tanto el único que ha podido utilizarla para firmarlo.
4. Finalmente, el emisor encripta la clave simétrica con la clave pública del servidor, otorgándole la capacidad de ser el único capaz de desencriptarla y por lo tanto acceder al contenido del mensaje.

Enviar mensaje al receptor

En este escenario es el servidor el primer receptor del mensaje original, enviado por el emisor, y es el encargado de enviarlo a su receptor final. Los pasos a seguir en este escenario se muestran en la Ilustración 8.

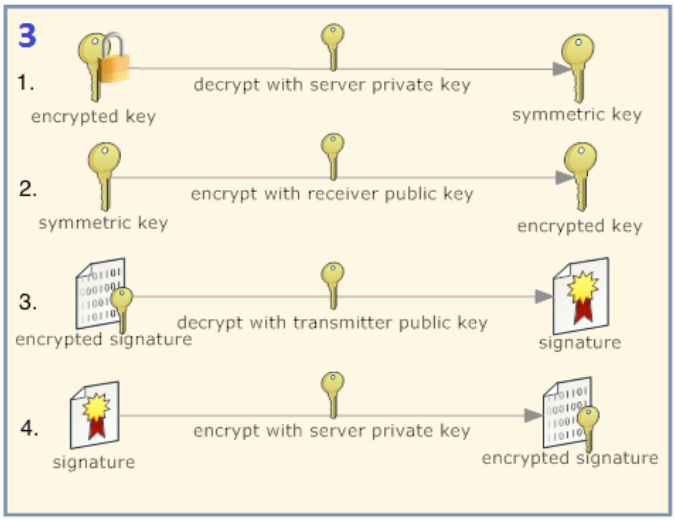


Ilustración 8: Envía mensaje al receptor

1. Desencripta la clave simétrica con su clave privada, llave para poder acceder al contenido del mensaje.
2. Encripta la clave simétrica con la clave pública del receptor final del mensaje, para que sólo este pueda desencriptarla.
3. Desencripta el resumen del mensaje, cifrado con la clave privada del emisor, y garantiza así la autoría del mensaje que acaba de recibir. En caso contrario, lo rechaza.
4. Encripta el resumen del mensaje con su clave privada, garantizando así, al futuro receptor final del mismo, que ha desencriptado el mensaje, de su autor original, y que garantiza así su autoría firmándolo el mismo.

Recibir y leer el mensaje

Este escenario se encarga de la recepción final del mensaje, lo que finaliza con el envío y se realizará completamente en el receptor. Su desarrollo se muestra en la Ilustración 9.

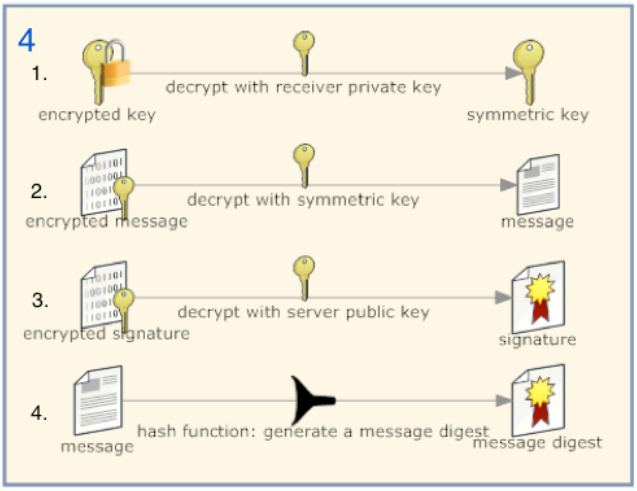


Ilustración 9. Recibe y desencripta el mensaje

1. El receptor desencripta con su clave privada la clave simétrica que el servidor ha cifrado, permitiéndole poder acceder al contenido del mensaje original.
2. Desencripta el mensaje original con dicha clave simétrica.

3. Desencripta el resumen del mensaje con la clave pública del servidor, con lo que garantiza que ha sido el servidor el que le ha enviado el mensaje, por lo tanto garantiza al autoría.
4. Crea un resumen del mensaje original y lo compara con el resumen recibido para ver si es igual y saber si el mensaje no ha sufrido ninguna modificación durante el envío, por lo tanto garantiza la integridad.

Implementación

Para la implementación de esta arquitectura se tuvieron en cuenta o se realizaron los siguientes puntos.

Proyecto toma de tiempos

En primer lugar se realizó un proyecto en Android para evaluar los distintos tipos de criptografía seleccionados en una implementación Java, en función de una serie de algoritmos a los que daba implementación y se medían los tiempos que empleaba en tres puntos principales:

- **Creación de las claves**: Generación de una clave aleatoria que luego podrá ser usada en la comunicación para encriptar/desencriptar el mensaje.
- **Encriptar**: Encriptado del mensaje con la clave generada.
- **Desencriptar**: Desencriptado del mensaje con la clave generada.

Todos estos tiempos fueron evaluados posteriormente para seleccionar la mejor solución, y se discutirán más ampliamente en el 0.

Proyecto conexión Midgar

Una vez seleccionada la mejor opción, se modificó la aplicación móvil para dispositivos Android Midgar, encargada de realizar el registro de los dispositivos y la comunicación para incluir las medidas de seguridad tomadas y proporcionar un funcionamiento seguro. Las modificaciones realizadas fueron las siguientes:

- **Creación del objeto**: A la hora de crear un objeto y asociarlo a un nombre y una descripción, se generan automáticamente su par de claves y su clave simétrica.
- **Registro**: En el registro el dispositivo envía, además de su identificador, descripción y sensores, su clave pública generada. Cuando el envío se realiza correctamente, el servidor añade a su respuesta su propia clave pública para posteriores conexiones con el dispositivo.
- **Envío de información:** Cuando comienzan las conexiones el dispositivo comienza a enviar su información cada cierto tiempo, dado por el intervalo especificado, y antes de este envío, se realiza la encriptación del mensaje, de la clave y la firma digital. En el envío, en lugar de enviar una cosa sola, envía mediante REST los tres valores.

- **Recepción de información:** A la hora de recibir información del servidor, extrae del mensaje enviado los tres valores, y los desencripta para obtener el mensaje original y actuar en consecuencia.

Proyecto Servidor Midgar

Se modificó el servidor de la aplicación, implementando en Ruby, para dar soporte a las nuevas medidas de seguridad incluidas. Las modificaciones fueron las siguientes:

- **Modificación de la base de datos**: Para almacenar las claves públicas de los dispositivos conectados.
- **Fichero de texto**: Para almacenar el par de claves del servidor.
- **Respuesta al registro**: Dónde ahora envía, además la correcta transacción del registro, su clave pública para posteriores comunicaciones.
- **Recepción de un mensaje**: Desencripta el mensaje obtenido, firmado por el dispositivo emisor, y crea un resumen del mensaje para comprobar que no ha sido modificado durante el envío.
- **Envío del mensaje**: Una vez comprobada la autoría y la integridad del mensaje recibido lo firma para enviar a su destinatario final y garantizar así su origen.

Metodología de Trabajo

Selección de los algoritmos

Antes de comenzar la implementación, se llevó a cabo una búsqueda de los algoritmos más usados en cada tipo de criptografía, tanto por rendimiento, tamaño de clave, uso y popularidad. Se seleccionó un grupo de ellos que serían implementados para cada selección.

Criptografía simétrica

Los algoritmos seleccionados, utilizados en Secret Key e Hybrid Criptography, fueron los siguientes:

- **Data Encryption Standard (DES)** [46] es un algoritmo de cifrado, es decir escogido como un estándar en 1976, y de amplio uso. Fue un algoritmo dudoso en sus inicios, con un tamaño de clave corto y continuas sospechas sobre su seguridad. DES motivó el concepto moderno del cifrado por bloques y criptoanálisis. El tamaño máximo de claves es de 64 bits, ofreciendo una seguridad que se cree endeble en ataques teóricos y que instaron la creación de su sucesor el algoritmo Triple DES.
- **Triple DES (3DES)** [47] surgió debido a las dudas sobre el tamaño de clave insuficiente del algoritmo DES, y por ello emplea el doble del tamaño de clave. Es mucho más resistente a ataques que el algoritmo DES debido a su longitud de clave, pero su uso está quedando relegado.
- **Advanced Encryption Standard (AES)** [48] es un algoritmo de cifrado por bloques adoptado también como estándar de cifrado. Sustituye al algoritmo DES y Triple DES como nuevo estándar de cifrado, tanto para hardware como software, siendo posible implementarse en poco tiempo y requiriendo poca memoria. Su tamaño de clave oscila entre un mínimo de 128 bits y un máximo de 256 bits.
- **International Data Encryption Algorithm (IDEA)** [49] es un algoritmo de cifrado por bloques creado y propuesto como sustituto del algoritmo DES, trabajando con una longitud de clave de 128 bits. Considerado uno de los algoritmos más seguros por su fortaleza ante el ataque por fuerza bruta.
- **Blowfish** [50] es un algoritmo de cifrado por bloques simétricos, utiliza bloques de 64 bits y claves que van desde los 32 bits a los 448 bits. Blowfish es un algoritmo que surgió para reemplazar al algoritmo DES, con sus problemas de seguridad.

Criptografía asimétrica

En lo relativo a la criptografía asimétrica, las elecciones buscaban un manejo de claves superior, velocidad y velocidad. Las selecciones, utilizadas en Public Key Criptography, Hybrid Criptography y Digital Signature, fueron las siguientes.

- **Rivest, Shamir, Adleman (RSA)** [51] es el algoritmo de criptografía asimétrica más ampliamente utilizado. El algoritmo RSA trabaja con longitudes de clave que van desde los 1024 bits a los 4096 bits siendo tamaños prácticamente imposibles de resolver por fuerza bruta, los ataques que presenta son propios de la comunicación. Además, permite firmar digitalmente mediante el encriptado con su clave privada.
- **ElGamal** [52] es un algoritmo de criptografía asimétrica, basado en la idea Diffie-Hellman, cuya seguridad se basa en el empleo del logaritmo discreto. Permite utilizar tamaños de claves que van desde los 1024 btis hasta los 4096 bits. Es también, al igual que el algoritmo RSA, válido para firmar digitalmente con su clave privada.

Funciones Hash

Para la función Hash se evaluaron los siguientes algoritmos.

- **MD5** es un algoritmo de reducción criptográfico, empleando para resumir mensajes y diseñado en el MIT, con gran difusión a nivel actual, pese a tener problemas detectados de colisión de hash que pueden comprometer su seguridad [53].
- **SHA-1** es un algoritmo de reducción criptográfica, de amplio uso y con su seguridad en entredicho debido a un conjunto de ataques divulgados sobre una estructura similar a la empleada en SHA-1 y que podría comprometer su seguridad [54].
- **SHA-2** es un algoritmo de reducción criptográfica, sucesor de SHA-1, y que tiene su uso sobre cuatro tamaños de bits, que van desde 224 bits hasta 512 bits. En 2011 un ataque rompió la seguridad de SHA-2 [55].
- **SHA-3** es un algoritmo de reducción criptográfica que surge como alternativa criptográfica frente a SHA-2, sin pretender reemplazarlo, debido a los ataques teóricos respecto a las versiones anteriores. Tiene su uso sobre tamaños que van desde que van desde 224 bits hasta 512 bits.

Toma de tiempos

Se ha desarrollado una aplicación nativa en Android en la que se utilizan implementaciones de los algoritmos mencionados en 0 para cifrar mensajes originales de la plataforma Midgar y claves simétricas generadas. Las acciones principales realizadas en cada algoritmo fueron: generación automática de claves, encriptado y desencriptado.

A la hora de calcular el consumo, y teniendo en cuenta la arquitectura anteriormente presentada, hay que considerar que las claves sólo se calculan en una ocasión, cuando el nodo se registra como un nuevo dispositivo en la plataforma IoT Midgar. Por ello, el consumo de su cálculo sólo sería computado una vez. Por el contrario el encriptado y desencriptado se utilizará durante cada envío de mensaje, por lo que el tiempo de cálculo de claves será menos relevante a la hora de calcular el consumo de un algoritmo.

Para calcular dicho tiempo utilizamos tres dispositivos móviles, con el sistema operativo Android, pero de diferente gama: baja, media y alta. De esta manera, buscamos ver el impacto de computación que puede tener la seguridad en los diferentes tipos de móviles existentes en el mercado, en función a dos aspectos.

Cálculos generales

Calculamos el tiempo empleado en el cálculo de claves, encriptado y desencriptado en los tres dispositivos móviles utilizados. Se hicieron 10.000 iteraciones para cada uno de estos cálculos, para evitar así casos especiales como el recolector de basura, posibles pausas no deseadas o parones del sistema operativo. Después con el resultado obtenido se hizo la media para obtener el tiempo medio por iteración.

Caso de uso Midgar

La solución final y que elegimos como mejor, se implementó en la plataforma IoT Midgar para comprobar su funcionamiento. Para ello, clasificamos los sensores en dos tipos de nodos:

- **Nodo crítico**: Son aquellas acciones que necesitan de varios pasos de mensaje cada poco tiempo debido al uso o posible uso que se les da a estos sensores. Estos son por ejemplo el acelerómetro, sensor de gravedad o el giroscopio. Un ejemplo de su uso es el de la prevención de accidentes, para ello se necesita el uso de la medición del acelerómetro cada pocos segundos como se ve en [56–58]. Para el cálculo de estos nodos, nosotros hemos estimado un intercambio de cuatro mensajes por segundo.
- **Nodo no crítico**: Son aquellas acciones que requieren un mejor paso de mensajes, debido a que el cambio entre una lectura y otra puede resultar mínimo, o a que su impacto no puede resultar tan crítico. Estos son por ejemplo la luz, la temperatura y la presión. Para el cálculo de estos nodos nosotros hemos estimado un intercambio medio de un mensaje por minuto.

Software utilizado

En este apartado se hablará sobre el trabajo realizado y sobre el software y el hardware que se utilizó para llevar a cabo tanto la realización como las pruebas.

Respecto a la evaluación de los algoritmos criptográficos utilizados, y debido a la posibilidad de que los resultados fueran dispares utilizando unas u otras, muestro las librerías utilizadas por si alguien desease repetir dichas pruebas:

- Librerías Security y Crypto de Java.
- Librería Bouncy Castle.

La aplicación de toma de tiempos que hace uso de dichas librerías se realizó en Android, dónde también se modificó la aplicación de uso de Midgar.
Se modificó el servidor creado con el framwork Ruby on Rails para dar cabida a las pruebas realizadas.

Hardware utilizado

Se han utilizado tres dispositivos móviles, cuyas prestaciones van en orden descendente: gama baja, media y alta., en función de su procesador.

- **Dispositivo 1**: HTC Desire 610 con un sistema operativo Android 4.4.2 y un procesador de cuatro núcleos a una velocidad de 1.2 GHz y 1GB de memoria RAM.

- **Dispositivo 2**: LG Optimus L9 con un sistema operativo Android 4.1.2 y un procesador de dos núcleos a una velocidad de y 1GB de memoria RAM.
- **Dispositivo 3**: LG Optimus L7 con un sistema operativo Android 4.0 y un procesador de un núcleo a una velocidad de 1GHz y 512MB de memoria RAM.

Resultados Obtenidos

En esta sección describiremos la metodología empleada para desarrollar esta investigación y las pruebas que se han realizado. Tras ello, mostraremos los resultados y los discutiremos. El principal objetivo de este trabajo es proponer una posible solución para lograr los aspectos anteriormente mencionados: confidencialidad, autenticación, integridad, no repudio y privacidad. De esta manera, intentaremos garantizar seguridad en las comunicaciones en Internet of Things, utilizando como caso de estudio la plataforma Midgar.

La técnica que hemos decidido emplear para conseguir esta seguridad es la criptografía. No obstante, la capacidad limitada de computación de los objetos conectados a la IoT, hace necesario evaluar el coste temporal que emplearía cada tarea en diferentes técnicas y algoritmos criptográficos para seleccionar las más adecuadas. Debido a la estructura seleccionada, debemos analizar algoritmos de criptografía simétrica, asimétrica y funciones hash, para seleccionar la mejor solución posible que cumpla con los objetivos propuestos.

Criptografía Simétrica

En primer lugar se presentan las opciones de criptografía simétrica que hemos seleccionado: DES (64), 3DES(128), AES(128 y 256), Blowfish(128 y 256) e IDEA(128 y 256). No todos los algoritmos gozan de las mismas condiciones iniciales, ya que no todos soportan los mismos tamaños de clave y, en el caso del algoritmo IDEA, no permite generar una clave aleatoria, por lo que sus valores de cálculo de clave no son incluidos. Para realizar los cálculos en la tabla, se emplean los tiempos del cálculo de clave de AES en IDEA.

Para realizar las mediciones se ha encriptado un mensaje original de la plataforma Midgar que podría ser enviado de un objeto al servidor Código fuente 1.

```
<send>
        <data>
                <datum>28</datum>
                <service>18</service>
        </data>
</send>
```

Código fuente 1. Ejemplo de mensaje de datos de Midgar

En la Tabla 3 de la tabla se muestran los resultados obtenidos en los dispositivos empleados.

En ella se muestran los datos para los tres dispositivos y los algoritmos utilizados. Después para cada algoritmo se muestra el tamaño en bits utilizado, los milisegundos que tardo en

crear la clave y en encriptar y desencriptar el texto Código fuente 1 y la suma total de la operación. Al final, se muestran los cálculos acerca de cuánto tiempo se gastaría aplicando ese método de encriptación en un nodo crítico y uno no crítico. Para este cálculo se sumó el tiempo que tardaría encriptar y desencriptar un nodo crítico y uno no crítico, sin incluir el cálculo de la clave ya que su cálculo solamente se realiza durante el registro del dispositivo.

Devices	Algorithms	Size (bits)	Keys (ms)	Encrypt (ms)	Decrypt (ms)	Total (ms)	1 hour communication			
							Critical (ms)	%	Non Critical (ms)	%
Device 1	DES	64	0,0355	0,1575	0,1328	0,3258	4180,32	0,1161	17,42	0,0005
	3DES	128	0,0368	0,2739	0,2649	0,5756	7758,72	0,2155	32,33	0,0009
	AES	128	0,0340	0,1247	0,1362	0,2949	3756,96	0,1044	15,65	0,0004
	AES	256	0,0401	0,1310	0,1536	0,3247	4098,24	0,1138	17,08	0,0005
	Blowfish	128	0,0344	1,1503	1,1607	2,3454	33278,40	0,9244	138,66	0,0039
	Blowfish	256	0,0391	1,1542	1,1435	2,3368	33086,88	0,9191	137,86	0,0038
	IDEA	128	0,0340	0,1941	0,1358	0,3639	4750,56	0,1320	19,79	0,0005
	IDEA	256	0,0401	0,1965	0,1281	0,3647	4674,24	0,1298	19,48	0,0005
Device 2	DES	64	0,0375	0,1945	0,3865	0,6185	8366,40	0,2324	34,86	0,0010
	3DES	128	0,0314	0,3149	0,5028	0,8491	11774,88	0,3271	49,06	0,0014
	AES	128	0,0237	0,1342	0,3494	0,5073	6963,84	0,1934	29,02	0,0008
	AES	256	0,0424	0,1597	0,3809	0,5830	7784,64	0,2162	32,44	0,0009
	Blowfish	128	0,0218	1,2402	1,4471	2,7091	38697,12	1,0749	161,24	0,0045
	Blowfish	256	0,0302	1,2276	1,4410	2,6988	38427,84	1,0674	160,12	0,0044
	IDEA	128	0,0237	0,1617	0,3519	0,5373	7395,84	0,2054	30,82	0,0009
	IDEA	256	0,0424	0,1556	0,3383	0,5363	7112,16	0,1976	29,63	0,0008
Device 3	DES	64	0,0332	0,2469	0,4770	0,7571	10424,16	0,2896	43,43	0,0012
	3DES	128	0,0435	0,3974	0,6522	1,0931	15114,24	0,4198	62,98	0,0017
	AES	128	0,0288	0,1960	0,4777	0,7025	9701,28	0,2695	40,42	0,0011
	AES	256	0,0343	0,2200	0,5063	0,7606	10458,72	0,2905	43,58	0,0012
	Blowfish	128	0,0289	1,7291	2,0003	3,7583	53703,36	1,4918	223,76	0,0062
	Blowfish	256	0,0364	1,7231	2,0301	3,7896	54046,08	1,5013	225,19	0,0063
	IDEA	128	0,0288	0,5054	0,4710	1,0052	14060,16	0,3906	58,58	0,0016
	IDEA	256	0,0343	0,5161	0,4631	1,0135	14100,48	0,3917	58,75	0,0016

Tabla 3. Resultados de los algoritmos simétricos

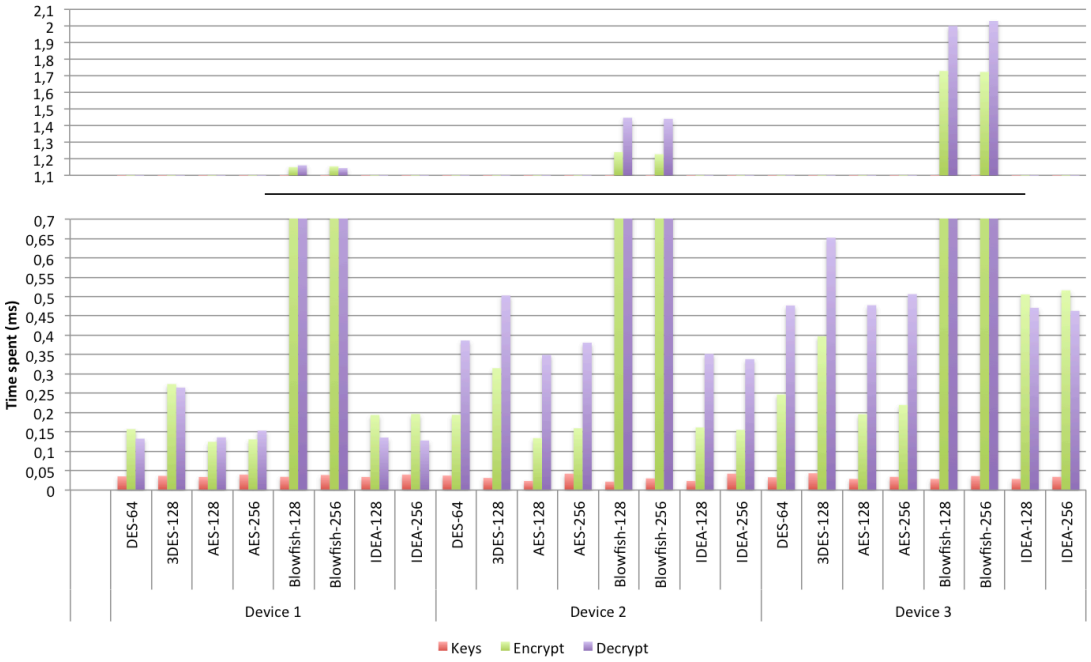


Ilustración 10. Gráfica de resultados de los algoritmos simétricos

Analizando la Tabla 3 y la Ilustración 10, nosotros podemos sugerir las siguientes interpretaciones:

- El cálculo de claves llega a ser prácticamente irrelevante ya que las diferencias son pequeñas y en ninguno de los casos supera los 0,05 ms.
- El algoritmo Blowfish128 bits y Blowfish 256bits, emplea un tiempo de encriptado y desencriptado muy alto, resultando mucho más costoso que el resto de algoritmos evaluados.
- El algoritmo DES, es el más inseguro debido a que utilizar el menor tamaño de clave facilita descifrar las contraseñas por fuerza bruta, al contrario de lo que sucedería con contraseñas mayores y más robustas [31]. Pese a esto, presenta unos tiempos de cálculo superiores frente a otros algoritmos más seguros como el AES 128 bits y AES 256 bits.
- AES 256 bits tarda un poco más que AES 128 bits pero ofrece mayor seguridad al utilizar el doble de bits para la clave.
- 3DES 128 bits tarda el doble que AES 128 bits, permitiendo usar únicamente claves de dicho tamaño.
- IDEA 256 bits tarda un poco más que IDEA 128 bits pero ofrece mucha mayor seguridad al utilizar el doble de bits para la clave.
- IDEA 256 bits requiere un poco más de tiempo que AES 256 bits
- Así, finalmente la decisión estaría entre los algoritmos AES e IDEA, dónde los tiempos de AES son inferiores, sobre todo en encriptado, tanto para una clave de 128bits como de 256bits. Respecto al tiempo que el algoritmo emplea encriptando y desencriptando en una hora de comunicación, el porcentaje de AES es inferior en todos los casos, siendo el que menos tiempo emplea AES 128 bits, seguido de AES 256 bits.
- El tiempo empleado con AES 128 bits emplea, según el cálculo medio entre los tres dispositivos, es de 0,19% para nodos críticos y 0,0008% para nodos no críticos. Por otra parte AES 256 bits emplea 0,21% para nodos críticos y 0,0009% para nodos no críticos.

Al suponernos una mayor seguridad en su versión de 256 bits, con un consumo extra de 0,02% para nodos críticos y 0,0001% para nodos no críticos, seleccionaremos la versión de AES 256 para nuestra solución por ser la más rápida y segura.

Criptografía Asimétrica

En esta sección, evaluaremos el envío encriptado del mensaje mediante el uso de un algoritmo de clave pública. Este nos permite crear un par de claves, para cada nodo, e intercambiar información de forma segura asegurando que únicamente el receptor sea la persona capaz de desencriptar ese mensaje, es decir, su confidencialidad. Los algoritmos seleccionados en este punto fueron RSA y ElGamal.

Para realizar las mediciones, se ha encriptado una contraseña simétrica generada en nuestra aplicación Código fuente 2 y que consta de un tamaño intermedio de 128bytes.

```
W1PnUCA+fVEAOaKaxfu1cQ==
```

Código fuente 2. Contraseña

En la Tabla 4 se muestras los resultados obtenidos para los tres dispositivos y los algoritmos utilizados. Después para cada algoritmo se muestra el tamaño en bits utilizado, los milisegundos que tardo en crear la clave y en encriptar y desencriptar el texto Código fuente 1 y la suma total de la operación. Al final, se muestran los cálculos acerca de cuánto tiempo se gastaría aplicando ese método de encriptación en un nodo crítico y uno no crítico. Para este cálculo se sumo el tiempo que tardaría encriptar y desencriptar un nodo crítico y uno no crítico, sin incluir el cálculo de la clave ya que su cálculo solamente se realiza durante el registro del dispositivo.

Devices	Algorithms	Size (bits)	Keys (ms)	Encrypt (ms)	Decrypt (ms)	Total (ms)	1 hour communication			
							Critical (ms)	%	Non Critical (ms)	%
Device 1	RSA	1024	806,7	0,54	7,58	814,82	116928	3,248	487,2	0,014
	RSA	2048	2671,7	1,41	39,32	2712,43	586512	16,292	2443,8	0,068
	RSA	4096	24460,3	4,49	248,53	24713,32	3643488	101,208	15181,2	0,422
	ElGamal	1024	16,8	31,36	14,47	62,63	659952	18,332	2749,8	0,076
	ElGamal	2048	118,2	229,98	117,15	465,33	4998672	138,852	20827,8	0,579
	ElGamal	4096	880,5	1783,21	892,33	3556,04	38527776	1070,216	160532,4	4,459
Device 2	RSA	1024	869	0,746	8,99	878,736	140198,4	3,894	584,16	0,016
	RSA	2048	2821,7	1,807	40,62	2864,127	610948,8	16,971	2545,62	0,071
	RSA	4096	27357,3	5,46	256,44	27619,2	3771360	104,760	15714	0,437
	ElGamal	1024	17,7	32,54	15,86	66,1	696960	19,360	2904	0,081
	ElGamal	2048	123,2	240,36	120,32	483,88	5193792	144,272	21640,8	0,601
	ElGamal	4096	933,9	1864,92	933,53	3732,35	40297680	1119,380	167907	4,664
	RSA	1024	886,8	1,5	9,898	898,198	164131,2	4,559	683,88	0,019

Device 3	RSA	2048	6501	2,7	50,769	6554,469	769953,6	21,388	3208,14	0,089
	RSA	4096	42143,3	6,02	333,08	42482,4	4883040	135,640	20346	0,565
	ElGamal	1024	25,6	40,751	20,349	86,7	879840	24,440	3666	0,102
	ElGamal	2048	153,7	305,684	153,601	612,985	6613704	183,714	27557,1	0,765
	ElGamal	4096	1185,6	2364,47	1186,48	4736,55	51133680	1420,380	213057	5,918

Tabla 4. Resultados de los algoritmos asimétricos

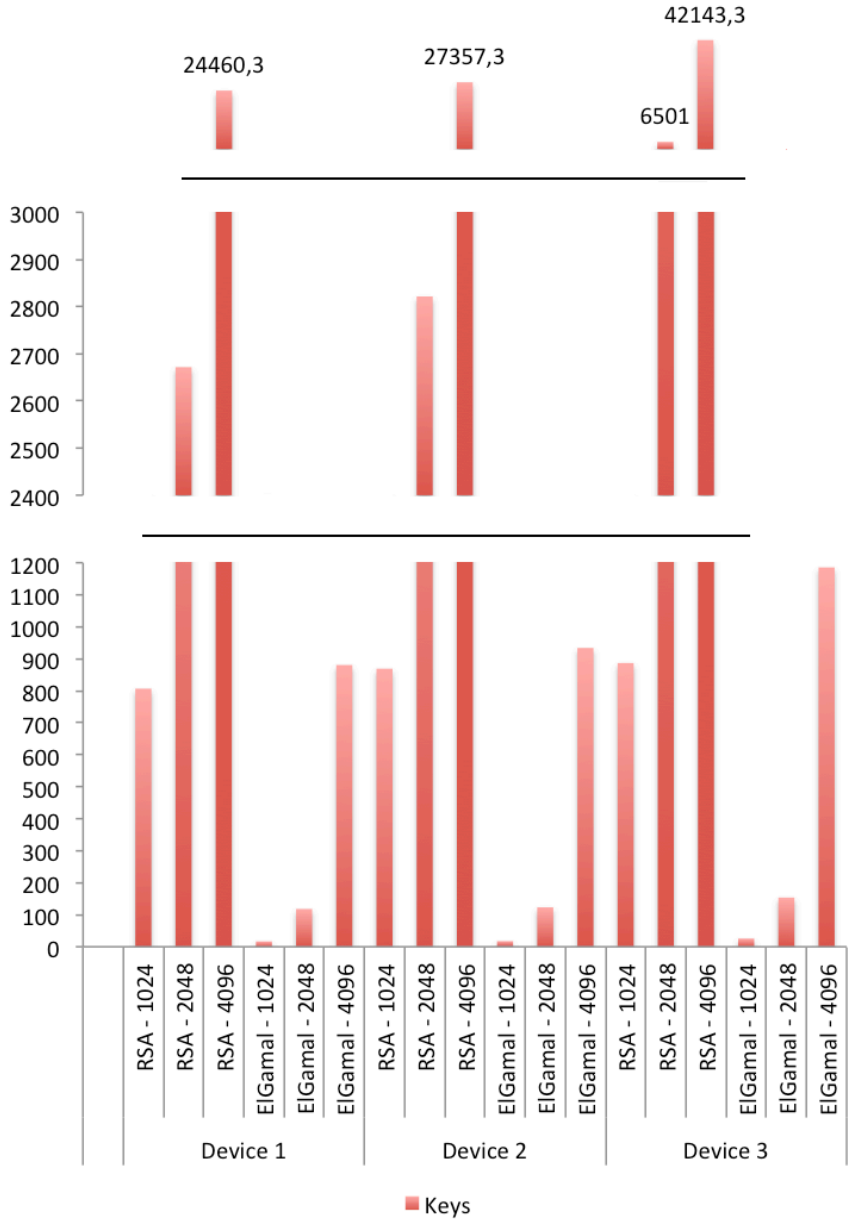


Ilustración 11. Gráfica de resultados de generación de clave en los algoritmos simétricos

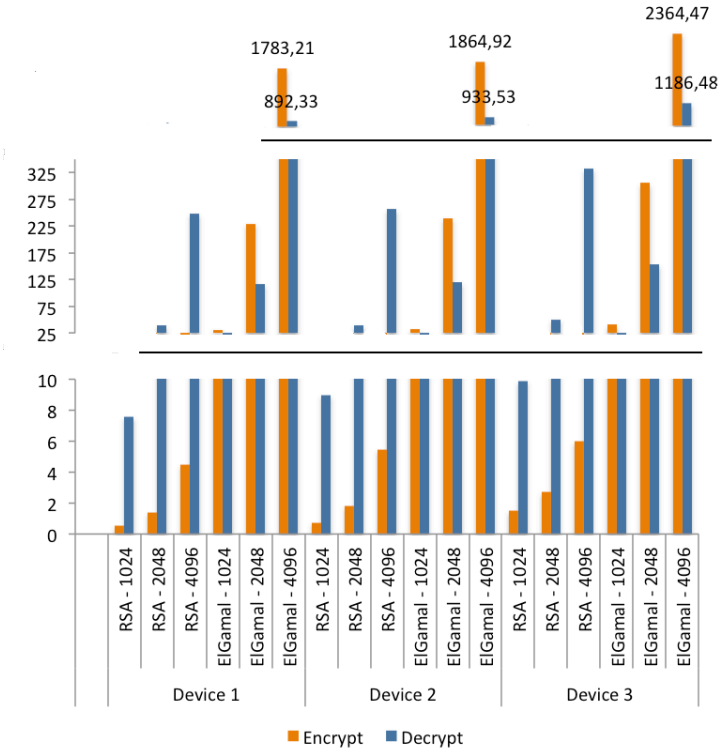


Ilustración 12. Gráfica de resultados de encriptar/desencriptar en los algoritmos simétricos

Analizando la Tabla 4 y la Ilustración 11 e Ilustración 12, nosotros podemos sugerir las siguientes interpretaciones:

- Los tiempos de generado de par de claves del algoritmo RSA son mucho superiores a ElGamal, situándose aún más en el extremo según aumenta el tamaño de clave seleccionado. RSA llega a emplear una media de 31,3 segundos en el caso más costoso, de 4096 bits, frente a 1 segundo de media de ElGamal, para el mismo tamaño de clave.
- Sin embargo, como se puede comprobar en la Ilustración 12 las cifras de encriptado y desencriptado de RSA son menores en todos los casos, por lo que se evalúa el coste en una hora de comunicación para ver finalmente que algoritmo tiene un mayor consumo.
- El porcentaje medio consumido en una hora de comunicación para nodos críticos de RSA para 1024, 2048 y 4096 bits es de 3,9%, 18,2% y 113,9% respectivamente. Por su parte, ElGamal, emplea 20,71%, 155,6% y 1203,3% para un tamaño de 1024, 2048 y 4096 bits respectivamente.
- En una hora de comunicación para nodos no críticos RSA emplea el 0,02%, 0,08% y el 0,5% para claves de 1024, 2048 y 4096 bits. Por su parte, ElGamal emplea, el 0,09%, 0,65% y el 5% para claves de 1024, 2048 y 4096 bits.
- De ello se deduce que para ElGamal no podría ser empleado con claves de 2048 y 4096 bits, ya que para las comunicaciones entre nodos críticos, excede el tiempo de cálculo encriptando y desencriptando en una hora, por lo que no sería viable. Lo mismo sucede con RSA para su versión de 4096 bits.
- Por otra parte, RSA presentaba un coste inicial muy superior para el cálculo de clave. Debido a que este coste solo se computa una vez, al registro del dispositivo en la plataforma, es menos relevante que las comunicaciones continuadas, en las que presenta valores mucho menores que ElGamal.

Nuestra recomendación, en función de los valores estudiados, es la utilización del algoritmo RSA con 2048 bits para nodos no críticos, dónde la comunicación es menor, y emplearía solamente el 0,08% del tiempo encriptando y desencriptando, ofreciendo un buen rendimiento. Para nodos críticos, dónde la comunicación es muy continuada, enviando hasta cuatro mensajes por minuto, la mejor opción, en nuestra opinión, pasaría por usar el algoritmo RSA con una longitud de clave de 1024 bits, que aunque ofrezca una seguridad menor al ser menos resistente a ser descubierta por fuerza bruta, ofrece un rendimiento mucho más ágil para este tipo de envío, con un gasto del 3.9% en el procesado de los mensajes.

Funciones Hash

A continuación mostramos los resultados de las pruebas de las diferentes funciones Hash. El objetivo de garantizar la integridad del mensaje. Para ello hemos decidido generar un resumen del mismo, para que el usuario receptor pueda comprobar que el mensaje que ha recibido coincide con el original que se le había enviado, haciendo el resumen del mensaje original y comprobando si coincide con el que ha recibido. Esto le permitirá discernir si ha sido o no manipulado durante el envío. Para ello deberemos seleccionar entre algunos de los algoritmos Hash existentes, el que mejor se adapte a nuestras necesidades en términos de consumo. Hemos realizado las pruebas con los algoritmos: MD5, SHA-1 y SHA-2 (256 y 512bits). Otros algoritmos como MD6, fueron descartados por no encontrar una librería que aportara implementación en Java.

Devices	Algorithms	Hash	1 hour communication			
			Critical	%	Non Critical	%
Device 1	MD5	0,2129	3065,76	0,08516	12,774	0,0004
	SHA-1	0,2551	3673,44	0,10204	15,306	0,0004
	SHA2-256	0,3976	5725,44	0,15904	23,856	0,0007
	SHA2-512	0,7836	11283,84	0,31344	47,016	0,0013
	SHA3-256	0,3437	4949,28	0,13748	20,622	0,0006
	SHA3-512	0,6197	8923,68	0,24788	37,182	0,0010
Device 2	MD5	0,2686	3867,84	0,10744	16,116	0,0004
	SHA-1	0,3023	4353,12	0,12092	18,138	0,0005
	SHA2-256	0,485	6984	0,194	29,1	0,0008
	SHA2-512	0,9227	13286,88	0,36908	55,362	0,0015
	SHA3-256	0,4374	6298,56	0,17496	26,244	0,0007
	SHA3-512	0,7676	11053,44	0,30704	46,056	0,0013
Device 3	MD5	0,3848	5541,12	0,15392	23,088	0,0006
	SHA-1	0,4605	6631,2	0,1842	27,63	0,0008
	SHA2-256	0,6977	10046,88	0,27908	41,862	0,0012
	SHA2-512	1,3104	18869,76	0,52416	78,624	0,0022
	SHA3-256	0,6737	9701,28	0,26948	40,422	0,0011
	SHA3-512	1,0228	14728,32	0,40912	61,368	0,0017

Tabla 5. Resultados de las funciones Hash

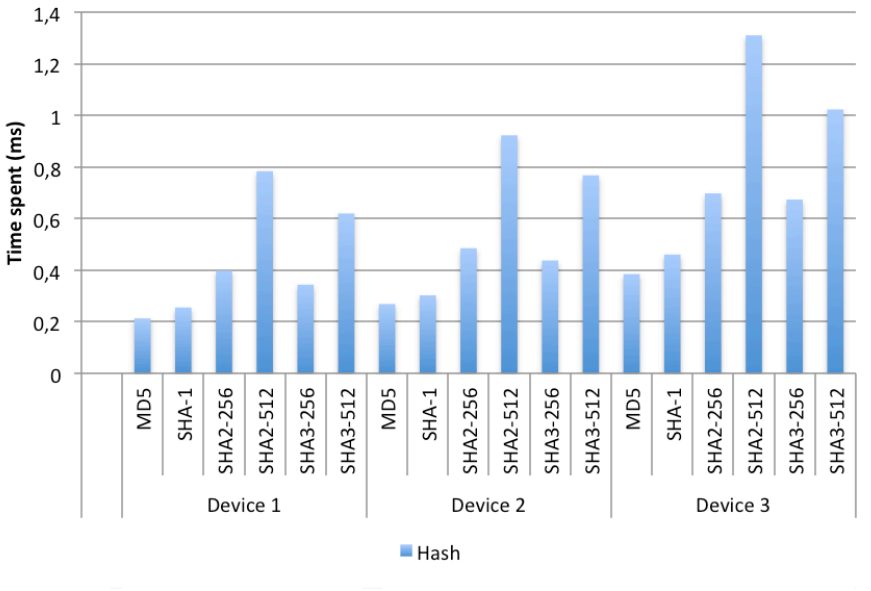


Ilustración 13. Gráfica de resultados de las funciones Hash

Analizando la Tabla 5 y la Ilustración 13, nosotros podemos sugerir las siguientes interpretaciones:

- El algoritmo MD5 con un tamaño de 128 bits es más rápido, en términos generales, que el resto.
- SHA-1 con un tamaño de 160 bits es sólo un poco más lento que MD5.
- SHA-2 para 256 y 512 bits, presenta unos tiempos superiores a los de SHA-3 para los mismos tamaños de claves.
- SHA-1 emplea el 0,1% en la comunicación de una hora para nodos críticos y el 0,0006% para nodos no críticos.
- SHA-2 emplea el 0,2% y el 0,4% para un tamaño de 256 y 512 bits en nodos críticos, y el 0,0009% y el 0,002% para un tamaño de 256 y 512 bits en nodos no críticos.
- SHA-3 emplea el 0,2% y el 0,3% para un tamaño de 256 y 512 bits en nodos críticos, y el 0,0008% y el 0,001% para un tamaño de 256 y 512 bits en nodos no críticos

Algunos problemas de seguridad presentes en estas funciones hash:

- MD5 es la opción más rápida de las evaluadas, pero presenta problemas debido a la colisión hash [53], por lo que no resulta la opción más segura.
- En 2005 se detectaron fallos de seguridad en SHA-1, por lo que no se recomienda su uso debido a la colisión [54]. Por ello, surgió SHA-2.
- En 2011 un ataque rompió la seguridad de SHA-2 [55], para 256 y 512 bits.

Por lo tanto, la decisión final es utilizar SHA-3, que presenta un rendimiento superior y un coste más bajo que SHA-2. En este caso podemos utilizar SHA-3 con 512 bits, ya que ofrece una seguridad mayor debido al tamaño y el consumo, 0,3% para comunicación entre nodos críticos y 0,001% para comunicación entre nodos no críticos, es bajo.

Conclusiones y Trabajo Futuro

Conclusiones

En este artículo nosotros presentamos una propuesta de seguridad para la Internet of Things, implementada en la plataforma de IoT Midgar, con el propósito de buscar el mayor grado de seguridad en el envío de los mensajes entre objetos en la red e intentando perder el menor tiempo de computo posible.

La solución pasa por adoptar las medidas características de la criptografía híbrida combinándola con la firma digital para así obtener los beneficios de cada una. Con esto obtendremos la confidencialidad y privacidad necesarias para el intercambio de mensajes de ámbito privado y añadimos integridad y autenticación, conceptos necesarios en una red IoT.

El estudio evalúa varios algoritmos de entre los tipos de criptografía seleccionada, con el objetivo de encontrar la solución más rápida posible, que consuma el menor tiempo y almacenamiento en un entorno tan cambiante de objetos con prestaciones tan diferentes, pero siempre manteniendo la seguridad necesaria. En este estudio se observa que la mejor opción es el uso de **AES**, **RSA** y **SHA3**.

Por un lado, en la criptografía simétrica **AES** para un tamaño de **256 bits** obtiene los mejores resultados tanto en tiempo de encriptado/desencriptado, dónde emplea el **0,21%** de las comunicaciones en una hora en nodos críticos y un **0,0009%** en la comunicación de nodos no críticos, como en términos de seguridad por tamaño de clave, dónde tarda en generar una clave de 256 bits una media de **0,04 ms**.

Respecto a la criptografía asimétrica, RSA es más rápido que ElGamal en una comunicación prolongada, cómo se espera que sean en un entorno IoT, ya que ElGamal llega a emplear más de el doble de tiempo que RSA pese a un tiempo de cálculo de claves inicial muy superior. RSA requiere una media de **0,9 ms** y de **4 s** para el cálculo de claves para un tamaño de **1024** y **2048 bits** respectivamente. Para una hora de comunicación entre nodos críticos, dónde se usaría la versión de **1024 bits**, RSA consumiría el **3,9%** del tiempo en tareas de encriptado/desencriptado. En la comunicación de una hora entre nodos no críticos RSA, dónde se emplearía la versión de **2048 bits**, emplearía únicamente el **0,08%** del tiempo.

Finalmente, para resumir el mensaje y garantizar su integridad, se emplearía la función Hash SHA3 para un tamaño de **512 bits**, dónde la función consumiría el **0,3%** del tiempo en las comunicaciones de una hora entre nodos críticos y el **0,001%** en nodos no críticos.

Esta seguridad, que usa de forma conjunta los algoritmos **RSA, AES y SHA3**, tendría un impacto final de **4.4%** en el consumo de una hora en nodos críticos, y un **0,08%** en nodos no críticos.

Sin embargo, hay que recalcar que en el caso de RSA, este posee un alto tiempo de generación automática del par de claves, pero debido a que dicho cálculo sólo se realiza en una ocasión, este puede ser asumido por los dispositivos. Mientras, en el caso del encriptado y desencriptado, al ser realizados en cada intercambio de mensaje, estos tiempos son menores

para 1024 bits, en un 16.8% menos que ElGamal en nodos críticos y un 0,07% en nodos no críticos. En 2048 bits, ElGamal requeriría un 137,4% más en nodos críticos, y un 0,57% en no críticos. Por ello, la elección de RSA frente ElGamal.

No obstante, como se ve en los estudios, se puede llegar a consumir bastante tiempo dependiendo del algoritmo utilizado y, aún seleccionando el que menor tiempo consuma manteniendo la seguridad, un 4,4% en nodos críticos es bastante tiempo, sobretodo, en dispositivos de gama baja. No obstante, esto es algo necesario para mantener la seguridad y privacidad de los objetos cuando estos la requieran porque aunque cierta información personal pueda parecer irrelevante, otros datos como la ubicación o las cuentas bancarias, pueden suponer un gran peligro en caso de ser suplantadas.

Trabajo Futuro

En este estudio sólo tratamos el envío de datos seguros en una red insegura entre Smart Objects. Esto da lugar a que queden muchos caminos abiertos para seguir investigando y/o mejorando el presente trabajo. Algunas líneas futuras de investigación podrían ser:

- **Mejora en la computación o el procesamiento de los algoritmos**: permitir así mantener una arquitectura igual o similar pero mejorando considerablemente los resultados en tiempo de consumo dependiendo de la viabilidad y los contrastes de las mejoras realizadas.
- **Registro seguro**: introducción de registro seguro para los objetos que se quieran registrar en una red IoT manteniendo la privacidad de este registro.
- **Optimización del envío de datos de la plataforma**: ahora se envían grandes cantidades de información, que podrían ser reducidas. De esta forma se reduciría el tiempo de encriptación y desencriptación de los mensajes y por tanto se agilizarían las comunicaciones.
- **Ataques de replay**: otro problema en la IoT son los ataques de repetición. Para ello, es necesario crear una solución rápida y de bajo costo.

Difusión de los Resultados

Para elegir la revista a la que se enviaría el artículo se hizo un análisis de las revistas existentes, para ello nos basamos en las revistas de las que habíamos leído artículos de temas relevantes o similares.

Ilustración 14. Computer Communications

- **Acceso**: http://www.journals.elsevier.com/computer-communications/
- **Editorial**: Elsevier.
- **Índice de impacto**: 1.695.
- **Índice de impacto en 5 años**: 1.625.
- **Categorías:** COMPUTER SCIENCE, INFORMATION SYSTEMS

Ilustración 15. Computer Networks

- **Acceso**: http://www.journals.elsevier.com/computer-networks/
- **Editorial**: Elsevier.
- **Índice de impacto**: 1.256.
- **Índice de impacto en 5 años**: 1.714.
- **Categorías:** COMPUTER SCIENCE, HARDWARE & ARCHITECTURE, INFORMATION SYSTEMS

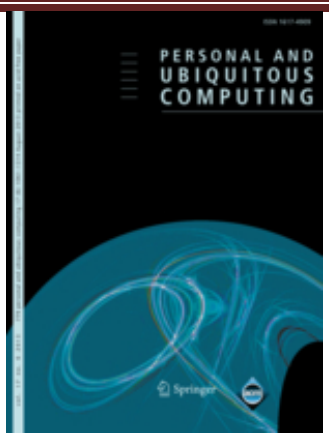


Ilustración 16. Personal and Ubiquitous Computing

- **Acceso**: http://www.springer.com/computer/hci/journal/779
- **Editorial**: Springer.
- **Índice de impacto**: 1.518.
- **Índice de impacto en 5 años**: 1.577.
- **Categorías:** COMPUTER SCIENCE, INFORMATION SYSTEMS

Finalmente la elección estuvo entre **Computer Communications** por su Special Issue[1] sobre IoT, y **Computer Networks**, que también tenía un Special Issue sobre el tema[2]. La decisión fue la revista **Computer Communications** por tener el mayor índice de impacto con **1.695**.

[1] Computer Communications: **http://www.journals.elsevier.com/computer-communications/call-for-papers/special-issue-on-the-internet-of-things-research-challenges/**

[2] Computer Networks: http://www.journals.elsevier.com/computer-networks/call-for-papers/special-issue-on-industrial-technologies-and-applications-fo/

Presupuesto

Planificación

El presupuesto se hizo acorde a la planificación del proyecto que se realizó previa al desarrollo del mismo. En primer lugar la planificación era la siguiente.

1	✓		**Inicio del proyecto.**	**16,8 días**	**vie 28/11/14**	**vie 12/12/14**
11	✓		**Investigación previa.**	**22,4 días?**	**vie 12/12/14**	**vie 02/01/15**
26			**Estudio del estado del arte**	**17,6 días?**	**mar 20/01/15**	**mar 03/02/15**
45			Reunión II - Estado del arte y enfoque	1,6 días	vie 20/02/15	vie 20/02/15
46			**Alcance del proyecto**	**5,8 días**	**vie 20/02/15**	**mié 25/02/15**
52			**Diseño del prototipo**	**10,6 días**	**jue 26/02/15**	**vie 06/03/15**
57			**Planificación del proyecto**	**73,6 días?**	**vie 28/11/14**	**lun 02/02/15**
79			**Implementación del prototipo**	**120,8 días?**	**mar 03/02/15**	**mar 19/05/15**
131			Reunión III - Implementación del prototipo	1,6 días	lun 18/05/15	lun 18/05/15
132			**Implantación del sistema.**	**4,8 días?**	**mar 19/05/15**	**jue 21/05/15**
136			Reunión IV - Evaluación de las pruebas obtenidas	1,6 días	jue 21/05/15	jue 21/05/15
137			**Documentación**	**41,6 días?**	**mar 28/04/15**	**mié 03/06/15**
158			**Cierre del proyecto**	**4,2 días**	**mié 03/06/15**	**vie 05/06/15**
161			Presentación del proyecto	1,6 días	lun 08/06/15	lun 08/06/15

Ilustración 17. Planificación inicial

Por motivos personales se demoró el inicio del proyecto, por lo que la planificación cambio del siguiente modo. Siendo la siguiente, la planificación final del proyecto.

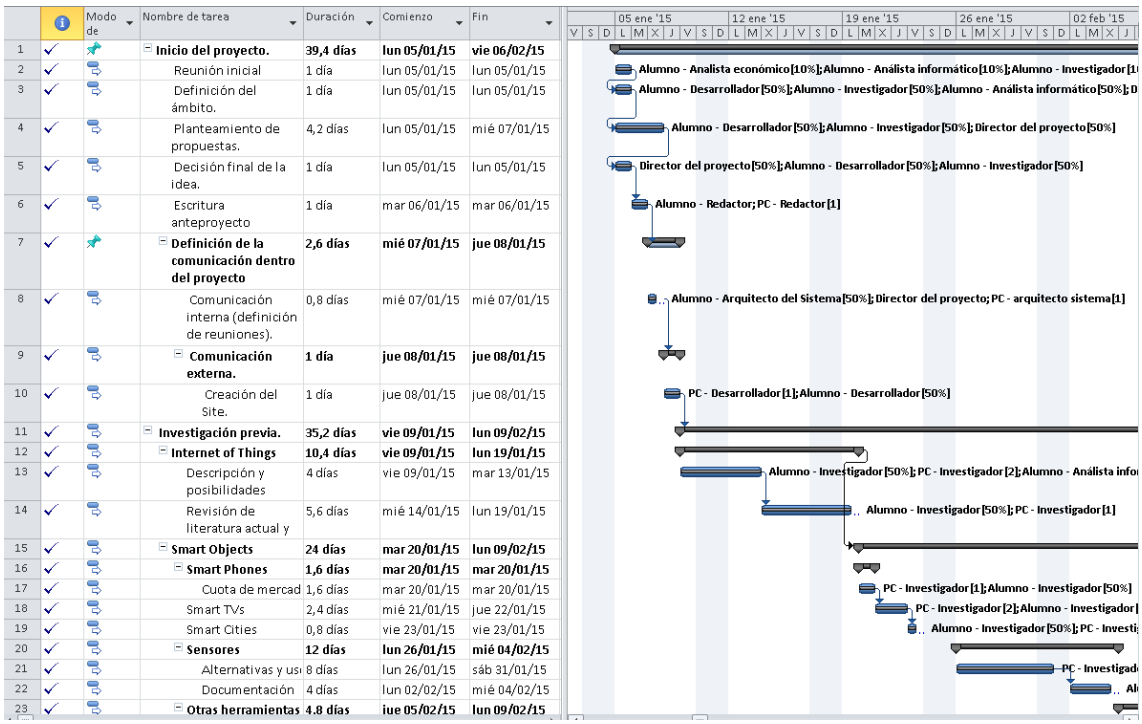
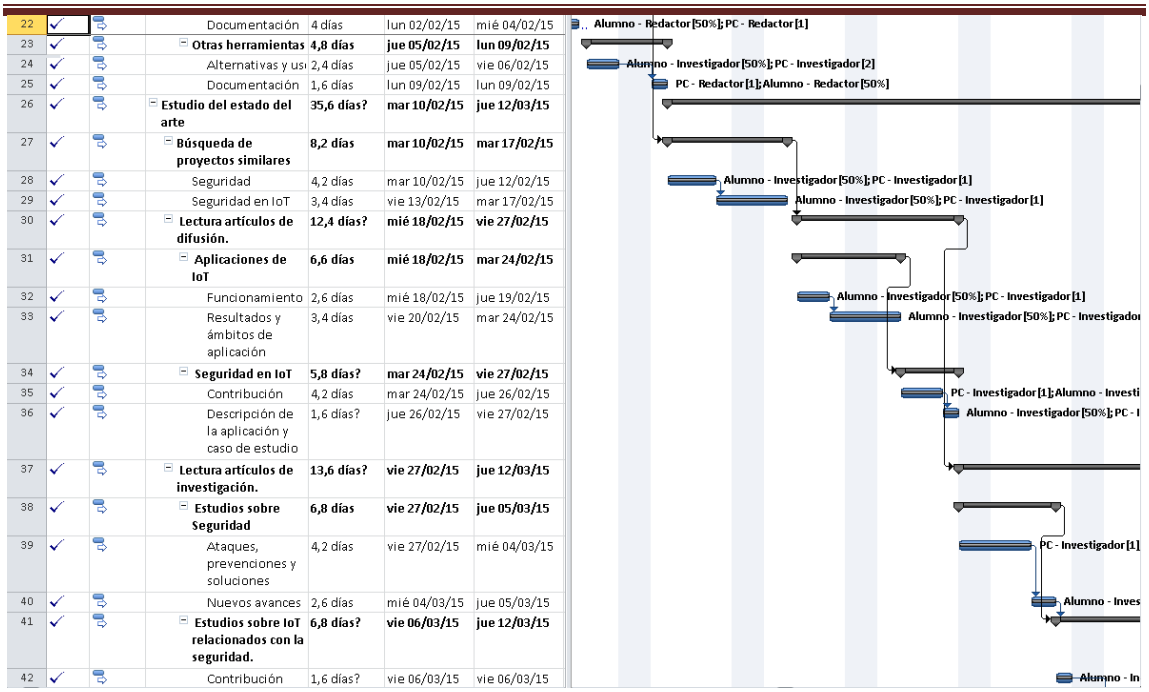


Ilustración 18. Planificación del proyecto (I)

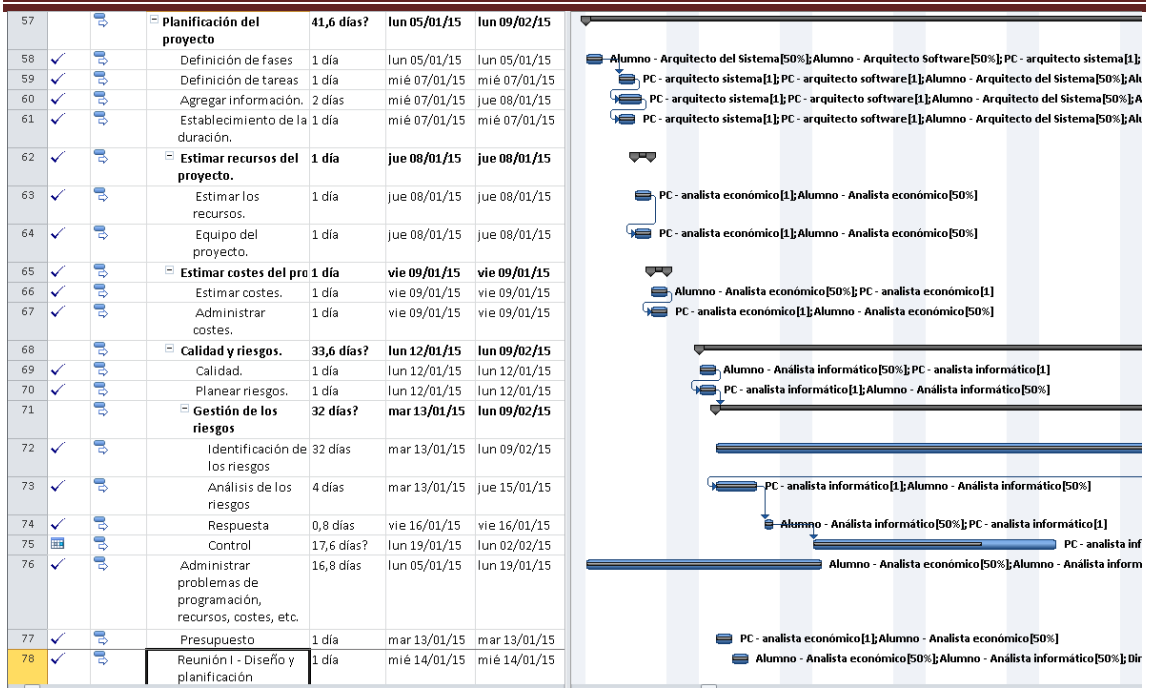


22	✓		Documentación	4 días	lun 02/02/15	mié 04/02/15
23	✓		**Otras herramientas**	**4,8 días**	**jue 05/02/15**	**lun 09/02/15**
24	✓		Alternativas y usa	2,4 días	jue 05/02/15	vie 06/02/15
25	✓		Documentación	1,6 días	lun 09/02/15	lun 09/02/15
26	✓		**Estudio del estado del arte**	**35,6 días?**	**mar 10/02/15**	**jue 12/03/15**
27	✓		**Búsqueda de proyectos similares**	**8,2 días**	**mar 10/02/15**	**mar 17/02/15**
28	✓		Seguridad	4,2 días	mar 10/02/15	jue 12/02/15
29	✓		Seguridad en IoT	3,4 días	vie 13/02/15	mar 17/02/15
30	✓		**Lectura artículos de difusión.**	**12,4 días?**	**mié 18/02/15**	**vie 27/02/15**
31	✓		**Aplicaciones de IoT**	**6,6 días**	**mié 18/02/15**	**mar 24/02/15**
32	✓		Funcionamiento	2,6 días	mié 18/02/15	jue 19/02/15
33	✓		Resultados y ámbitos de aplicación	3,4 días	vie 20/02/15	mar 24/02/15
34	✓		**Seguridad en IoT**	**5,8 días?**	**mar 24/02/15**	**vie 27/02/15**
35	✓		Contribución	4,2 días	mar 24/02/15	jue 26/02/15
36	✓		Descripción de la aplicación y caso de estudio	1,6 días?	jue 26/02/15	vie 27/02/15
37	✓		**Lectura artículos de investigación.**	**13,6 días?**	**vie 27/02/15**	**jue 12/03/15**
38	✓		**Estudios sobre Seguridad**	**6,8 días**	**vie 27/02/15**	**jue 05/03/15**
39	✓		Ataques, prevenciones y soluciones	4,2 días	vie 27/02/15	mié 04/03/15
40	✓		Nuevos avances	2,6 días	mié 04/03/15	jue 05/03/15
41	✓		**Estudios sobre IoT relacionados con la seguridad.**	**6,8 días?**	**vie 06/03/15**	**jue 12/03/15**
42	✓		Contribución	1,6 días?	vie 06/03/15	vie 06/03/15

Ilustración 19. Planificación del proyecto (II)

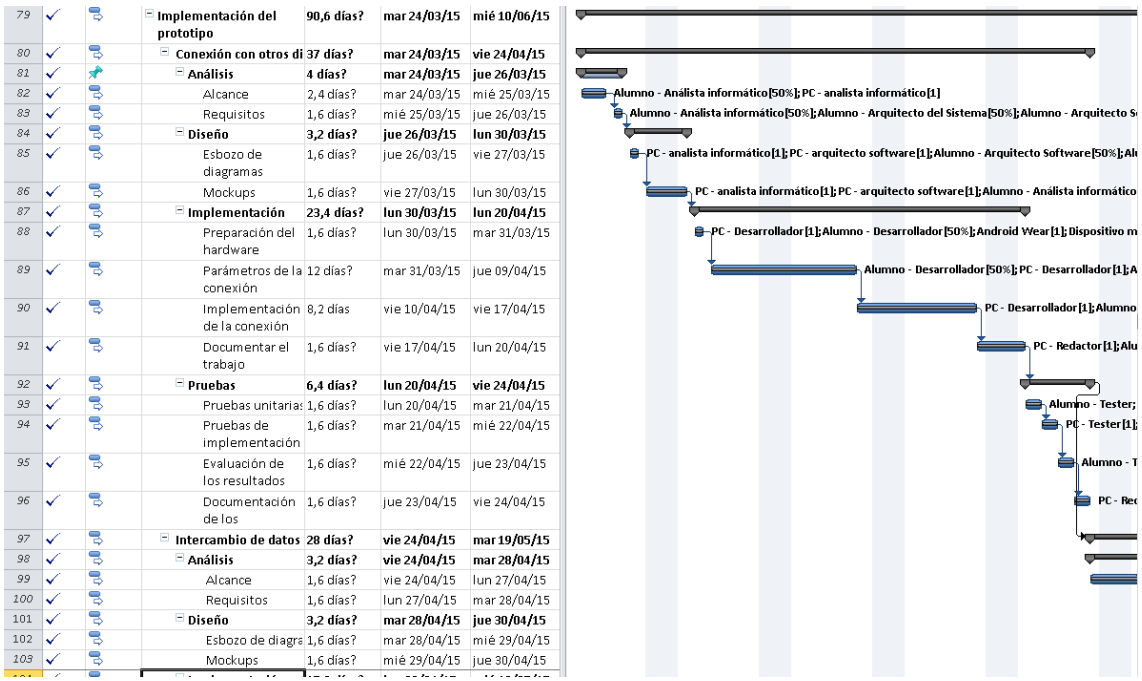
38	✓		**Estudios sobre Seguridad**	**6,8 días**	**vie 27/02/15**	**jue 05/03/15**
39	✓		Ataques, prevenciones y soluciones	4,2 días	vie 27/02/15	mié 04/03/15
40	✓		Nuevos avances	2,6 días	mié 04/03/15	jue 05/03/15
41	✓		**Estudios sobre IoT relacionados con la seguridad.**	**6,8 días?**	**vie 06/03/15**	**jue 12/03/15**
42	✓		Contribución	1,6 días?	vie 06/03/15	vie 06/03/15
43	✓		Ámbito	2,6 días	lun 09/03/15	mar 10/03/15
44	✓		Descripción y resultados	2,6 días	mar 10/03/15	jue 12/03/15
45	✓		Reunión II - Estado del arte y enfoque	1,6 días	mar 10/03/15	mar 10/03/15
46	✓		**Alcance del proyecto**	**98,6 días**	**mié 11/03/15**	**jue 04/06/15**
47	✓		Objetivo del proyecto	1 día	mié 11/03/15	mié 11/03/15
48	✓		Limitaciones del proyecto	1 día	mié 11/03/15	jue 12/03/15
49	✓		Materiales empleados	1 día	jue 12/03/15	jue 12/03/15
50	✓		**Estudio de viabilidad**	**1,6 días?**	**vie 13/03/15**	**vie 13/03/15**
51	✓		Estimación de la carga de trabajo y costes	1,6 días?	vie 13/03/15	vie 13/03/15
52	✓		**Diseño del prototipo**	**11 días**	**lun 16/03/15**	**mar 24/03/15**
53	✓		Definición del sistema.	5 días	lun 16/03/15	jue 19/03/15
54	✓		Definición de requisitos.	2 días	jue 19/03/15	vie 20/03/15
55	✓		Identificación de susbsistemas.	2 días	vie 20/03/15	lun 23/03/15
56	✓		Diseño de plan de pruebas.	2 días	lun 23/03/15	mar 24/03/15

Ilustración 20. Planificación del proyecto (III)



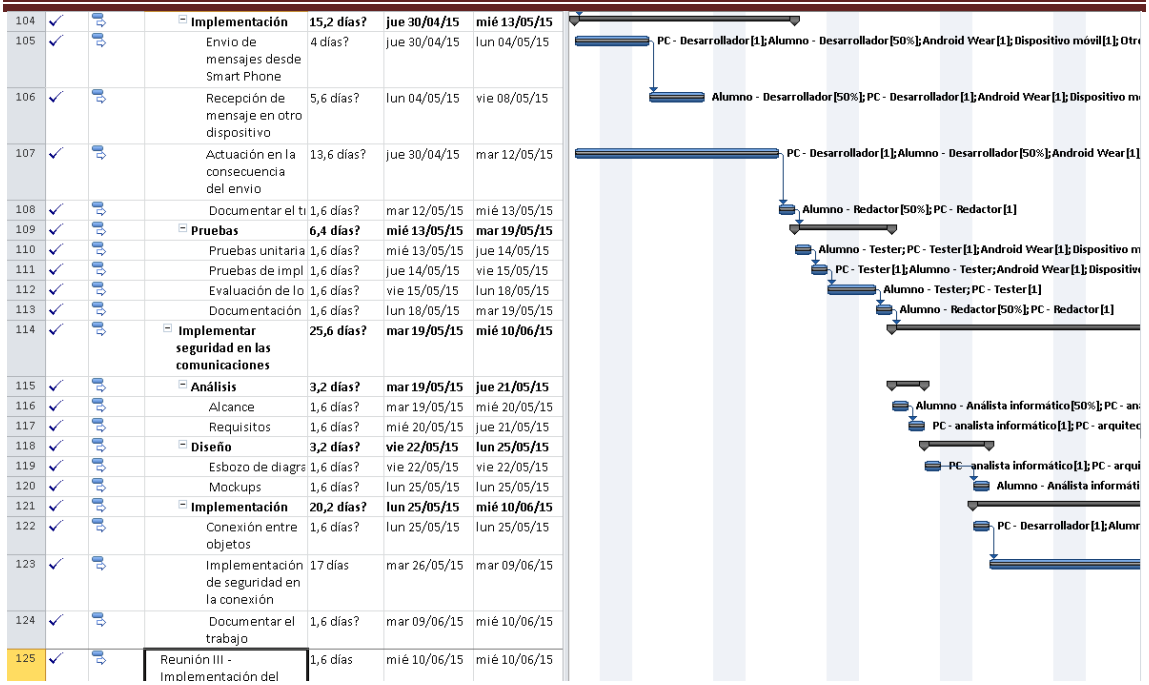
57			**Planificación del proyecto**	**41,6 días?**	**lun 05/01/15**	**lun 09/02/15**
58	✓		Definición de fases	1 día	lun 05/01/15	lun 05/01/15
59	✓		Definición de tareas	1 día	mié 07/01/15	mié 07/01/15
60	✓		Agregar información.	2 días	mié 07/01/15	jue 08/01/15
61	✓		Establecimiento de la duración.	1 día	mié 07/01/15	mié 07/01/15
62	✓		**Estimar recursos del proyecto.**	**1 día**	**jue 08/01/15**	**jue 08/01/15**
63	✓		Estimar los recursos.	1 día	jue 08/01/15	jue 08/01/15
64	✓		Equipo del proyecto.	1 día	jue 08/01/15	jue 08/01/15
65	✓		**Estimar costes del pro**	**1 día**	**vie 09/01/15**	**vie 09/01/15**
66	✓		Estimar costes.	1 día	vie 09/01/15	vie 09/01/15
67	✓		Administrar costes.	1 día	vie 09/01/15	vie 09/01/15
68			**Calidad y riesgos.**	**33,6 días?**	**lun 12/01/15**	**lun 09/02/15**
69	✓		Calidad.	1 día	lun 12/01/15	lun 12/01/15
70	✓		Planear riesgos.	1 día	lun 12/01/15	lun 12/01/15
71			**Gestión de los riesgos**	**32 días?**	**mar 13/01/15**	**lun 09/02/15**
72	✓		Identificación de los riesgos	32 días	mar 13/01/15	lun 09/02/15
73	✓		Análisis de los riesgos	4 días	mar 13/01/15	jue 15/01/15
74	✓		Respuesta	0,8 días	vie 16/01/15	vie 16/01/15
75			Control	17,6 días?	lun 19/01/15	lun 02/02/15
76	✓		Administrar problemas de programación, recursos, costes, etc.	16,8 días	lun 05/01/15	lun 19/01/15
77	✓		Presupuesto	1 día	mar 13/01/15	mar 13/01/15
78	✓		Reunión I - Diseño y planificación	1 día	mié 14/01/15	mié 14/01/15

Ilustración 21. Planificación del proyecto (IV)



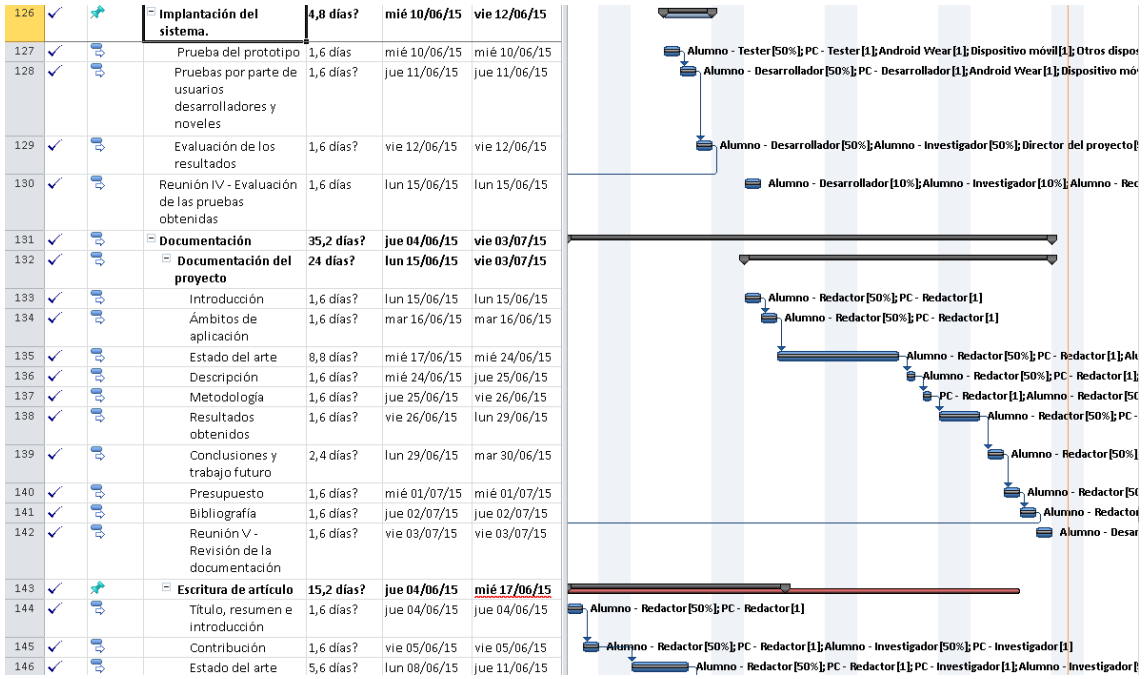
79	✓		**Implementación del prototipo**	**90,6 días?**	**mar 24/03/15**	**mié 10/06/15**
80	✓		**Conexión con otros di**	**37 días?**	**mar 24/03/15**	**vie 24/04/15**
81	✓		**Análisis**	**4 días?**	**mar 24/03/15**	**jue 26/03/15**
82	✓		Alcance	2,4 días?	mar 24/03/15	mié 25/03/15
83	✓		Requisitos	1,6 días?	mié 25/03/15	jue 26/03/15
84	✓		**Diseño**	**3,2 días?**	**jue 26/03/15**	**lun 30/03/15**
85	✓		Esbozo de diagramas	1,6 días?	jue 26/03/15	vie 27/03/15
86	✓		Mockups	1,6 días?	vie 27/03/15	lun 30/03/15
87	✓		**Implementación**	**23,4 días?**	**lun 30/03/15**	**lun 20/04/15**
88	✓		Preparación del hardware	1,6 días?	lun 30/03/15	mar 31/03/15
89	✓		Parámetros de la conexión	12 días?	mar 31/03/15	jue 09/04/15
90	✓		Implementación de la conexión	8,2 días	vie 10/04/15	vie 17/04/15
91	✓		Documentar el trabajo	1,6 días?	vie 17/04/15	lun 20/04/15
92	✓		**Pruebas**	**6,4 días?**	**lun 20/04/15**	**vie 24/04/15**
93	✓		Pruebas unitarias	1,6 días?	lun 20/04/15	mar 21/04/15
94	✓		Pruebas de implementación	1,6 días?	mar 21/04/15	mié 22/04/15
95	✓		Evaluación de los resultados	1,6 días?	mié 22/04/15	jue 23/04/15
96	✓		Documentación de los	1,6 días?	jue 23/04/15	vie 24/04/15
97	✓		**Intercambio de datos**	**28 días?**	**vie 24/04/15**	**mar 19/05/15**
98	✓		**Análisis**	**3,2 días?**	**vie 24/04/15**	**mar 28/04/15**
99	✓		Alcance	1,6 días?	vie 24/04/15	lun 27/04/15
100	✓		Requisitos	1,6 días?	lun 27/04/15	mar 28/04/15
101	✓		**Diseño**	**3,2 días?**	**mar 28/04/15**	**jue 30/04/15**
102	✓		Esbozo de diagra	1,6 días?	mar 28/04/15	mié 29/04/15
103	✓		Mockups	1,6 días?	mié 29/04/15	jue 30/04/15

Ilustración 22. Planificación del proyecto (V)



104	✓		**Implementación**	**15,2 días?**	**jue 30/04/15**	**mié 13/05/15**
105	✓		Envio de mensajes desde Smart Phone	4 días?	jue 30/04/15	lun 04/05/15
106	✓		Recepción de mensaje en otro dispositivo	5,6 días?	lun 04/05/15	vie 08/05/15
107	✓		Actuación en la consecuencia del envio	13,6 días?	jue 30/04/15	mar 12/05/15
108	✓		Documentar el tr	1,6 días?	mar 12/05/15	mié 13/05/15
109	✓		**Pruebas**	**6,4 días?**	**mié 13/05/15**	**mar 19/05/15**
110	✓		Pruebas unitaria	1,6 días?	mié 13/05/15	jue 14/05/15
111	✓		Pruebas de impl	1,6 días?	jue 14/05/15	vie 15/05/15
112	✓		Evaluación de lo	1,6 días?	vie 15/05/15	lun 18/05/15
113	✓		Documentación	1,6 días?	lun 18/05/15	mar 19/05/15
114	✓		**Implementar seguridad en las comunicaciones**	**25,6 días?**	**mar 19/05/15**	**mié 10/06/15**
115	✓		**Análisis**	**3,2 días?**	**mar 19/05/15**	**jue 21/05/15**
116	✓		Alcance	1,6 días?	mar 19/05/15	mié 20/05/15
117	✓		Requisitos	1,6 días?	mié 20/05/15	jue 21/05/15
118	✓		**Diseño**	**3,2 días?**	**vie 22/05/15**	**lun 25/05/15**
119	✓		Esbozo de diagra	1,6 días?	vie 22/05/15	vie 22/05/15
120	✓		Mockups	1,6 días?	lun 25/05/15	lun 25/05/15
121	✓		**Implementación**	**20,2 días?**	**lun 25/05/15**	**mié 10/06/15**
122	✓		Conexión entre objetos	1,6 días?	lun 25/05/15	lun 25/05/15
123	✓		Implementación de seguridad en la conexión	17 días	mar 26/05/15	mar 09/06/15
124	✓		Documentar el trabajo	1,6 días?	mar 09/06/15	mié 10/06/15
125	✓		Reunión III - Implementación del	1,6 días	mié 10/06/15	mié 10/06/15

Ilustración 23. Planificación del proyecto (VI)



126	✓		**Implantación del sistema.**	**4,8 días?**	**mié 10/06/15**	**vie 12/06/15**
127	✓		Prueba del prototipo	1,6 días	mié 10/06/15	mié 10/06/15
128	✓		Pruebas por parte de usuarios desarrolladores y noveles	1,6 días?	jue 11/06/15	jue 11/06/15
129	✓		Evaluación de los resultados	1,6 días?	vie 12/06/15	vie 12/06/15
130	✓		Reunión IV - Evaluación de las pruebas obtenidas	1,6 días	lun 15/06/15	lun 15/06/15
131	✓		**Documentación**	**35,2 días?**	**jue 04/06/15**	**vie 03/07/15**
132	✓		**Documentación del proyecto**	**24 días?**	**lun 15/06/15**	**vie 03/07/15**
133	✓		Introducción	1,6 días?	lun 15/06/15	lun 15/06/15
134	✓		Ámbitos de aplicación	1,6 días?	mar 16/06/15	mar 16/06/15
135	✓		Estado del arte	8,8 días?	mié 17/06/15	mié 24/06/15
136	✓		Descripción	1,6 días?	mié 24/06/15	jue 25/06/15
137	✓		Metodología	1,6 días?	jue 25/06/15	vie 26/06/15
138	✓		Resultados obtenidos	1,6 días?	vie 26/06/15	lun 29/06/15
139	✓		Conclusiones y trabajo futuro	2,4 días?	lun 29/06/15	mar 30/06/15
140	✓		Presupuesto	1,6 días?	mié 01/07/15	mié 01/07/15
141	✓		Bibliografía	1,6 días?	jue 02/07/15	jue 02/07/15
142	✓		Reunión V - Revisión de la documentación	1,6 días?	vie 03/07/15	vie 03/07/15
143	✓		**Escritura de artículo**	**15,2 días?**	**jue 04/06/15**	**mié 17/06/15**
144	✓		Título, resumen e introducción	1,6 días?	jue 04/06/15	jue 04/06/15
145	✓		Contribución	1,6 días?	vie 05/06/15	vie 05/06/15
146	✓		Estado del arte	5,6 días?	lun 08/06/15	jue 11/06/15

Ilustración 24. Planificación del proyecto (VII)

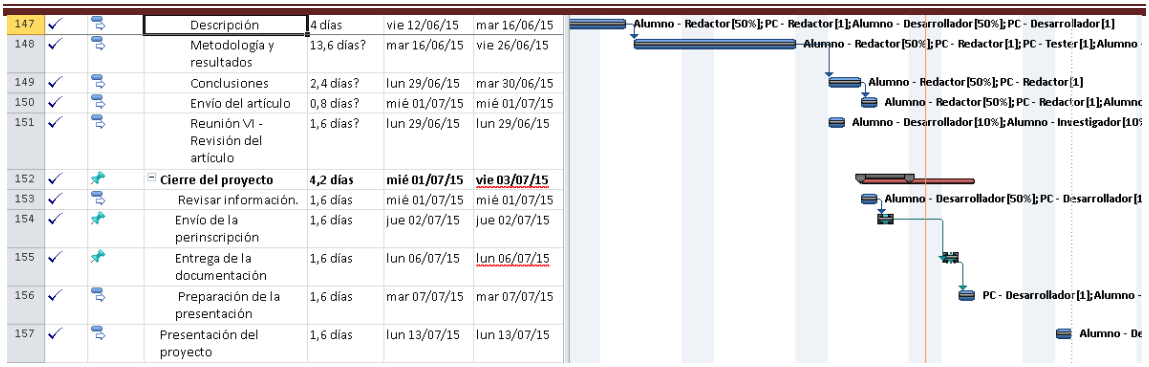


Ilustración 25. Planificación del proyecto (VIII)

Presupuesto

En función de la planificación el presupuesto del proyecto quedaría de la siguiente manera.

Recursos humanos

Nombre del recurso	Iniciales	Tasa estándar (€/h)	Horas trabajadas	Coste total recurso
Alumno - Análista informático	AI	25,00 €	256	6.400,00 €
Alumno - Arquitecto Software	ASO	30,00 €	76	2.280,00 €
Alumno - Arquitecto del Sistema	ASI	30,00 €	54	1.620,00 €
Alumno - Desarrollador	DE	20,00 €	224,7	4.494,00 €
Alumno - Tester	T	15,00 €	94	1.410,00 €
Director del proyecto	DP	50,00 €	32,5	1.625,00 €
Alumno - Analista económico	AE	15,00 €	72,5	1.087,50 €
Alumno - Investigador	I	25,00 €	215,7	5.392,50 €
Alumno - Redactor	R	10,00 €	172,2	1.722,00 €
			Coste total	26.031,00 €

Recursos materiales

Nombre del recurso	Tasa estándar	Horas utilizado	Coste total
PC - analista económico	10,00 €	11	110,00 €
PC - analista informático	10,00 €	23	230,00 €
PC - arquitecto software	10,00 €	17	170,00 €

PC - arquitecto sistema	10,00 €	12	120,00 €
PC - Desarrollador	10,00 €	17	170,00 €
PC - Tester	10,00 €	10	100,00 €
PC - Investigador	10,00 €	26	260,00 €
PC - Redactor	10,00 €	24	240,00 €
Dispositivo móvil	30,00 €	15	450,00 €
Otros dispositivos y sensores	30,00 €	17	510,00 €
		Coste total	2.360,00 €

Resumen del presupuesto

Recurso	Coste	IVA (%)	Coste total recurso
Humano	26.031,00 €	21,00%	31.497,51 €
Material	2.360,00 €	21,00%	2.855,60 €
Coste total	28.391,00 €	Coste total (+ iva)	34.353,11 €

Bibliografía

[1] F. Telefónica, La Sociedad de la Información en España 2014, Grupo Planeta Spain, 2015.

[2] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, Commun. Mag. IEEE. 40 (2002) 102–114.

[3] Y. Wang, G. Attebury, B. Ramamurthy, A survey of security issues in wireless sensor networks, (2006).

[4] M. Chui, M. Löffler, R. Roberts, The internet of things, McKinsey Q. 2 (2010) 1–9.

[5] H. Gu, D. Wang, A content-aware fridge based on RFID in smart home for home-healthcare, in: Adv. Commun. Technol. 2009. ICACT 2009. 11th Int. Conf., IEEE, 2009: pp. 987–990.

[6] L. Atzori, A. Iera, G. Morabito, The Internet of Things: A survey, Comput. Networks. 54 (2010) 2787–2805. doi:10.1016/j.comnet.2010.05.010.

[7] C. González García, B.C. Pelayo G-Bustelo, J. Pascual Espada, G. Cueva-Fernandez, Midgar: Generation of heterogeneous objects interconnecting applications. A Domain Specific Language proposal for Internet of Things scenarios, Comput. Networks. 64 (2014) 143–158. doi:10.1016/j.comnet.2014.02.010.

[8] G. Kortuem, F. Kawsar, D. Fitton, V. Sundramoorthy, Smart objects as building blocks for the internet of things, Internet Comput. IEEE. 14 (2010) 44–51.

[9] K.A. Hribernik, Z. Ghrairi, C. Hans, K. Thoben, Co-creating the Internet of Things - First Experiences in the Participatory Design of Intelligent Products with Arduino, (2011) 1–9.

[10] J. Bohn, V. Coroamă, M. Langheinrich, F. Mattern, M. Rohs, Living in a World of Smart Everyday Objects—Social, Economic, and Ethical Implications, Hum. Ecol. Risk Assess. An Int. J. 10 (2004) 763–785. doi:10.1080/10807030490513793.

[11] A. Bröring, P. Maué, C. Malewski, K. Janowicz, Semantic mediation on the sensor web, in: Geosci. Remote Sens. Symp. (IGARSS), 2012 IEEE Int., IEEE, 2012: pp. 2910–2913.

[12] S.H. Park, S.H. Won, J.B. Lee, S.W. Kim, Smart home–digitally engineered domestic life, Pers. Ubiquitous Comput. 7 (2003) 189–196.

[13] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson, A. Oliveira, Smart Cities and the Future Internet: Towards Cooperation Frameworks for Open Innovation., Futur. Internet Assem. 6656 (2011) 431–446.

[14] E. Borgia, The Internet of Things vision : Key features , applications and open issues, Comput. Commun. 54 (2014) 1–31. doi:10.1016/j.comcom.2014.09.008.

[15] Harmony Institute, Purposeful storytelling and designing, 2013.

[16] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, G. Vassilacopoulos, Enabling data protection through PKI encryption in IoT m-Health devices, IEEE 12th Int. Conf. Bioinforma. Bioeng. BIBE 2012. (2012) 25–29. doi:10.1109/BIBE.2012.6399701.

[17] N.I.C. NIC, Disruptive Civil Technologies: Six Technologies with Potential Impacts on US Interests out to 2025, (2008).

[18] P.N. Mahalle, N.R. Prasad, R. Prasad, Novel Threshold Cryptography-based Group Authentication (TCGA) Scheme for the Internet of Things (IoT), (2015).

[19] H. Suo, J. Wan, C. Zou, J. Liu, Security in the internet of things: A review, Proc. - 2012 Int. Conf. Comput. Sci. Electron. Eng. ICCSEE 2012. 3 (2012) 648–651. doi:10.1109/ICCSEE.2012.373.

[20] Martin Feldhofer, S. Dominikus, J. Wolkerstorfer, Strong Authentication for RFID Systems Using the AES Algorithm, Cryptogr. Hardw. Embed. Syst. - CHES 2004. 3156 (2004) 357–370. doi:10.1007/b99451.

[21] B. Calmels, S. Canard, M. Girault, H. Sibert, Low-cost cryptography for privacy in RFID systems, Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 3928 LNCS (2006) 237–251. doi:10.1007/11733447_17.

[22] O. Savry, F. Pebay-Peyroula, F. Dehmas, G. Robert, J. Reverdy, RFID Noisy Reader How to Prevent from Eavesdropping on the Communication?, Springer, 2007.

[23] J. Hernández-Ramos, A. Jara, Distributed Capability-based Access Control for the Internet of Things, J. Internet …. (2013) 1–16. http://isyou.info/jisis/vol3/no34/jisis-2013-vol3-no34-01.pdf.

[24] P. Mahalle, B. Anggorojati, Identity establishment and capability based access control (IECAC) scheme for Internet of Things, … (WPMC), 2012 15th …. (2012) 184–188. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6398758.

[25] P. Mahalle, Identity Authentication and Capability Based Access Control (IACAC) for the Internet of Things, J. Cyber …. 1 (2013) 309–348. http://forskningsbasen.deff.dk/Share.external?sp=S72df67ac-3ea5-41da-8cf0-523c6f71bbf2&sp=Saau.

[26] S. Gusmeroli, S. Piccione, D. Rotondi, A capability-based security approach to manage access control in the Internet of Things, Math. Comput. Model. 58 (2013) 1189–1205. doi:10.1016/j.mcm.2013.02.006.

[27] T. Polk, S. Turner, Security challenges for the internet of things, … Interconnecting Smart Objects with Internet, …. (2011). https://www.iab.org/wp-content/IAB-uploads/2011/03/Turner.pdf.

[28] C.P. Mayer, Security and privacy challenges in the internet of things, Electron. Commun. EASST. 17 (2009).

[29] a. a. Nyre, A survey on security in mobile ad hoc networks, Secur. Commun. Networks (IWSCN), 2009 Proc. 1st Int. Work. (2009).

[30] L. Da Vinci, Leonardo's Notebooks: Writing and Art of the Great Master, Black Dog & Leventhal, 2013.

[31] M. Agrawal, P. Mishra, A comparative survey on symmetric key encryption techniques, Intern. J. Comput. Sci. Eng. 4 (2012) 877–882. http://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=82397469&site=ehost-live.

[32] L. Tan, N. Wang, Future internet: The internet of things, in: Adv. Comput. Theory Eng. (ICACTE), 2010 3rd Int. Conf., IEEE, 2010: pp. V5–376.

[33] D. Miorandi, S. Sicari, F. De Pellegrini, I. Chlamtac, Internet of things: Vision, applications and research challenges, Ad Hoc Networks. 10 (2012) 1497–1516.

[34] R. Acharya, K. Asha, Data integrity and intrusion detection in wireless sensor networks, in: Networks, 2008. ICON 2008. 16th IEEE Int. Conf., IEEE, 2008: pp. 1–5.

[35] R. Roman, P. Najera, J. Lopez, Securing the Internet of things, Computer (Long. Beach. Calif). 44 (2011) 51–58. doi:10.1109/MC.2011.291.

[36] G. Kortuem, F. Kawsar, V. Sundramoorthy, D. Fitton, Smart objects as building blocks for the internet of things, (2009). doi:10.1109/MIC.2009.143.

[37] G.G. Meyer, K. Främling, J. Holmström, Intelligent Products: A survey, Comput. Ind. 60 (2009) 137–148. doi:10.1016/j.compind.2008.12.005.

[38] G. Zhao, X. Si, J. Wang, X. Long, T. Hu, A novel mutual authentication scheme for Internet of Things, Proc. 2011 Int. Conf. Model. Identif. Control. (2011) 563–566. doi:10.1109/ICMIC.2011.5973767.

[39] G. Gan, Z. Lu, J. Jiang, Internet of Things Security Analysis, 2011 Int. Conf. Internet Technol. Appl. (2011) 1–4. doi:10.1109/ITAP.2011.6006307.

[40] L. Eschenauer, V.D. Gligor, A key-management scheme for distributed sensor networks, 9th ACM Conf. Comput. Commun. Secur. (2002) 41–47.

[41] N. Ye, Y. Zhu, R.C. Wang, R. Malekian, Q.M. Lin, An efficient authentication and access control scheme for perception layer of internet of things, Appl. Math. Inf. Sci. 8 (2014) 1617–1624. doi:10.12785/amis/080416.

[42] Z. Hu, The research of several key question of internet of things, in: Intell. Sci. Inf. Eng. (ISIE), 2011 Int. Conf., IEEE, 2011: pp. 362–365.

[43] G. V Lioudakis, E. Koutsoloukas, N. Dellas, S. Kapellaki, G.N. Prezerakos, D. Kaklamani, et al., A proxy for privacy: the discreet box, in: EUROCON, 2007. Int. Conf. &# 34; Comput. as a Tool&# 34;, IEEE, 2007: pp. 966–973.

[44] E. Mykletun, J. Girao, D. Westhoff, Public key based cryptoschemes for data concealment in wireless sensor networks, IEEE Int. Conf. Commun. 5 (2006) 2288–2295. doi:10.1109/ICC.2006.255111.

[45] O. Bergmann, S. Gerdes, C. Bormann, Simple Keys for Simple Smart Objects, Smart Object Secur. …. (2012). http://www.tschofenig.priv.at/sos-papers/OlafBergmann.pdf\nhttp://tools.ietf.org/html/draft-gilger-smart-object-security-workshop-02.

[46] M.E. Smid, D.K. Branstad, Data Encryption Standard: past and future, Proc. IEEE. 76 (1988) 550–559. doi:10.1109/5.4441.

[47] H. Kummert, The PPP Triple-DES Encryption Protocol (3DESE), (1998).

[48] D. Selent, Advanced encryption standard, 6 (2010) 1–14.

[49] M.-P. Leong, O.Y.H. Cheung, K.H. Tsoi, P.H.W. Leong, A bit-serial implementation of the international data encryption algorithm IDEA, in: Field-Programmable Cust. Comput. Mach. 2000 IEEE Symp., IEEE, 2000: pp. 122–131.

[50] A. Mousa, Data encryption performance based on Blowfish, in: ELMAR, 2005. 47th Int. Symp., IEEE, 2005: pp. 131–134.

[51] D.R. Smith, J.T. Palmer, Universal fixed messages and the Rivest-Shamir-Adleman cryptosystem, Mathematika. 26 (1979) 44–52.

[52] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, in: Adv. Cryptol., Springer, 1985: pp. 10–18.

[53] M. Stevens, Fast Collision Attack on MD5., IACR Cryptol. ePrint Arch. 2006 (2006) 104.

[54] B. Schneier, Schneier on Security: Cryptanalysis of SHA-1, Schneier. Com. (2005).

[55] D. Khovratovich, C. Rechberger, A. Savelieva, Bicliques for preimages: attacks on Skein-512 and the SHA-2 family, in: Fast Softw. Encryption, Springer, 2012: pp. 244–263.

[56] C.J. James, S. Kumar, Detection of posture and motion by accelerometer sensors, in: Electron. Comput. Technol. (ICECT), 2011 3rd Int. Conf., IEEE, 2011: pp. 369–371.

[57] F. Foerster, M. Smeja, J. Fahrenberg, Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring, Comput. Human Behav. 15 (1999) 571–583.

[58] J. Mäntyjärvi, J. Himberg, T. Seppänen, Recognizing human motion with multiple acceleration sensors, in: Syst. Man, Cybern. 2001 IEEE Int. Conf., IEEE, 2001: pp. 747–752.

Anexos

Artículo

Midgar: Study of communications security among Smart Objects using a platform of heterogeneous devices for the Internet of Things

Gonzalo Sánchez-Arias [a]*, Cristian González García[a], Cristina Pelayo García-Bustelo[a]

[a] *University of Oviedo, Department of Computer Science, Sciences Building, C/Calvo Sotelo s/n 33007, Oviedo, Asturias, Spain. Tel: +34985103397*

[a] *uo204530@uniovi.es, gonzalezgarciacristian@hotmail.com, crispelayo@uniovi.es*

* Corresponding author.

ABSTRACT

In the last years, the Internet of Things has been a revolution in terms of applications and research. Currently, there is a great variety of nodes connected to one another to create different applications in fields, ranging from sport to business, among others. The nature of these applications compromises our private information about our bank accounts, health, and location. This makes us take safety measures to achieve a secure communication, where the interception of a message by means of a malicious user cannot compromise our privacy.

This security encompasses a very broad concept that can be addressed in multiple ways. This work focuses on the techniques and encryption algorithms to be used in the messages exchanged among the nodes on a secure Internet of Things network. To our case study, we have used the Midgar platform. In order to do this, we are going to evaluate the different possibilities of traditional security related to encryption. Afterwards, we are going to analyse the results to determine which one is the best solution, in terms of costs, and considering the different type of end devices.

Keywords: Internet of Things; Security; Cryptography; Ubiquitous Computing; Smart Objects;

1 Introduction

Nowadays, we can find a myriad of applications which, benefiting from the available technology, were created to make our daily lives easier [1,2]. Thanks to this, we have being able to monitor different kinds of data, allowing us to act consequently or to respond to the situation automatically. These only represent a little sample of all the different applications

that could be applied in our everyday life in the future [2]. In a general sense, these applications are the product of the rise of Smart Objects [3] and the great range of possibilities that they offer, allowing us to access and use personal information and data from our surroundings. Furthermore, users generally keep their mobile devices on all day, even during night-time, which means that the sensors integrated in the gadgets are still able to collect data: the user is inadvertently revelling information about where he is and what he is doing [4]. These possibilities exist due to the different sensors above-mentioned incorporated in our smartphones and our tablets [5], in addition to other technologies which appear or evolve in parallel with this revolution. I.e.: the use of sensors [6] or the new SmartBands. All of which is developed in the actual context of the Internet revolution, of the cloud computing, and of the large datasets processing through Big Data. The combination of this context and the new technologies make possible the Internet of Things (IoT).

The Internet of Things allows us to keep objects scattered all over the world in order to receive information in our homes, at work, in our car or even in the bottom of the ocean [7]. Those objects can explore their surroundings, communicate with other objects or with human beings, helping them to complete their tasks in new intuitive ways [1]. I.e.: the sensors placed in a particular infrastructure can offer information about the environmental conditions or detect someone trespassing, and it could act accordingly [5]. In that way, the connection between different types of nodes becomes possible in applications for sports or business related contexts, as dissimilar as they might be. So, those nodes can complete general tasks as controlling a thermostat, or more personal ones as collecting and reporting information about our health, our regular routes, our bank accounts or our location, among other data.

In [8], they place SmartBands around some users wrists while they assist as public to the narration of a story. As the plot develops, the users have to 'enter' the story, so they will be confronted with different situations and scenes, to which they have to react and even eventually make decisions. The goal is to collect data about the sensations the user has experiment during the test and to make a quantitative analysis. To gather the information, the SmartBands analyses the electrodermal changes of the subject's skin and it sends the data to other objects in order to identify moods of excitement, concentration or surprise. Those bracelets could be of use in many different contexts, i.e. to inform about the level of concentration of an employee while working or of a student during a lecture.

In [9] develop services in the benefit of people with special needs and/or the elderly, a system of wireless communication which allows an easy contact with their caretakers and the storage of data in the cloud. Another possible use could be to watch patients with chronic diseases [5]. Nevertheless, the nature of the application of this technology and of the data itself poses a problem about privacy, which must translate into the adoption of specific security systems to prevent further problems.

The advancements have facilitated a new vision where information from millions or even billions of devices can be collected, processed and exploited collaboratively within a global Internet of Things [10] and the personal character of this information jeopardise our intimacy and force us to take security measures in order to secure the communications without the risk of having one of our messages intercepted by malicious individuals. These security measures could be applied in different ways and, due to the novelty of the own IoT, they should be checked regularly having in mind other aspects that may have been overlooked previously

[11]. The nature of these vulnerabilities could be rather simple, as a physical attack as a result of the individual or lonely character of the nodes, which can lack of vigilance; or more technical, using the wireless communication to spy the channel used to transfer the data [2].

That is the reason why this research work is going to focus mainly on the communication between the objects and the transmission of a secure message through an insecure channel. To this purpose, we will look for the measures which can prevent the messages exchanged by the nodes in an IoT web from being violated by malicious users, in such way that, even though they intercept them, they won't be able to read them.

The resources we will be using consist on cryptographic techniques issued from the traditional security to encrypt the message sent at the moment of the communication. First of all, the techniques will be evaluated, depending on the kind of used cryptography. Then, we will analyse the cryptography of the secret or private key, the public or asymmetric key, and the hybrid one. After that, we will study the algorithm which could be used on each and every type above-mentioned in order to establish the best possible combination. We must perform this analysis phase because these techniques were not created to be used in IoT [12], and their process and use could present major impediments when applied to some devices.

This security measures should be relevant to the exchange of messages between the objects interconnected through the IoT, so they will be applied in the platform Midgar [7]. Such platform generates applications that interconnect heterogeneous objects to one another, collecting information about them and making them act accordingly, in a context where the security cannot be neglected.

This article is divided in the following way: in the section 2, we discuss the IoT and the new cryptographic methods; in the section 3, we show the case study to implement de security in the IoT; in the section 5, we present the results and the evaluations about the studied performances of the algorithms; in the section 6, the conclusions of this research work are exposed; in section 7, we describe the possible options which should be considered in future works.

2 State of the art

In the vision of the Internet of Things there are innumerable objects connected to the Internet, transferring information from anywhere in the world and connecting to one another. As it is said in [1], this vision is as futuristic as it is troubling, because this way, all of our movements, actions and decisions are continually being recorded by every electronic device we have all over our homes, as the kitchen or the living room, or the ones we use for travelling in our cars on weekends, as a GPS device or a smartphone. The emerging scenario IoT be formed of a group of heterogeneous devices with different capabilities [13], this nature allows any heterogeneous device to collect information about our daily lives, no matter its function, in order to act according to the data it has, and to improve our quality of life through the task automation or semi-automation. Nevertheless, this has a great inconvenient: most of our information risks of being exposed, no matter the importance of the data, from the content of our fridges to the information about our bank accounts. That is the reason why everyday

objects can jeopardise the security of our information, due to the fact that the IoT is able to transmit the risks more widely than the Internet was capable to till this day [2,14].

2.1 Internet of Things

Nowadays more and more objects integrate sensors and have the capacity of communicating. The networks built to support this communication will create new business models [5] and a future environment in which the information will be within reach and it will be always available. This will be possible thanks to some ubiquitous devices, which will be connected to one another and which will be offering whole new possibilities of communication [15]. Such future will actually be built as the technology evolves and the market demand grows [2]. This is what we called Internet of Things, that is to say, the interconnection of heterogeneous and ubiquitous objects through the Internet [2,7,16]. Their development is tied to the Internet and their basic function is to collect actions and data from the users.

In [2], also study the huge impact that the IoT will have on certain aspects of our daily lives and in the potential users' behaviour, i.e. home automation, assisted living, e-health, learning improvement, business and logistics automation.

Several examples of these are: the sensors integration in the gadgets will slowly promote the change in our daily lives: insurance companies could use the sensors of a car or install new ones to know how or where that vehicle is driven and the risks that driving could involve [5]; other example of this change in the daily activities, that would imply the automation of some fields is e-health, where a doctor could computerise in advance possible hospitalisations which were not previously planned thanks to the monitoring of daily aspects of users' lives, since in 2025 it is expected that there will be IoT nodes in every daily basis activities [14], which would reduced hospitalisation and treatment costs in a billion dollars per year, just in the United States [5]. The analysts expect the number of devices connected in a network IoT will reach 50 billion by 2020 [17,18].

All this innovations would be effective because of the incorporation of a electronic improvement in the daily physical objects, which will turn them into "smart things", allowing them to integrate the global infrastructure [5,15,16]. That is the reason why it is important to assure the security in those communications and the identity of the users [19].

2.2 Smart Objects

The Smart Objects are objects capable of interacting with one another and with the environment [20] through sensors and other elements integrated in the product to this end; they also have the sufficient level of intelligence to engage the process of making their own decisions, having in mind the data they have procured themselves through those communications. For instance, the smartphones, the tablets and the Smart TVs are good examples of Smart Objects that have a large presence in our daily lives.

According to [20], the Smart Objects can be classified in three types, regarding their intelligence. The aspects we should take into account are the own intelligence of the object; the origin of this intelligence, whether it is its own or it is due to the other objects on the network; and the own nature of the object, whether it is an individual thing which can only

obtain data or it is a set of devices which can work and process data even if one of the components isn't active.

We basically used Smart Objects, with a level of intelligence, management of the information and notification of the problem and with the location of the intelligence through the network. We do not use SmartTags cards such as Radio Frequency Identifications (RFID) or Near Field Communication (NFC) for its low processing power.

2.3 Security on the Internet of Things

It seems that the position of the Internet of Things in a near future and the parallel evolution of the technology, which hands us more and more possibilities, will force us to give priority to the IoT security system. That way, it is necessary to look for the best solutions in order to avoid the violations of sensitive and private information of the users and to avoid the leak of all their private lives.

The low processing capacity of some devices used in IoT, which can be identified by the low capacity regarding energy and computing resources [2], could endanger the security of the communication, turning it into the soft spot of the system [11]. Some devices involved in the communication are passive components, that is to say, they cannot implement complex schemes to support and enhance the security levels [2], as it is the case of the SmartTags. Accordingly to [11], as the information is transferred through the Internet the attacks are a serious problem, because it could make the said communication unavailable. That's the reason why those mechanisms should be improved and why we should focus in the preventive measures [11,21].

As it was mentioned above, it is necessary to find a safe method of sending the personal data of the users to maintain their privacy and, at the same time, provide a certain level of security for the Smart Objects on the IoT. In this section, we will explain the preventive measures which have been developed in other fields and which could be applied to the Internet of Things, allowing us to send safe messages through an unsafe environment.

This research work looks for a way to ensure the user's **confidentiality**. In order to do that, we will use cryptography. The cryptography is the main way of hiding information to reach this purpose, as it guarantee that only one person, the one who have the key, could be able to read the message, no matter who he is. This is a common way to make accessible certain information only to the authorised staff. In fact, it has already been used by Leonardo Da Vinci, in the Renaissance times [22].

Nevertheless, to allow the safe message exchange amongst different objects or people, it is necessary to establish a way to ensure the **authenticity** in the sensors [11]. That is how we can make the objects connected to the network to identify themselves in a unique and clear way, so they won't be supplant by a third party. The authentication is it difficult though, because it requires of authentication infrastructures adapted to the situation and servers which achieve this goal exchanging correct messages with other nodes [2]. In the context of the IoT, not only the users can access the data, but the authorised objects too. Therefore, it is necessary to define a mechanism to control the access and an authentication process [15].

Not every object has the same storage or process capacities, so not every one has the same conditions when using a technology. The RFID systems have very limited resources which rends impossible to generate and process the keys to the authentication [2]. These tags are mainly passive and they just answer the questions of the reader object charged of interacting with them. Their problem is that the attacker could intercept the data sent from a tag to an authorised reader [2] or he could just read a card that has no encrypted data on it. Accordingly to [2], the mainly solutions for this issue consist on the authentication of the authorised readers. However, for that to become real, we'll need tags capable of authenticate themselves and authentication scenarios with appropriate infrastructures and servers which will exchange messages with the nodes; all of which has elevated costs, far from the vision we have for the IoT.

Another important aspect when we want to transfer a safe message is its **integrity**. The integrity consists on the verification that the message is identical to when it was sent, with the same values and characteristics, because that is the way in which the receiver should get it. That way, we can also check that the message hasn't been modified by a third party, which could have intercepted it and forwarded it. The solutions to the matter of data integrity can guarantee that an opponent cannot change the information contained or that the final receiver can check if the message have been altered and, if so, eventually eliminate it. This problem has been studied before, in every traditional computer and communication systems [2,23].

Even so, one of the nodes could deny have received or sent the message. To avoid that, we require implementing the **non-repudiation**. This way, the communication can't be declined: the sender can't deny that the message was sent and the receiver in turn can't deny that he received it. Thanks to this we can guarantee the authorship of a message.

To finish off, the other main purpose of Smarts Objects' research is the **privacy**; we look to adopt mechanisms which can protect the privacy of the human beings and of the objects connected to the network [11]. This investigation [11] states that, in an environment as automated as ours, people can't be really conscious as to what point their information is exposed or their privacy could be violated, due to the use of sensors without knowing its limitations.

That's the reason why the privacy has become one of the principal problems of the IoT [2,11,24]. Accordingly to [24], the own nature of IoT and its exploitation through sensors have created entities as the "Big Brother" which study the users without their permission and which could create a world where the information about someone else could be at hand. And other type of information, i.e. regarding the environment or other automated management.

An environment that interconnects heterogeneous objects should treat encrypted data in a safe mode, in which only the authorised users could read it, even if a third party intercepts the information and want to use it. In this research, we propose a possible solution.

2.4 Criptography

More and more devices will get connected to the network in IoT [2,14]. The sensors distributed in a smart city could communicate to one another with smartphones, consequently, this channel needs security protocols which connect all these devices to the Internet [24]. In order

to establish this secure communication in IoT, we could use cryptography as a security measure.

There are different ways to encrypt a message, which can produce changes in its structure, in the type and quantity of the keys, its size, the capacity of process and storage necessary to complete the task, etc.

The **Secret Key** o **Symmetric Key Cryptography** is a method of data encrypting in which the sender and the receiver share the same key. Both parts should agree in the password they are going to use. After that, the sender will encrypt the message with it, while the receiver will just use the same key to decrypt it.

The strong point of this type of encrypting is the speed, so this method is much more employed when we need to encrypt large packages of data. On the contrary, it has other problems. The main one is the key distribution, because both users need to know it. If we send it through a non secure channel, the key could be intercepted and anyone could decrypt the message and get access to the content. Another problem is the key storage in a big network, since we would need a key for every node next to another one. There is also the problem about integrity on the exchange of the keys, since, in the case of being intercepted, the key could be used for a malicious user who could modify the message. To conclude, this method doesn't guarantee the non-repudiation or the authentication.

The **Public Key** or **Asymmetric Key Cryptography** is a data encrypting algorithm which uses some keys to send the messages: both sender and receiver have their own public and private key. The public key could be handed to all the nodes which we want to establish a connection with; the private key instead must be preserved and could only be accessed by the proprietary. When we want to send a message to a node, it has to encrypt it with that node's public key. To continue, the node will decrypt it with its personal key. This process turns the node into the only able to decrypt that message.

On the contrary, if it is done on the other sense, if we encrypt the message whit the private key, anyone could read the message, because everyone would have the sender's public key. This way, we create a **digital signature**, since the sender has his own private key, so he is the only one able to encrypt the data with it. We could even verify who sign it.

The strong point of this method is the distribution of the keys, especially if we compare it to the symmetric encryption, since we have eliminated the need of exchanging the key, which translates into a better security. On the other hand, it is a slower method than the symmetric one. Another advantage in its favour is the confidentiality; with this method we can assure that the receiver will be the only one to read it, since the message was encrypted with the user public key, it will be necessary to have the receiver's private key to read it, which only he possess. In despite of that, the problems of integrity, authentication and repudiation are still present.

The **Hybrid Key Cryptography** is a cryptographic method which uses a symmetric key and an asymmetric one. It needs a secret key, as it is necessary with the symmetric cryptography, and two keys, as it is required in the asymmetric cryptography. This way, when we desire to send a message to another user, it encrypts the message with the secret key, and this one is

encrypted with the public key of the node to which the message is sent. This node will decrypt the secret key with its own private key, which will be used to read the original message.

This method is faster than the asymmetric encryption, but it is slower than the symmetric one. As it happens in the Public Key Cryptography, in the Hybrid Key Cryptography we reach the desired level of confidentiality, as the receiver is the only one able to read it, but, in this case, we also reach the level of integrity, since the message cannot be modified. The authentication and repudiation problems are still presents in this system, since with this method we cannot guarantee the authorship of the message.

The **digital signature** is based on the sender's authentication, so the receiver could identify the entity which has created this electronic transaction, and so he could check that the message has not been modified. The operation is partially the same to the asymmetric and hybrid cryptography: both the sender and the receiver need to have their public and private keys. The sender will communicate its public key, so the receiver will know it and could use it. The sender will encrypt the message with his private key; as he is the only one to have it, this will guarantee the authorship of the message. Then he will use a hash function, sending it to the receiver accompanied of the original encrypted message. This user shall read the message using the sender's private key, to which he will apply the hash function, just to check that the message he has it is the same as the original, guaranteeing that it was not change on the transferring process.

Thanks to this method, we gain on authentication aspect, allowing identifying the original source of the message, since he is the only one with access to his private key, and consequently on the non-repudiation aspect too, since the sender won't be able to denied the authorship of the message.

On the other hand, this method has a similar problem to the Symmetric Cryptography one: we lose the privacy and the confidentiality.

A **hash function** is an algorithm which it is applied to a text chain to "sum up" the original fixed-length message, from which the original communication is impossible to obtain. A secure hash function has the following characteristics: the original message is impossible to obtain (one-way function); we obtain a fixed-length message which is inferior to the original in size terms (compression), it has to be easy to calculate (ease), and it has to be difficult to find two messages with the same output (collision resistance). In that way, when we apply a hash function to the original message, the receiver will be able to know if there has been any modification on the original input, which guarantees its integrity.

In the Table 6, we show the different types of cryptographic measures that could be used and we state whether they fulfil the necessary security measures above-mentioned, all of this to find the option which will provide us fewer inconveniences.

Cryptography type	Key distribution	Confidentiality and privacy	Authentication	Non-repudiation	Integrity
Symmetric	No	No	No	No	No

Asymmetric	Yes	Yes	No	No	No
Hybrid	Yes	Yes	No	No	Yes
Key exchange	Yes	Yes	No	No	Yes
Digital signature	Yes	No	Yes	Yes	Yes

Table 6. Cryptography type´s comparison

As we can observe, any of the analysed methods reach all of our security standards, forcing us to combine some of these process in order to obtain a system with all the five security measures. In this comparison we exclude important issues due to the IoT and the final environment where the system will be applied, for instance, the speed when operating it depending on the keys size, on the cryptography type used, on the algorithms, etc. This aspects will be treated on section 4.

2.5 Proposals for the Internet of Things Security in research

Nowadays, the IoT is developing its primary phase. We have long way for the research in this field, since *de* facto there is a literally blank space about his matter [11]. Every time there will be more nodes connected to the network, with particular and different characteristics and this gives as a result the problem of having limited and heterogeneous computing capacities of most of these objects. Due to the IoT and Smart Objects growth, the security issue will be the priority of the researchers [11].

The first of these problems is the password size. The principle is simple: the longer the password is, the safer it is, but it also gets heavier [2] and the computing necessities will be bigger. The password size depends on what type of cryptography is used. Symmetric algorithms, like Advanced Encryption Standard, (AES) are limited to passwords of 128-256bits, meanwhile with others, as Shamir y Adleman (RSA), is recommended to use bigger passwords, for instance a1024bits password offers a better security.

Accordingly to [2], the cryptographic algorithms spend a great quantity of resources in terms of energy and bandwidth. These solutions cannot be applied nether at the origin nor at the destination, so they cannot be used in the IoT at all, since they are going to include elements, as the RFID tags or sensor's nodes, which are limited precisely regarding energy capacity, communications and computing.

The traditional key exchange and the key distribution protocols based on infrastructures are not practical in a big scale network due to the unknown network's topology before the deployment [19]. All of this is applied to Smart Objects with limited capacities for limitation and storage, as we mentioned above [25].

Due to these problems we shall look for the best solution, the one that adapts better to the security measures required by the platform, with less consumption and with less computing necessities due to its use.

In [4], we perceive the cryptography as a useful mechanism to apply on sensors, since they are considered as a difficult aspect to secure, due to the presence of multiple nodes, which have reduce computing and storage capacity. These authors have created an authentication

protocol with only a safe variable key, using only one encryption method. In this case, it is necessary to create a key matrix between the sender and the receiver, also transferring the key coordinates instead of the own key. The authors use the case of a car's remote key, if a third person monitors the signal and records it, he could repeat it and open the door. One of the solutions proposed is that the matrix should be change frequently, so no one else knows it. Includes that RFID and Near Field Communication (NFC), one of the first technologies in the IoT, were not designed to this purpose, having secure communications.

In [25,26], we look for a simple and mutual authentication and to establish a safe key based in Elliptic Curve Cryptography (ECC). The advantage of this method is its session key, based in ECC, which has lower storage needs. We search control and access mechanism, at the sacrifice of cryptographic mechanisms, which used an encrypted access control to access the messages. This way we lower the consumption levels, so this is as encryption method more adequate to the IoT. Nevertheless, this solution cannot guarantee that someone else intercepts the message and modifies it.

Contrary to these measures based on the key calculation for the exchange and encryption, in [19] we assure the key predistribution as the only way of key distribution in an unknown connected sensors network. The keys should be storage in the sensor nodes to be able to establish a safe connectivity between them, based in the probability of the key exchange between the nodes of a random graph and uses a protocol of simple key sharing detection for the key distribution, revocation and rewrite node. This involves security problems due to the exchange of keys, and the integrity of the message sent is not guaranteed.

In [27], present a symmetric cryptographic system based in the algorithm AES to codify messages and identify RFID, due to the limited capacities of this card, in order to modify the protocol and propose another challenge-response system, which produces a secure cryptographic mechanism. This way in [28] argue that the high security solutions could use a RFID environment without affecting the present communication protocols, if we choose correctly and combine it with low cost cryptographic algorithms. They use a basic function of symmetric or asymmetric encryption, i.e. AES. Both proposition [27,28] look for a solution to the authentication problem in the case of RFID cards, since they have a low processing capacity and because the cards are passive and they just answer to the reader questions, so an attacker could intercept the answer from an authorised reader [2]. There are other possible solutions proposed to solve this matter of communication between the reader and the RFID cards, for instance [29], where the signal transferred by the reader is similar to a pseudo-noise. That signal is modulate by the RFID cards, so its transmission cannot be detected by malicious users. This method is still in a very early phase of its development [2]. It is an interesting solution to guarantee the safe exchange between the reader and the card, but we are not interested in this aspect of the IoT network.

In [30] use a jump encryption, in which every node can receive the message in plain text, so they need a high credibility in the nodes charge of the transmission. This method could be used in business whose security criteria are not too demanding, on the contrary, to guarantee the security it should be used a complete encryption [11,15].

There is a solution of a higher level, as in [31], which requires a proxy acting with the user and at the same time with the servers, in order to guarantee that the personal data collected are only used to the services authorised by the providers.

There are other solutions based on the public cryptography, which are more popular. We find it in [32] where they are used in a wireless network with a certain type of sensors of limited capacities. The problem presents itself when adapting the system to the sensor's capacity, the encrypted transfer cannot guarantee the integrity of the message, its authentication or its resistance to certain attacks.

In [33], the researchers were conscious of these menaces of the IoT, origination in the wireless network but also in the more physical form. That way, they propose an implementation of the following protocol Constrained Application Protocol (COAP), on the Datagram Transport Layer Security (DTLS). Its aim is to use a little implementation of the COAP, to avoid the energy consumption, and to adapt the DTLS to the Smart Objects limitations, measure destined to devices of a bigger capacity. They use the symmetric cryptographic algorithm AES in the communications due to the higher consumption of the asymmetric cryptography. The communication between the new nodes, still to be registered and the central node requires a third party who guarantees the authenticity of both. The key exchange and the pre-loaded keys and this last process make it vulnerable to an attack man-in-the-middle, which cannot be protected through a cryptographic system, but changing the physical limitation of the nodes and its potency. In our opinion, the use of hybrid cryptography and digital signature cannot avoid the man-in-the-middle attack, but it could reduce the risks in the communications, since a message intercepted could not be read or modified: the final receiver would know it thanks to the author sign and the message digest.

Article reference	Safe key distribution	Confidentiality and protection	Authentication	Non-repudiation	Integrity
[4]	X				
[26]	X	X	X	X	
[25]	X	X	X	X	
[27]		X			
[28]		X			
[30]	X	X			
[31]					
[32]	X	X			
[29]		X			X
[33]			X	X	

Table 7. Solution for the IoT in the present reseach.

In the Table 7 we show a summary of the analysis offered by some articles specialised on the field. Some of the solutions do not offer a solution for the authentication, the non-repudiation or the integrity issues, all of them characteristics of the digital signature; while there are other methods who offers us complex solutions, but which cannot guarantee the integrity or the privacy of the message. We will study a solution using cryptographic methods of the traditional security in order to meet the objectives.

3 Case Study

In this work, we will try to investigate about the security on the Internet of Things and to look for a solution which could provide us a better security without interfering with its performance. We know that the Smart Objects storage and computing capacity is quite limited nowadays. One of the possible solutions to this problem is for us to know better these systems' limitations. For this purpose, we will use a Midgar IoT network in order to test the level of security. Due to the low performance of the sensors and the RFID cards, we need an intermediary who takes the roll of reader: i.e. smartphones, as Arduino, which could be comparated to a high-end mobile device, or a raspberry pi, comparable to a superior-end mobile device. This research work will focus in the smartphones, since they are easier to use to make studies like ours, even though the platform where it is installed work with all the mentioned types.

3.1 Midgar platform

Midgar [7] is a platform which allows inexpert users to model and create applications which connect heterogeneous objects through the IoT, allowing to register in the IoT network any kind of device which is able of passing out the right message.

The platform has a register application, in which we could register the devices you wants to connect and exposes all its sensors. That way, we could keep a register in which we list all of the sensors of our mobile devices. Then, you could create applications which require them: localisation, accelerometer, etc.

After that, a user whit no previous knowledge of software development could model applications through the use of his Domain-Specific Language to interconnect the different devices registered.

That way, the devices are registered and possess a unique identification which difference them from the other devices on the network, but the communication, which it is established through the Internet, is made without any particular security measure. These, as we have mentioned before, should prevent the message to be read by other users, since they contain private information, and they should guarantee our security, being emitted by an authorised node and registered in the application.

To conclude the research done in the Midgar platform, it was necessary the following modifications:

Para incluir el trabajo realizado en la plataforma Midgar fueron necesarias las siguientes modificaciones:

- Modify the application Android, on the register and when sending information.
- Modify the Ruby on Rails server in order to receive and send information.
- Modify the BBDD MySQL fields to management the necessary keys.

3.2 Implementation

The solution that we propose is a combination of hybrid cryptography and digital signature, even though most of the authentication mapping is based in hash functions or public key

cryptography [26]. The aim of this research is to combine the benefits of both systems without including their disadvantages. The benefits of the hybrid cryptography are: confidentiality, privacy and message integrity; its soft spots are the message authorship and its non-repudiation. These characteristics will be compensated thanks to the digital signature, whose benefits are precisely the message authorship and its non-repudiation. In that way, this one will lose on privacy, confidentiality and integrity, aspects which will be compensated by the hybrid cryptography. Putting together the best qualities of both systems, we could guarantee all of the preventive measures which we need to reach the desired level of security in the IoT.

In order to do this prototype, we have to implement the following architecture, which is divided en four parts, regarding the function it is managing and the moment of the communication.

3.2.1 Register Object

In this section, we will modify the Midgar platform original application's performance, in which two new devices will be connected to the IoT network.

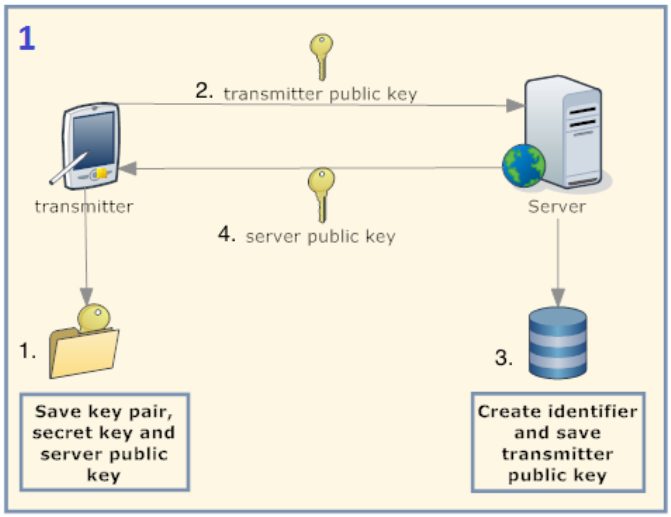


Figure 26. Register Object

The aim of this modification is to create the necessary keys to establish a safe communication in this precise moment, when the device is registered. To do this, we will proceed as follows: Figure 26.

1 The device will create a secret key and a pair of keys: one prívate, another public, and it will storage it in its internal memory.
2 When this device will be registered it will send the public key to the server.
3 The server will storage the user' public key, as well as the device's ID number.
4 In response to this, the server will send to the device its public key to maintain the future communications established between both of them.

4.1.1 Send messages to the server

The message exchange with the server is the principal scenario of the communication between two objects, whose aim is that the sender object sends its information to the server, and from

this the data arrive to the receiver object. For this to be possible, the following steps will be necessary. Figure 27.

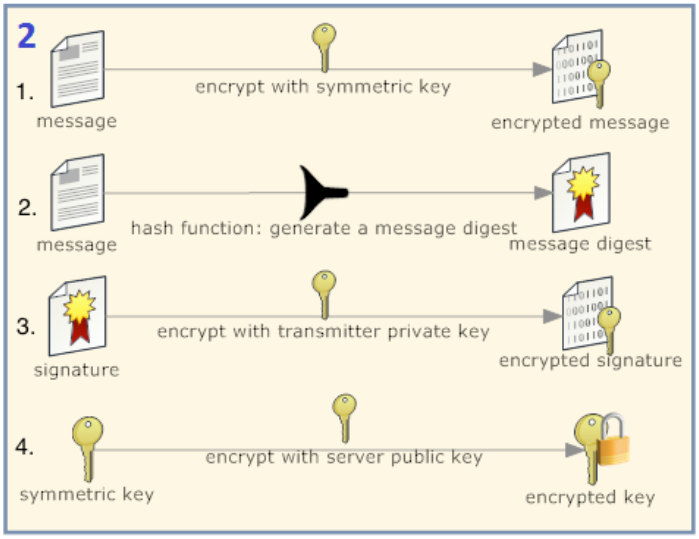


Figure 27. Sends message to the server

1. The sender encrypts the message and sends it with its symmetric key, which will be the only one which can be used to decrypt it.
2. It creates a message digest through a hash function, in order record a summary version of the message, with a smaller delivery cost, so we can also guarantee its integrity.
3. The sender will sign the message digest with its private key. It can be decrypted with its public key. This way, we can guarantee the authorship of the message, since he's the only one who possess his private key, consequently he's the only one who can use it to sign the message digest.
4. Finally, the sender encrypts the symmetric key with the public key of the server, so he could be the only one able to decrypt it, so the only one capable of accessing the content of the message.

4.1.2 Send messages to the receiver

In this case scenario, the server is the first receiver of the original message, which is sent by the sender, and it is charged of sending it to its final receiver. The following steps in this example are shown in the Figure 28.

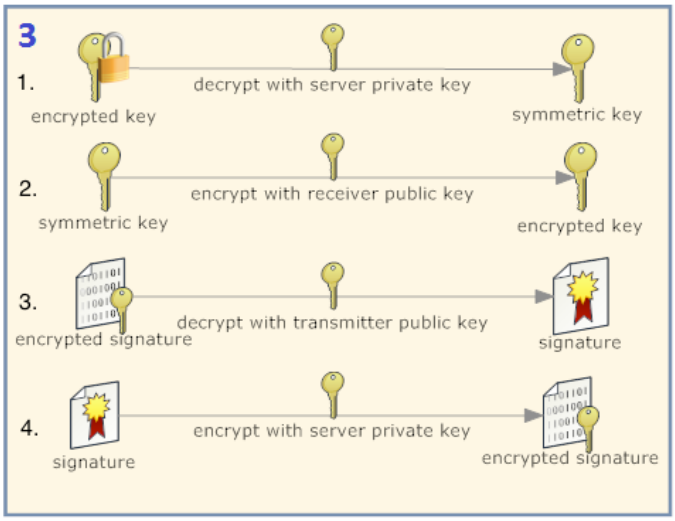


Figure 28: Send messages to the receiver

1. It decrypts the symmetric key with its private key in order to access the content of the message.
2. It encrypts the symmetric key with the receiver's public key, so only him could decrypt it.
3. It decrypts the message digest, encrypted with the sender's private key, guaranteeing this what the authorship of the message he would just have received. In the opposite case, it will reject it.
4. It encrypts the message digest with its own private key, so the future final receiver will be able to know that the server has decrypt the message, who is its original author, because it signed by himself.

4.1.3 Receives and decrypts the messages

This case scenario is dedicated to the final message reception, which supposes the end of the exchange, as the receiver would have retrieved its message. Its development is shown in the Figure 29.

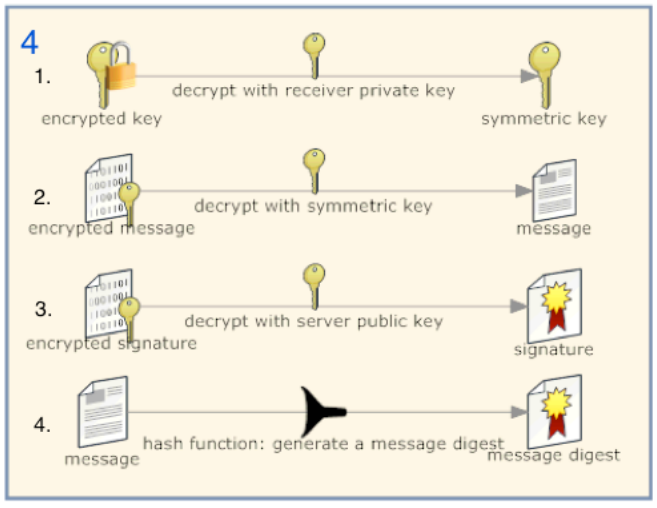


Figure 29. Receives and decrypts the messages

1. The receiver decrypts with its private key the symmetric key encrypted by the server, allowing to access the contents of the original message.
2. Decrypts the original message with the symmetric key.
3. Decrypts the message digest with the public key of the server, ensuring that it was the server that had sent the message, thus guarantees the authorship.
4. Create a digest of the original message and compares it with the summary received to see if it is equal and whether the message has not been modified during sending, thus ensuring integrity.

4.2 Algorithm selections

Before starting the implementation, it was necessary to look for the most used algorithms in every type of cryptography, regarding to the performance, the size of the key, its use and its popularity. A group of these operations was chosen to be used in each selection.

The selected algorithms, Data Encryption Standard (DES) [34], Triple DES (3DES) [35], AES [36], International Data Encryption Algorithm (IDEA) [37] y Blowfish [38], were used in the Secret Key and Hybrid Cryptography.

Regarding the asymmetric cryptography, these choices aimed to a superior keys and speed management. The selections used in Public Key Cryptography, Hybrid Cryptography and Digicial Signature were the followings: RSA [39] and ElGamal [40], because of their capacity to sign texts, among others.

For Hash functions the following algorithms were evaluated: MD5 [41], SHA-1 [42], SHA-256 [43], SHA-512, SHA3-256 [44] and SHA3-512. MD6 was discarded due to not find implementation for Java.

4.3 Time keeping

It has been developed an Android app in which we use the mentioned algorithms implementations in 4.2 to encrypt original message of the Midgar platform and it generated symmetrical keys. The principal actions taken in each algorithm were: automate key generation, encryption and decryption.

When calculating the uptake, and considering the architecture presented above, we have to determinate that the keys are only calculated one time, when the node registers itself as a new device on the IoT Midgar platform. That's the reason why the uptake of its calculation will only compute one time. On the contraty, the encryption and decryption will be used every time the message is sent. This way, the time to calculate the keys will be less relevant when calculation an algorithm uptake.

To calculate that, we use three mobile devices with Android operating systems: low-end, average and low-end. This way, we looked for the computing impact that the security could have in the different types of mobile phones in the market, regarding two aspects:

- **General calculation**: we estimate the time spent in the key calculation, encryption and decryption in the three used mobile devices. We made about 10.000 iterations for each and every calculation, so we would avoid the special cases as the garbage collector or possible undesired pauses or operating system stoppages. After that, the result was averaged to obtain the average time by iteration.

- **In case of using Midgar**: The final solution and the one we chose as the better one, it as implemented to the IoT Midgar platform to check its performance. This way, we can classify the sensors in two types:
    - **Critical node**: those actions requires send a lot of messages in short time due to the use or potential use that is given to these sensors. These are, for instance, the accelerometer, gravity sensor or the gyroscope. We could use those examples for the accident prevention. So this could be important, for example to prevent accidents, in which case we will need to keep track of the measurements of the accelerometer every few seconds, as we could see in [45–47]. So we can calculate those nodes, we have estimate an exchange of 4 messages per second.
    - **Non-critical node:** These are those actions that require a better message communication, due to the fact that a mistake on a reading could be minimal, or its impact could be no so critic. For example: the light, temperature and the pressure. We have estimated an average exchange of one message per minute.

4.4 Used software and hardware

In this section, we will treat the research done and we will present the software and the hardware which was used to do the tests.

Regarding the evaluation of the used cryptographic algorithms and, due to the possibility of obtaining different results depending on the algorithms used, we list the used libraries, in order to allow future researchers to repeat the tests:

- Library Security[3] and Crypto[4] from Java.
- Library Bouncy Castle[5].

We have used three mobile devices, whose performance has been sort in descending order: low-end, mid-end and high-end, taking in consideration its processing unit.

- **Device 1**: HTC Desire 610 running Android 4.4.2 with a quad-core processor at a speed of 1.2 GHz and 1GB of RAM.
- **Device 2**: LG Optimus L9 running Android 4.1.2 with two-core processor at a speed of 1GHz and 1GB of RAM.
- **Device 3**: LG Optimus L7 running Android 4.0 with a one core processor at speed of 1GHz and 512MB of RAM.

5 Evaluation and discussion

This section will describe the methodology used to develop this research and the tests that have been performed. After that, we will show the results and discuss them. The main objetive of this paper is to propose a possible solution to achieve the above mentioned aspects: confidentiality, authentication, integrity, non-repudiation and privacy. Therefore, we try to ensure secure communications on the Internet of Things, using as a case study the Midgar platform.

[3] Java Security: http://docs.oracle.com/javase/7/docs/api/java/security/package-summary.html
[4] Java Crypto: http://docs.oracle.com/javase/7/docs/api/javax/crypto/package-summary.html
[5] Bouncy Castle: https://www.bouncycastle.org/java.html

The technique we have chosen to achieve this security is with cryptography. However, the limited computing capacity of objects connected to the IoT, makes necessary to evaluate the temporal cost that each task would employ in different techniques and cryptographic algorithms to select the most appropriate. Due to the selected structure, we must analyse symmetric algorithms, asymmetric encryption and hash functions to select the best solution that meets the proposed objectives.

5.1 Symmetric Cryptography

In the first place, the symmetric cryptography options we have selected are presented: DES (64), 3DES(128), AES(128 y 256), Blowfish(128 y 256) and IDEA(128 y 256). Not all algorithms have the same initial conditions, due to not all of them support the same key sizes and, in the case of IDEA algorithm, it does not allow to generate a random key, so its calculation of key values are not included. For calculations in the table, the time of calculation AES key is used in IDEA.

To perform the measurements, an original Midgar platform message, that could be sent from the object to the server, has been encrypted Source Code 3.

```
<send>
        <data>
                <datum>28</datum>
                <service>18</service>
        </data>
</send>
```

Source Code 3. Midgar message example

Tabla 3 shows the results obtained in the used devices. It shows the data of the three devices and algorithms that have been used. After, for each algorithm, it shows the bit size used, the milliseconds it took to create the key and to encrypt and decrypt the text of the Source Code 3 and the total amount of the operation. In the end, it shows the calculations about how much time would be spent using this encryption method at a critical node and at non-critical node. For this calculation, the time it takes to encrypt and decrypt a critical node and one non-critical was added, not including the determination of the key because its calculation is only performed during the device registration.

Devices	Algorithms	Size (bits)	Keys (ms)	Encrypt (ms)	Decrypt (ms)	Total (ms)	1 hour communication			
							Critical (ms)	%	Non Critical (ms)	%
	DES	64	0,0355	0,1575	0,1328	0,3258	4180,32	0,1161	17,42	0,0005

Device 1	3DES	128	0,0368	0,2739	0,2649	0,5756	7758,72	0,2155	32,33	0,0009
	AES	128	0,0340	0,1247	0,1362	0,2949	3756,96	0,1044	15,65	0,0004
	AES	256	0,0401	0,1310	0,1536	0,3247	4098,24	0,1138	17,08	0,0005
	Blowfish	128	0,0344	1,1503	1,1607	2,3454	33278,40	0,9244	138,66	0,0039
	Blowfish	256	0,0391	1,1542	1,1435	2,3368	33086,88	0,9191	137,86	0,0038
	IDEA	128	0,0340	0,1941	0,1358	0,3639	4750,56	0,1320	19,79	0,0005
	IDEA	256	0,0401	0,1965	0,1281	0,3647	4674,24	0,1298	19,48	0,0005
Device 2	DES	64	0,0375	0,1945	0,3865	0,6185	8366,40	0,2324	34,86	0,0010
	3DES	128	0,0314	0,3149	0,5028	0,8491	11774,88	0,3271	49,06	0,0014
	AES	128	0,0237	0,1342	0,3494	0,5073	6963,84	0,1934	29,02	0,0008
	AES	256	0,0424	0,1597	0,3809	0,5830	7784,64	0,2162	32,44	0,0009
	Blowfish	128	0,0218	1,2402	1,4471	2,7091	38697,12	1,0749	161,24	0,0045
	Blowfish	256	0,0302	1,2276	1,4410	2,6988	38427,84	1,0674	160,12	0,0044
	IDEA	128	0,0237	0,1617	0,3519	0,5373	7395,84	0,2054	30,82	0,0009
	IDEA	256	0,0424	0,1556	0,3383	0,5363	7112,16	0,1976	29,63	0,0008
Device 3	DES	64	0,0332	0,2469	0,4770	0,7571	10424,16	0,2896	43,43	0,0012
	3DES	128	0,0435	0,3974	0,6522	1,0931	15114,24	0,4198	62,98	0,0017
	AES	128	0,0288	0,1960	0,4777	0,7025	9701,28	0,2695	40,42	0,0011
	AES	256	0,0343	0,2200	0,5063	0,7606	10458,72	0,2905	43,58	0,0012
	Blowfish	128	0,0289	1,7291	2,0003	3,7583	53703,36	1,4918	223,76	0,0062
	Blowfish	256	0,0364	1,7231	2,0301	3,7896	54046,08	1,5013	225,19	0,0063
	IDEA	128	0,0288	0,5054	0,4710	1,0052	14060,16	0,3906	58,58	0,0016
	IDEA	256	0,0343	0,5161	0,4631	1,0135	14100,48	0,3917	58,75	0,0016

Table 8. Symmetric algorithms table result

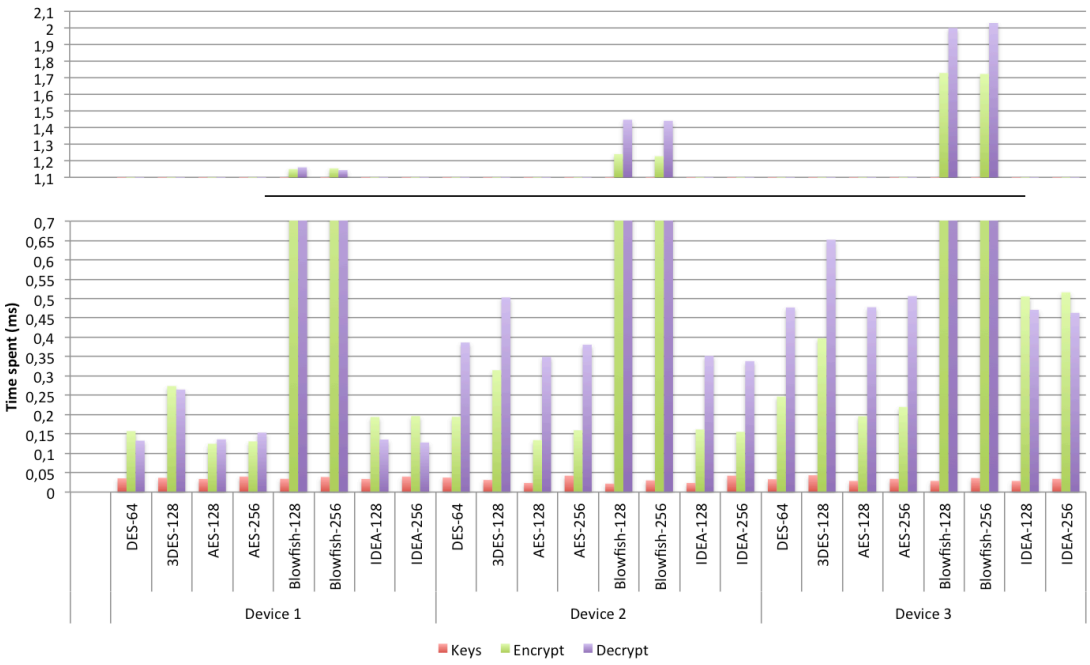


Figure 30. Symmetric algorithms results

Analyzing Tabla 3 and Ilustración 10, we can suggest the following interpretations:

- The key calculation becomes practically irrelevant because the differences are small and in no case exceed 0,05 ms.
- The algorithms Blowfish 128bits and Blowfish 256bits, require much time to encrypt and decrypt, resulting much more expensive than other algorithms evaluated.
- The DES algorithm is the most insecure because they use smaller key and it facilitates crack passwords by brute force, contrary to what would happen with larger and more robust passwords [48]. Despite this, it presents a longer time calculation versus safer algorithms as the AES 128bits and AES 256bits.
- AES 256bits takes a little longer than AES 128bits but provides greater security when using the double bits to the key.
- 3DES 128bits takes double time than as AES 128bits, but only allows to use 128bits keys.
- IDEA 256bits takes a little bit more time than IDEA 128bits but offers much greater security using the double bits to the key.
- IDEA 256bits requires a little more time than AES 256bits.
- Finally, the decission would be between the algorithms AES and IDEA, where the AES times are lower, especially on encrypt, for both keys, 128bits and 256bits. In relation to the time that the algorithm takes to encrypt and decrypt in one hour of communication, the AES percentage is lower in every case. AES 128bits is the one that takes less time, followed by AES 256bits.
- The time that AES 128bits spend, according to the average estimate between the three devices, is 0,19% for critical nodes and 0,0008% for non-critical nodes. Besides, AES 256bits spend 0.21% for critical nodes and 0,0009% for non-critical nodes.

We assume a high security in 256bits version, with an extra 0,02% consumption for critical nodes and 0,0001% for non-critical nodes, so we will choose the AES 256bits version for our solution due to it's the faster and more secure.

5.2 Asymmetric Cryptography

In this section, we will evaluate the send of encrypted messages by using a public key algorithm. This will allow us to create a pair of keys for each node, and change information in a safe way, making sure that only the receptor will be allowed to decrypt the message, i.e. confidentiality. The algorithms selected were RSA and ElGamal.

For measurements, a symmetric password generated in our application has been encrypted Source Code 4 and has an average size of 128bytes.

```
W1PnUCA+fVEAOaKaxfu1cQ==
```

Source Code 4. Key

Tabla 4 shows the obtained results for the three devices and the algorithms that we used. For each algorithm, it shows the bits size used, the millisecond that took to creat the key and encrypt and decrypt the text Source Code 3 and the total sum of the operation. In the end, it shows the calculation about how many time would it take to apply this encrypt method on a critical node and in a non-critical one. For this calculation, the time spend on encrypt a critical

and non-critical node was added to the one spend on decrypt. This result not include the key calculation, due to that calculation is only done during the device registration.

Devices	Algorithms	Size (bits)	Keys (ms)	Encrypt (ms)	Decrypt (ms)	Total (ms)	1 hour communication			
							Critical (ms)	%	Non Critical (ms)	%
Device 1	RSA	1024	806,7	0,54	7,58	814,82	116928	3,248	487,2	0,014
	RSA	2048	2671,7	1,41	39,32	2712,43	586512	16,292	2443,8	0,068
	RSA	4096	24460,3	4,49	248,53	24713,32	3643488	101,208	15181,2	0,422
	ElGamal	1024	16,8	31,36	14,47	62,63	659952	18,332	2749,8	0,076
	ElGamal	2048	118,2	229,98	117,15	465,33	4998672	138,852	20827,8	0,579
	ElGamal	4096	880,5	1783,21	892,33	3556,04	38527776	1070,216	160532,4	4,459
Device 2	RSA	1024	869	0,746	8,99	878,736	140198,4	3,894	584,16	0,016
	RSA	2048	2821,7	1,807	40,62	2864,127	610948,8	16,971	2545,62	0,071
	RSA	4096	27357,3	5,46	256,44	27619,2	3771360	104,760	15714	0,437
	ElGamal	1024	17,7	32,54	15,86	66,1	696960	19,360	2904	0,081
	ElGamal	2048	123,2	240,36	120,32	483,88	5193792	144,272	21640,8	0,601
	ElGamal	4096	933,9	1864,92	933,53	3732,35	40297680	1119,380	167907	4,664
Device 3	RSA	1024	886,8	1,5	9,898	898,198	164131,2	4,559	683,88	0,019
	RSA	2048	6501	2,7	50,769	6554,469	769953,6	21,388	3208,14	0,089
	RSA	4096	42143,3	6,02	333,08	42482,4	4883040	135,640	20346	0,565
	ElGamal	1024	25,6	40,751	20,349	86,7	879840	24,440	3666	0,102
	ElGamal	2048	153,7	305,684	153,601	612,985	6613704	183,714	27557,1	0,765
	ElGamal	4096	1185,6	2364,47	1186,48	4736,55	51133680	1420,380	213057	5,918

Table 9. Asymmetric algorithms table result

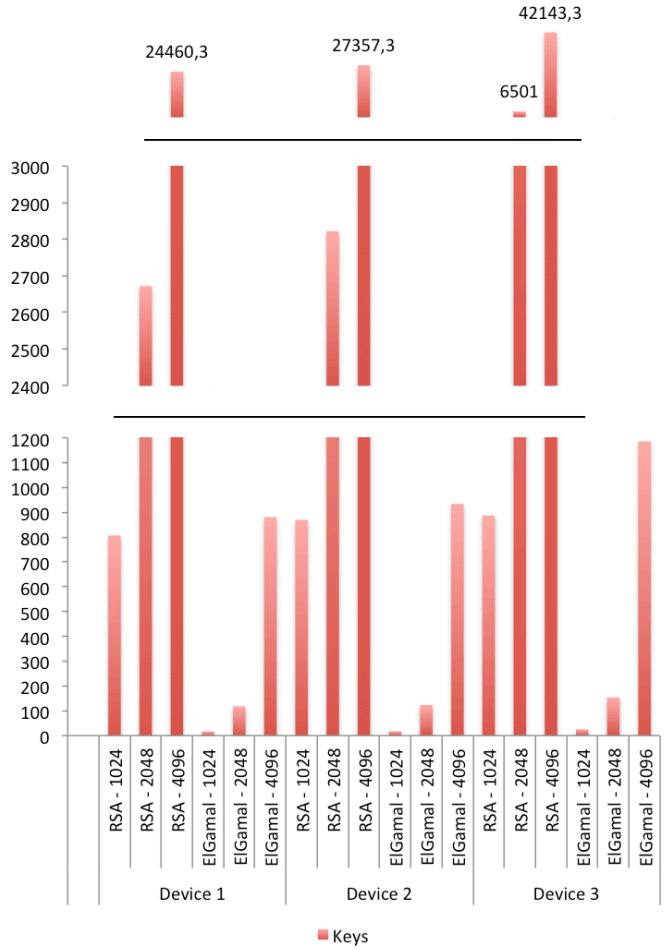


Figure 31. Key creation on asymmetric algorithms

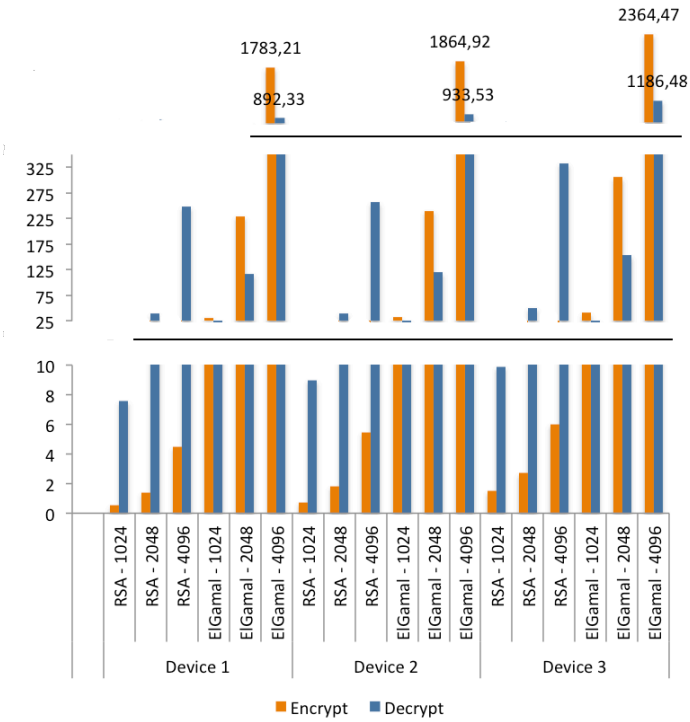


Figure 32. Encrypt and decrypt on asymmetric algorithms

Analyzing Tabla 4 and Ilustración 11 e Ilustración 12, we can suggest the following interpretations:

- RSA algorithm's key pair generation time are much higher than ElGamal, and increases as the key size increases. RSA employs an average of 31,3 seconds in the most expensive case, 4096bits, versus ElGamal with 1 second average for the same key size.
- As we can be seen in Figure 3, the RSA times in encryption and decryption are lower in all cases, so the cost in one hour communication is evaluated to see finally that algorithm has a higher consumption.
- The average percentage consumed in one hour communication for critical nodes for 1024, 2048 and 4096bits RSA is 3,9%, 18,2% and 113,9% respectively. Whereas, ElGamal, employs the 20,71%, 155,6% and 1203,3% for a size of 1024, 2048 and 4096bits respectively.
- In a one hour communication for non-critical nodes RSA employs the 0,02%, 0,08% and 0,5% for key sizes of 1024, 2048 and 4096bits. For its part, ElGamal employs 0,09%, 0,65% and 5% for 1024, 2048 and 4096 key bits.
- From this, it follows that ElGamal not may be used with keys of 2048 and 4096bits size, because for critical communications between nodes exceeds the computation time encrypting and decrypting in an hour, so it would not be viable. The same goes for the version of RSA 4096bits.
- Moreover, RSA showed a much higher initial cost for calculating key. As this cost is computed only once, when registration of the device on the platform, is less relevant to continuing communications, where presents much smaller values than ElGamal.

Our recommendation, depending on the values studied, is the utilization of 2048bits RSA algorithm at non-critical nodes, where communication is lower, and employs only 0,08% of time in encrypting and decrypting, offering good performance. For critical nodes, where communication is very continuous, with sending up to four messages per minute, the best option, in our opinion, would by using the RSA algorithm with a key length of 1024bits, that

although it offers less security for being less resistant to be discovered by brute force, offers a much more agile performance for this type of sending, spending 3,9% in the processing of messages.

5.3 Hash functions

Now we show the tests results of different hash functions. The aim is to ensure the message integrity. So we have decided to create a summary thereof by which the receiving user can verify that the message received matches the original that had been sent, with a summary of the original message and checking if it matches the had you received. This will allow you to discern if has been manipulated in sending.To do this, we must select among some existing hash algorithms, the one that best suits our needs in terms of consumption. We have tested it with the algorithms: MD5, SHA-1, SHA-2 (256 and 512bits) and SHA-3 (256 and 512bits),

Devices	Algorithms	Hash	1 hour communication			
			Critical	%	Non Critical	%
Device 1	MD5	0,2129	3065,76	0,08516	12,774	0,0004
	SHA-1	0,2551	3673,44	0,10204	15,306	0,0004
	SHA2-256	0,3976	5725,44	0,15904	23,856	0,0007
	SHA2-512	0,7836	11283,84	0,31344	47,016	0,0013
	SHA3-256	0,3437	4949,28	0,13748	20,622	0,0006
	SHA3-512	0,6197	8923,68	0,24788	37,182	0,0010
Device 2	MD5	0,2686	3867,84	0,10744	16,116	0,0004
	SHA-1	0,3023	4353,12	0,12092	18,138	0,0005
	SHA2-256	0,485	6984	0,194	29,1	0,0008
	SHA2-512	0,9227	13286,88	0,36908	55,362	0,0015
	SHA3-256	0,4374	6298,56	0,17496	26,244	0,0007
	SHA3-512	0,7676	11053,44	0,30704	46,056	0,0013
Device 3	MD5	0,3848	5541,12	0,15392	23,088	0,0006
	SHA-1	0,4605	6631,2	0,1842	27,63	0,0008
	SHA2-256	0,6977	10046,88	0,27908	41,862	0,0012
	SHA2-512	1,3104	18869,76	0,52416	78,624	0,0022
	SHA3-256	0,6737	9701,28	0,26948	40,422	0,0011
	SHA3-512	1,0228	14728,32	0,40912	61,368	0,0017

Table 10. Hash algorithms result

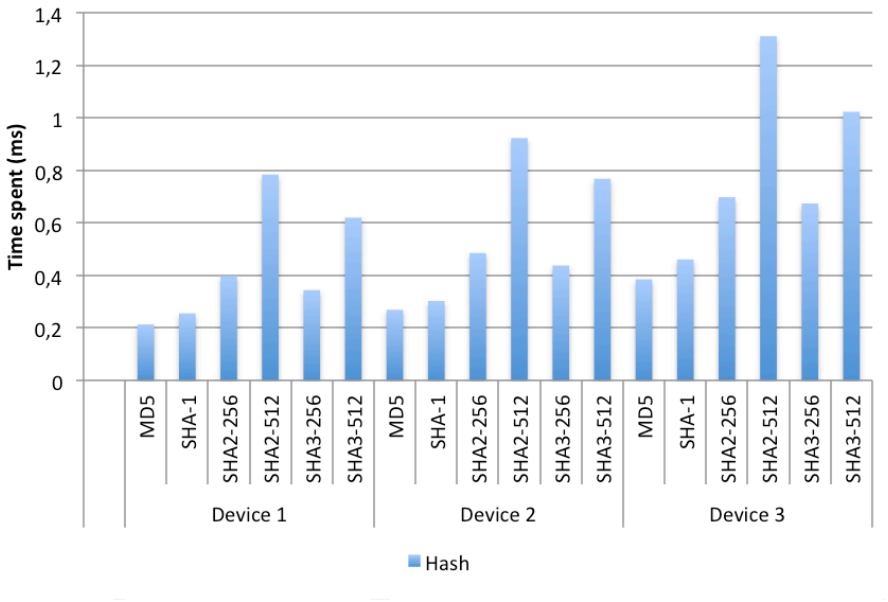


Figure 33. Hash algorithms result

Analyzing Tabla 5 and Ilustración 13, we can suggest the following interpretations:

- The MD5 algorithm with a size of 128bits is faster, overall, than the rest.
- SHA-1 with a size of 160bits is only slightly slower than MD5.
- SHA-2 for 256 and 512bits, presents a higher comsumption times than SHA-3 for the same key sizes.
- SHA-1 employs a 0,1%, in a one hour communication, for critical nodes and a 0,0006% for non-criticial nodes.
- SHA-2 employs a 0,2% and a 0,4% for size of 256 and 512bits in critical nodes, and a 0,0009% and a 0,002% for a size of 256 and 512bits at non-critical nodes.
- SHA-3 employs a 0,2% and a 0,3% for a size of 256 y 512bits in critical nodes, and a 0,0008% and a 0,001% for a size of 256 and 512bits in non-critical nodes.

Some security issues present in these hash functions:

- MD5 is the fastest option of evaluated, but presents problems due to hash collision [49], so not the safest option.
- In 2005 security breaches were detected in SHA-1, so its use is not recommended due to the collision [42]. Therefore, it emerged SHA-2.
- In 2011, an attack broke SHA-2 security [50], 256 and 512bits.

Therefore, the final decision is to use SHA-3, that presents a superior performance and lower cost than SHA-2. In this case we can use SHA-3 with 512bits because it offers greater security due to size, and the consumption, 0,3% for communication among critical nodes and 0,001% for communication among non-critical nodes, is low.

6 Conclusion

In this article we present a proposal of security for the Internet of Things, implemented in Midgar IoT platform, in order to seek greater security in sending messages between objects on the network and trying to lose as little time as possible in computing time.

The solution is to adopt measures characteristics of the hybrid cryptography combining it with the digital signature to obtain the benefits of each. With this we obtain the confidentiality and privacy needed to exchange messages private and add integrity and authentication, concepts needed in a network IoT.

The study evaluated several algorithms among selected cryptography types, in order to find the quickest possible solution that consumes less time and storage in a changing environment with objects with different features, but always maintaining the necessary security. This study shows that the best choice is to use **AES**, **RSA** and **SHA3**.

On one hand, the **AES** symmetric cryptography algorithm for a size of **256bits** obtained the best results in time of encryption/decryption, where employs the **0,21%** of one hour communications in critical nodes and **0,0009%** in non-critical nodes communication, and also in terms of security by key size, where it takes to generate a 256bits key an average of **0,04 ms**.

Respect to asymmetric cryptography, RSA is faster than ElGamal in a prolonged communication, because ElGamal get to use more than twice as long time than RSA despite an initial keys calculation time much higher. RSA requires an average of **0,9 ms** and **4 s** for calculating keys to a size of **1024** and **2048bits** respectively. For a one hour communication among critical nodes, where the **1024bits** version would be used, RSA consume **3,9%** of time on tasks of encryption/decryption. In a one hour communication among non-critical nodes, RSA, where the **2048bits** version would be used, consume only **0,08%** of time. Finally, to summarize the message and guarantee its integrity, **SHA3** Hash function would be used for a size of **512bits**, where the function consume **0,3%** of time, in a one hour communications, between critical nodes and **0,001%** non-critical nodes.

This security, that uses jointly **RSA**, **AES** and **SHA3** algorithms, would have an ultimate impact of **4,4%** in consumption in a one hour communication at critical nodes, and **0,08%** in non-critical nodes.

However, must be emphasized that in the case of RSA, this is high time for the automatic generation of the key pair, but because this calculation is done only once, it can be assumed by the devices. While, in the case of encryption and decryption, to be performed in each message exchange, these times are lower for 1024bits, a 16,8% less than ElGamal in critical nodes and 0,07% in non-critical nodes. In 2048bits, the ElGamal require a 137,4% increase in consumption in critical nodes, and 0,57% in non-critical nodes. Therefore, the choice of RSA versus ElGamal.

However, as shown in studies, you can consume quite some time depending on the algorithm used and, e ven selecting which consume less time maintaining security, a 4,4% at critical nodes is a long time, especially in low-end devices.

However, this is something necessary to maintain the security and privacy of the objects when they require it because even though some personal information may seem irrelevant, other information such as the location or bank accounts, could become in a great danger in case of being supplanted.

7 Future Work

In this study only we try sending data secure in an insecure network among Smart Objects. This results in remaining open many avenues for further investigation and/or improving this work. Some future research could be:

- **Improve in the computing or in the algorithms processing**: thus allow maintaining the same or similar architecture but with a considerably better results in time consuming depending on the viability and the contrasts of improvements.
- **Secure registration:** secure registry for objects that want to register in a IoT platform for maintaining the privacy of this record.
- **Optimizing data sending in the platform**: now sends large amounts of information, that could be reduced. This will reduce the time of encryption and decryption of messages and therefore will expedite the communications.
- **Replay attacks**: another problem in IoT are the replay attacks. For that, it is necessary create a quickly and low cost solution.

8 References

[1] J. Bohn, V. Coroamă, M. Langheinrich, F. Mattern, M. Rohs, Living in a World of Smart Everyday Objects—Social, Economic, and Ethical Implications, Hum. Ecol. Risk Assess. An Int. J. 10 (2004) 763–785. doi:10.1080/10807030490513793.

[2] L. Atzori, A. Iera, G. Morabito, The Internet of Things: A survey, Comput. Networks. 54 (2010) 2787–2805. doi:10.1016/j.comnet.2010.05.010.

[3] Fundación Telefónica, La Sociedad de la Información en España 2014, 2015.

[4] P.N. Mahalle, N.R. Prasad, R. Prasad, Novel Threshold Cryptography-based Group Authentication (TCGA) Scheme for the Internet of Things (IoT), (2015).

[5] M. Chui, M. Löffler, R. Roberts, The internet of things, McKinsey Q. 2 (2010) 1–9.

[6] M. Swan, Sensor Mania! The Internet of Things, Wearable Computing, Objective Metrics, and the Quantified Self 2.0, J. Sens. Actuator Networks. 1 (2012) 217–253. doi:10.3390/jsan1030217.

[7] C. González García, B.C. Pelayo G-Bustelo, J. Pascual Espada, G. Cueva-Fernandez, Midgar: Generation of heterogeneous objects interconnecting applications. A Domain Specific Language proposal for Internet of Things scenarios, Comput. Networks. 64 (2014) 143–158. doi:10.1016/j.comnet.2014.02.010.

[8] Harmony Institute, Purposeful storytelling and designing, 2013.

[9] C. Doukas, I. Maglogiannis, V. Koufi, F. Malamateniou, G. Vassilacopoulos, Enabling data protection through PKI encryption in IoT m-Health devices, IEEE 12th Int. Conf. Bioinforma. Bioeng. BIBE 2012. (2012) 25–29. doi:10.1109/BIBE.2012.6399701.

[10] W.M. Ibrahim, A.-E.M. Taha, H.S. Hassanein, Using smart vehicles for localizing isolated Things, Comput. Commun. (2014). doi:10.1016/j.comcom.2014.05.013.

[11] H. Suo, J. Wan, C. Zou, J. Liu, Security in the internet of things: A review, Proc. - 2012 Int. Conf. Comput. Sci. Electron. Eng. ICCSEE 2012. 3 (2012) 648–651. doi:10.1109/ICCSEE.2012.373.

[12] T. Polk, S. Turner, Security challenges for the internet of things, … Interconnecting Smart Objects with Internet, …. (2011). https://www.iab.org/wp-content/IAB-uploads/2011/03/Turner.pdf.

[13] E. Borgia, The Internet of Things vision : Key features , applications and open issues, Comput. Commun. 54 (2014) 1–31. doi:10.1016/j.comcom.2014.09.008.

[14] N.I.C. NIC, Disruptive Civil Technologies: Six Technologies with Potential Impacts on US Interests out to 2025, (2008).

[15] D. Miorandi, S. Sicari, F. De Pellegrini, I. Chlamtac, Internet of things: Vision, applications and research challenges, Ad Hoc Networks. 10 (2012) 1497–1516.

[16] G. Kortuem, F. Kawsar, D. Fitton, V. Sundramoorthy, Smart objects as building blocks for the internet of things, Internet Comput. IEEE. 14 (2010) 44–51.

[17] E. Khorov, A. Lyakhov, A. Krotov, A. Guschin, A survey on IEEE 802.11ah: An enabling networking technology for smart cities, Comput. Commun. 58 (2015) 53–69. doi:10.1016/j.comcom.2014.08.008.

[18] D. Evans, The internet of things: How the next evolution of the internet is changing everything, CISCO White Pap. 1 (2011) 14.

[19] L. Eschenauer, V.D. Gligor, A key-management scheme for distributed sensor networks, 9th ACM Conf. Comput. Commun. Secur. (2002) 41–47.

[20] G.G. Meyer, K. Främling, J. Holmström, Intelligent products: A survey, Comput. Ind. 60 (2009) 137–148.

[21] C.P. Mayer, Security and privacy challenges in the internet of things, Electron. Commun. EASST. 17 (2009).

[22] L. Da Vinci, Leonardo's Notebooks: Writing and Art of the Great Master, Black Dog & Leventhal, 2013.

[23] R. Acharya, K. Asha, Data integrity and intrusion detection in wireless sensor networks, in: Networks, 2008. ICON 2008. 16th IEEE Int. Conf., IEEE, 2008: pp. 1–5.

[24] R. Roman, P. Najera, J. Lopez, Securing the Internet of things, Computer (Long. Beach. Calif). 44 (2011) 51–58. doi:10.1109/MC.2011.291.

[25] N. Ye, Y. Zhu, R.C. Wang, R. Malekian, Q.M. Lin, An efficient authentication and access control scheme for perception layer of internet of things, Appl. Math. Inf. Sci. 8 (2014) 1617–1624. doi:10.12785/amis/080416.

[26] G. Zhao, X. Si, J. Wang, X. Long, T. Hu, A novel mutual authentication scheme for Internet of Things, Proc. 2011 Int. Conf. Model. Identif. Control. (2011) 563–566. doi:10.1109/ICMIC.2011.5973767.

[27] Martin Feldhofer, S. Dominikus, J. Wolkerstorfer, Strong Authentication for RFID Systems Using the AES Algorithm, Cryptogr. Hardw. Embed. Syst. - CHES 2004. 3156 (2004) 357–370. doi:10.1007/b99451.

[28] B. Calmels, S. Canard, M. Girault, H. Sibert, Low-cost cryptography for privacy in RFID systems, Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 3928 LNCS (2006) 237–251. doi:10.1007/11733447_17.

[29] O. Savry, F. Pebay-Peyroula, F. Dehmas, G. Robert, J. Reverdy, RFID Noisy Reader How to Prevent from Eavesdropping on the Communication?, Springer, 2007.

[30] Z. Hu, The research of several key question of internet of things, in: Intell. Sci. Inf. Eng. (ISIE), 2011 Int. Conf., IEEE, 2011: pp. 362–365.

[31] G. V Lioudakis, E. Koutsoloukas, N. Dellas, S. Kapellaki, G.N. Prezerakos, D. Kaklamani, et al., A proxy for privacy: the discreet box, in: EUROCON, 2007. Int. Conf. &# 34; Comput. as a Tool&# 34;, IEEE, 2007: pp. 966–973.

[32] E. Mykletun, J. Girao, D. Westhoff, Public key based cryptoschemes for data concealment in wireless sensor networks, IEEE Int. Conf. Commun. 5 (2006) 2288–2295. doi:10.1109/ICC.2006.255111.

[33] O. Bergmann, S. Gerdes, C. Bormann, Simple Keys for Simple Smart Objects, Smart Object Secur. …. (2012). http://www.tschofenig.priv.at/sos-papers/OlafBergmann.pdf\nhttp://tools.ietf.org/html/draft-gilger-smart-object-security-workshop-02.

[34] M.E. Smid, D.K. Branstad, Data Encryption Standard: past and future, Proc. IEEE. 76 (1988) 550–559. doi:10.1109/5.4441.

[35] H. Kummert, The PPP Triple-DES Encryption Protocol (3DESE), (1998).

[36] D. Selent, Advanced encryption standard, 6 (2010) 1–14.

[37] M.-P. Leong, O.Y.H. Cheung, K.H. Tsoi, P.H.W. Leong, A bit-serial implementation of the international data encryption algorithm IDEA, in: Field-Programmable Cust. Comput. Mach. 2000 IEEE Symp., IEEE, 2000: pp. 122–131.

[38] A. Mousa, Data encryption performance based on Blowfish, in: ELMAR, 2005. 47th Int. Symp., IEEE, 2005: pp. 131–134.

[39] D.R. Smith, J.T. Palmer, Universal fixed messages and the Rivest-Shamir-Adleman cryptosystem, Mathematika. 26 (1979) 44–52.

[40] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, in: Adv. Cryptol., Springer, 1985: pp. 10–18.

[41] R. Rivest, The MD5 message-digest algorithm, (1992).

[42] B. Schneier, Schneier on Security: Cryptanalysis of SHA-1, Schneier. Com. (2005).

[43] N. Sklavos, O. Koufopavlou, On the hardware implementations of the SHA-2 (256, 384, 512) hash functions, in: Circuits Syst. 2003. ISCAS'03. Proc. 2003 Int. Symp., IEEE, 2003: pp. V–153.

[44] J.-P. Aumasson, L. Henzen, W. Meier, R.C.-W. Phan, Sha-3 proposal blake, Submiss. to NIST. (2008).

[45] F. Foerster, M. Smeja, J. Fahrenberg, Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring, Comput. Human Behav. 15 (1999) 571–583.

[46] C.J. James, S. Kumar, Detection of posture and motion by accelerometer sensors, in: Electron. Comput. Technol. (ICECT), 2011 3rd Int. Conf., IEEE, 2011: pp. 369–371.

[47] J. Mäntyjärvi, J. Himberg, T. Seppänen, Recognizing human motion with multiple acceleration sensors, in: Syst. Man, Cybern. 2001 IEEE Int. Conf., IEEE, 2001: pp. 747–752.

[48] M. Agrawal, P. Mishra, A comparative survey on symmetric key encryption techniques, Intern. J. Comput. Sci. Eng. 4 (2012) 877–882. http://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=82397469&site=ehost-live.

[49] M. Stevens, Fast Collision Attack on MD5., IACR Cryptol. ePrint Arch. 2006 (2006) 104.

[50] D. Khovratovich, C. Rechberger, A. Savelieva, Bicliques for preimages: attacks on Skein-512 and the SHA-2 family, in: Fast Softw. Encryption, Springer, 2012: pp. 244–263.