Universitat Oberta de Catalunya

uoc.edu

Seguridad en la Internet of Things: Ataques y medidas de seguridad en modelos Cloud de IoT

Nicolás Moral de Aguilar
Master universitario de Ciberseguridad y Privacidad
Seguridad en la IoT

Jorge Miguel Moneo
Jorge Miguel Moneo

1/06/2021

Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada 3.0 España de Creative Commons

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Seguridad en la Internet of Things: Ataques y medidas de seguridad en modelos Cloud de IoT*
Nombre del autor:	*Nicolás Moral de Aguilar*
Nombre del consultor/a:	*Jorge Miguel Moneo*
Nombre del PRA:	*Jorge Miguel Moneo*
Fecha de entrega (mm/aaaa):	06/2021
Titulación:	*Máster universitario en Ciberseguridad y Privacidad*
Área del Trabajo Final:	*Seguridad en la IoT*
Idioma del trabajo:	*Español*
Palabras clave	*IoT, Cloud, Seguridad*

Resumen del Trabajo (máximo 250 palabras): *Con la finalidad, contexto de aplicación, metodología, resultados y conclusiones del trabajo.*

En este trabajo se ha realizado un estudio sobre los ataques más comunes y las medidas de seguridad pertinentes relativas a un sistema basado en IoT. El trabajo se divide en dos partes.

En la primera parte del mismo se han abarcado los ataques más relevantes históricamente a sistemas basados en redes y dispositivos IoT, las vulnerabilidades más comunes y la forma de mitigarlas, y se realizaron también pruebas relativas a ataques de penetración en un entorno simulado.

La segunda parte se centra en medidas de seguridad, donde se estudian sistemas de identidad y autenticación segura enfatizando en la viabilidad de los mismos en sistemas de IoT, protocolos de cifrado seguro, métodos para el filtrado de amenazas, y por último se estudia las opciones de seguridad en los protocolos de IoT más utilizados. En esta segunda parte, se realiza también una prueba de concepto sobre el protocolo MQTT, implementando cifrado mediante TLS de las conexiones utilizando un servidor en la nube de Microsoft, Azure.

Abstract (in English, 250 words or less):

In this paper, a research on the most common attacks and relevant security measures related to an IoT-based system has been carried out. The paper is divided into two parts.

The first part covers the most relevant attacks on history over systems based on IoT networks and devices, the most common vulnerabilities and how to mitigate them, and a simulated environment was used to perform some penetration tests on a IoT-based system.

The second part focuses on security measures, in this part a study about identity and secure authentication has been performed emphasizing its viability in IoT systems. It also talks about secure encryption protocols, thread filtering methods, and finally, security options in the most used IoT protocols. In this second part, a proof of concept on the MQTT protocol is also performed, setting up TLS encryption of connections using a server in Azure, the Microsoft's cloud.

Índice

Lista de figuras

Introducción

Contexto y justificación del Trabajo

Para poder enfocar bien el planteamiento del problema, es necesario conocer en principio, el origen del mismo. Se expondrán los motivos por los que considero que este TFM tiene sentido.

- La rápida expansión de los dispositivos y redes IoT en gran parte del mundo, dando lugar a desarrollos muy rápidos y a multitud de empresas emergentes con poca experiencia en el sector. Además, han de tenerse en cuenta la cantidad de empresas de otros países que exportan tanto hardware como software a Europa, y los pobres o a veces incluso inexistentes controles de seguridad por los que han de pasar estos dispositivos.
- La limitación de costes de fabricación y desarrollo, es directamente proporcional tanto a la complejidad como a la seguridad de dichos sistemas. La mayoría basados en microcontroladores provenientes del mundo “Arduino” y derivados, están muy limitados a nivel de hardware, lo que obviamente, fuerza a las empresas en muchos desarrollos no siendo posible la implementación de cifrados complejos ni muchas de las implementaciones de seguridad habituales en sistemas de comunicación en la nube.
- La inexistencia de un estándar de obligado cumplimiento hace que las diferentes empresas utilicen protocolos e implementaciones privadas, la mayoría de las veces de código cerrado.
- Las limitaciones intrínsecas al protocolo IPv4, en el que existe un número reducido de direcciones IP usables. Siendo aún necesaria a día de hoy la intervención de hardware adicional como Routers o Hubs, dada la imposibilidad de conectar cada dispositivo directamente al sistema.

Por estos motivos aquí indicados, creo que es necesaria la mejora continua en seguridad dentro de entornos IoT, incluso más, si estos están en un servicio alojado en Cloud. La mayor parte de la información relativa al comportamiento de estos dispositivos puede permitirnos generar perfiles de los usuarios que hacen uso de ellos, obteniendo mucha información personal de forma directa, desde los sensores de los que dispongan estos dispositivos, o indirecta, a través del uso de los mismos mediante estudios de Big Data.

Esto, y la posible intervención de estos dispositivos en entornos de red local previos a la salida a internet, posibilita en mucha parte posibles ataques MitM y por tanto fugas de datos no solo relativas a la información que manejan estos sistemas sino también de redes privadas.

Objetivos del Trabajo

Dado que la temática del proyecto es un área bastante amplia es conveniente establecer unos límites de lo que se va a desarrollar en el mismo. Sin embargo, se establecerán dos partes bien diferenciadas. Para concretar, y siendo coherentes con el título del propio TFM, la primera parte estará enfocada en ataques a estos sistemas, y la segunda a medidas de seguridad. Dichos objetivos se listan a continuación.

Vectores de ataque

- Exposición de las vulnerabilidades con mayor impacto en entornos IoT.
- Exposición de ataques relevantes que han sufrido este tipo de sistemas.
- Pruebas de concepto en entornos de ataque simulados.

Medidas de seguridad

- Análisis sobre la parte de vectores de ataque, priorizando la resolución de las vulnerabilidades más usuales, con un enfoque adicional en las medidas de seguridad disponibles en cloud.
- Análisis sobre la gestión de identidad (IAM) y autenticación segura (MFA).
- Análisis de protocolos de cifrado seguro (TLS).
- Análisis de sistemas de filtrado de amenazas (Reverse Proxy, WAF y CASB).
- Análisis de seguridad en protocolos estándar.

Enfoque y método seguido

Dada la diferente naturaleza de los objetivos del trabajo es conveniente establecer una metodología por objetivos. Muchos de ellos son de enfoque mayoritariamente teórico, pero al existir varios de enfoque práctico, es necesario establecer una propia. Esta metodología lleva asociada la recolección de información y redacción de la memoria a lo largo del desarrollo de cada uno de los objetivos que se han establecido en este trabajo.

- Exposición de las vulnerabilidades más usuales en entornos IoT.

En esta primera etapa, se recogerán las vulnerabilidades que se producen de manera más usual dentro de los entornos IoT, y se plantearán posibles formas de comprometer o atacar a cada una de ellas.

- Exposición de ataques relevantes que han sufrido este tipo de sistemas.

Se detallarán las características de ataques importantes que han sufrido dispositivos IoT gestionados desde cloud, incluyendo la relevancia y los vectores de ataque de los mismos.

- Pruebas de concepto en entornos de ataque simulados.

En esta etapa, se preparará un entorno vulnerable simulado para comprobar la viabilidad de realizar ataques sobre sistemas IoT.

- Análisis sobre la parte de ataque, priorizando la resolución de las vulnerabilidades más usuales, con un enfoque adicional en las medidas de seguridad disponibles en cloud.

Como posibles mitigaciones de las vulnerabilidades más usuales que habremos detallado en la primera etapa, consistirá en su mayor parte en la asociación tanto a la metodología y clasificación de OWASP como en la parte de relacionada con hardware a la matriz de ataque (MITRE ATT&CK).

- Análisis sobre la gestión de identidad (IAM) y autenticación segura (MFA).

Se hará un análisis sobre los sistemas más usuales de gestión de identidad en entornos cloud, y los distintos métodos de autenticación multifactor disponibles, incidiendo en la viabilidad de la implementación en sistemas IoT.

- Análisis de protocolos de cifrado seguro (TLS).

Se desarrollará una comparativa de los diferentes protocolos de cifrado seguro, viendo las diferentes características de los mismos. Cuales son seguros a medio largo plazo, la viabilidad de la implementación en el hardware IoT existente. También se verá su implementación en protocolos estándar de domótica que puedan utilizarse en entornos cloud.

- Análisis de sistemas de filtrado de amenazas (Reverse Proxy, WAF y CASB).

Se estudiarán sistemas de filtrado de amenazas, explicando el funcionamiento de los tres grandes tipos de sistemas en entornos reales. Proxies inversos, Web Application Firewalls y Cloud Access Security Brokers. Se incluirán las directivas y reglas usuales que han de incluirse en ellos para bastionar el acceso desde y hacia dispositivos IoT locales.

- Análisis de seguridad en protocolos estándar.

Se realizará un estudio de la seguridad actualmente disponible en protocolos estándar de dispositivos IoT (MQTT, Zigbee…) y un análisis sobre si es o no suficiente.

Planificación

Cada uno de los objetivos anteriormente planteados se compone de unas tareas asociadas. Para mejorar la gestión de tiempo van a enumerarse estas tareas incluyendo también el tiempo asociado a la redacción y preparación del

TFM. Se tendrá en cuenta el plan de trabajo establecido por la UOC, adaptando estas tareas a la gestión de entregas del TFM.

Primera Parte (Vectores de Ataque) – Entrega 2

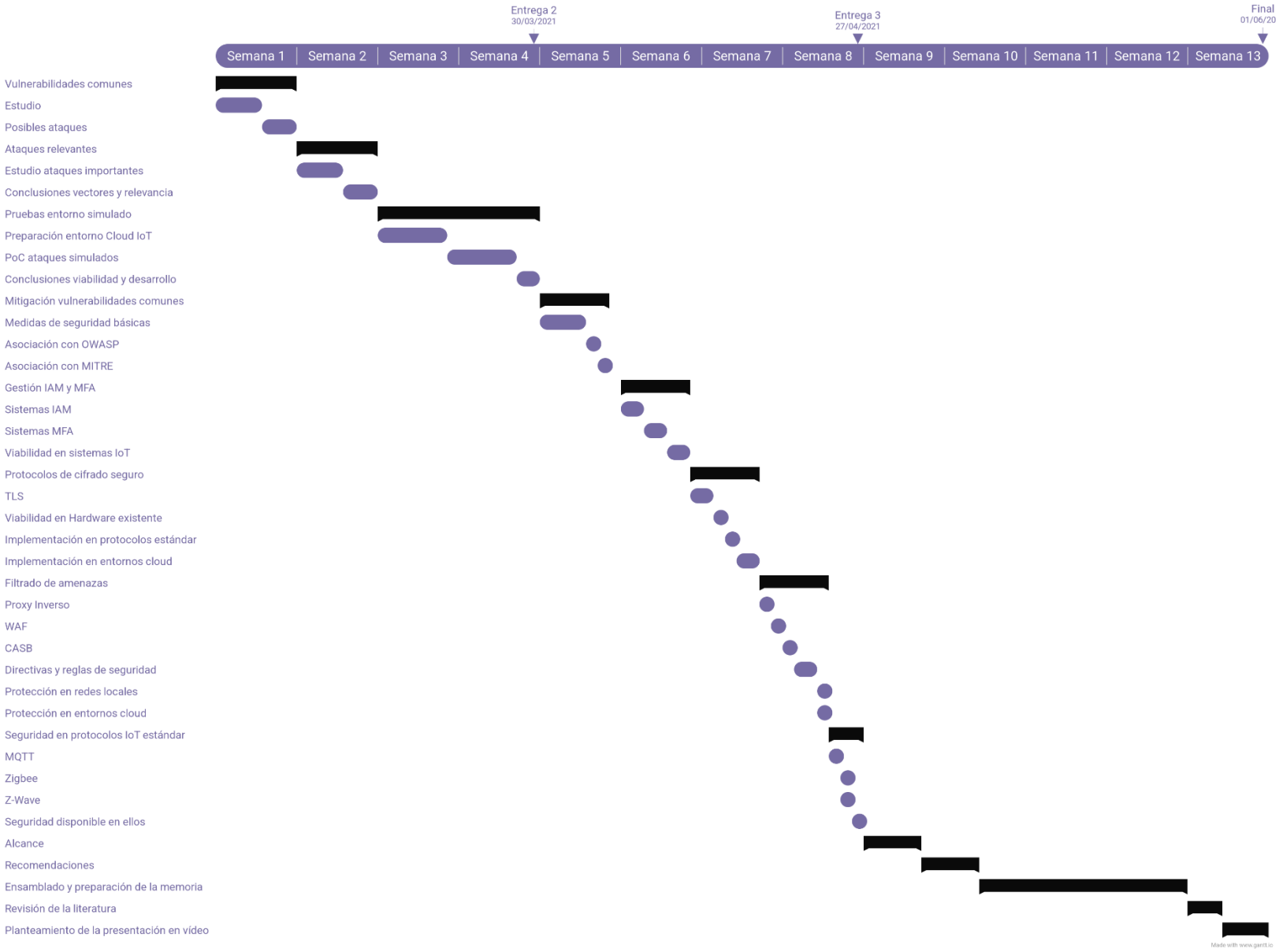
1. Vulnerabilidades comunes
   1.1. Estudio de las vulnerabilidades más comunes y su posible compromiso
2. Ataques relevantes
   2.1. Estudio de algunos de los ataques más importantes en sistemas IoT
   2.2. Conclusiones sobre los vectores de entrada y la relevancia de estos ataques
3. Pruebas en entorno simulado
   3.1. Preparación de un entorno IoT con conexión a Cloud
   3.2. Pruebas de concepto de ataques simulados
   3.3. Conclusiones sobre la viabilidad y el desarrollo de las pruebas

Segunda parte (Medidas de seguridad) - Entrega 3

1. Gestión de identidad y autenticación segura
   1.1. Sistemas IAM
   1.2. Tipos de autenticación
   1.3. Viabilidad en sistemas IoT
2. Protocolos de cifrado seguro
   2.1. TLS
   2.2. Viabilidad en hardware existente
   2.3. Implementación en MQTT con TLS en entornos cloud
3. Filtrado de amenazas
   3.1. Proxy Inverso
      3.1.1. Cacheo de contenido
      3.1.2. WAF
      3.1.3. Enmascaramiento de IP
      3.1.4. Balanceador de carga
   3.2. CASB
   3.3. Protección en redes locales
   3.4. Protección en entornos cloud
4. Seguridad en protocolos IoT estándar
   4.1. MQTT
   4.2. Zigbee
   4.3. Z-Wave

Última parte (Finalización de la memoria y presentación) - Entrega 4

1. Conclusiones
2. Líneas de investigación abiertas
3. Ensamblado y preparación de la memoria
4. Revisión de la literatura


Entrega 2
30/03/2021
Entrega 3
27/04/2021
Final
01/06/20
Semana 1
Semana 2
Semana 3
Semana 4
Semana 5
Semana 6
Semana 7
Semana 8
Semana 9
Semana 10
Semana 11
Semana 12
Semana 13
Vulnerabilidades comunes
Estudio
Posibles ataques
Ataques relevantes
Estudio ataques importantes
Conclusiones vectores y relevancia
Pruebas entorno simulado
Preparación entorno Cloud IoT
PoC ataques simulados
Conclusiones viabilidad y desarrollo
Mitigación vulnerabilidades comunes
Medidas de seguridad básicas
Asociación con OWASP
Asociación con MITRE
Gestión IAM y MFA
Sistemas IAM
Sistemas MFA
Viabilidad en sistemas IoT
Protocolos de cifrado seguro
TLS
Viabilidad en Hardware existente
Implementación en protocolos estándar
Implementación en entornos cloud
Filtrado de amenazas
Proxy Inverso
WAF
CASB
Directivas y reglas de seguridad
Protección en redes locales
Protección en entornos cloud
Seguridad en protocolos IoT estándar
MQTT
Zigbee
Z-Wave
Seguridad disponible en ellos
Alcance
Recomendaciones
Ensamblado y preparación de la memoria
Revisión de la literatura
Planteamiento de la presentación en vídeo
Made with www.gantt.io

Estado del arte

El término IoT es una expresión que se refiere a los objetos comunes que con el avance tecnológico actual se están interconectando a Internet. A día de hoy se tiene acceso prácticamente a Internet desde cualquier lugar. Con la evolución de la tecnología a día de hoy multitud de elementos de uso diario se conectan a internet tanto para controlar los dispositivos de forma remota, como para obtener información de ellos. Con esta nueva forma de interaccionar con elementos de la vida cotidiana, establecemos Internet como una parte necesaria en la vida.

La tendencia es que esto siga evolucionando de forma exponencial. El ejemplo más cercano a esta expansión para el usuario es la domótica, aunque también utilizamos dispositivos wearables que monitorizan nuestras constantes vitales y vida diaria en general. La seguridad de la información ha pasado a cobrar una importancia crucial, dado que el número de dispositivos no para de crecer, y, por tanto, la exposición de nuestros datos en la red. Toda la información que se transfiere puede llegar a ser crítica si los dispositivos conectados no son utilizados de forma correcta.

Tal como dicta el informe de 2014 de HP FORTIFY[1] (FORTIFY, 2016) el 80% de estos dispositivos tienen problemas en su proceso de autenticación y 6 de cada 10 funcionan sobre interfaces de usuario vulnerables. Por ello deben seguirse metodologías de seguridad como las que forman parte de OWASP[2] (OWASP, 2018) o MITRE ATT&CK[3] (MITRE).

A lo largo de este trabajo se comentarán casos reales, pero podemos imaginar en términos de seguridad lo que podría suponer la intervención de un actor malicioso si consiguiese acceso no autorizado a nuestros dispositivos conectados. Aunque generalmente la IoT suele referirse a entornos domésticos, engloba también dispositivos del entorno empresarial, donde las pérdidas podrían ser mucho mayores económicamente hablando.

Hemos de ser conscientes de lo que a día de hoy es una realidad y establecer controles y protocolos que se ajusten a las necesidades de seguridad que surgen a partir del uso de estos sistemas. La falta de seguridad en multitud de ellos ha de ser evidenciada, y en su mayor parte reparada.

En este trabajo se darán a conocer los vectores de ataque más usuales que son utilizados al intentar comprometer la seguridad de un dispositivo IoT, y posteriormente las medidas de protección recomendadas para mitigar estos posibles ataques en su mayor parte para preservar la privacidad de la información interviniente en estos sistemas.

---

[1] “Internet of Things Research Study”. HP FORTIFY - http://fortifyprotect.com/HP_IoT_Research_Study.pdf
[2] “OWASP Top Ten” - https://owasp.org/www-project-top-ten/
[3] “MITRE ATT&CK MATRIX” - https://attack.mitre.org/

Hay multitud de estudios que hacen referencia a esta temática, identificando posibles riesgos asociados a la misma. La materialización de las amenazas de seguridad a las que estamos expuestos al hacer uso de estos sistemas que transfieren información a distintos servicios cloud, pueden afectar a la accesibilidad del dispositivo, a la integridad de la información de la que disponen los servicios de las compañías que hay detrás de estos dispositivos e incluso a la identidad que se genera para cada uno de los usuarios que forma parte de estos sistemas. Y no solo esto, también la pérdida de la disponibilidad puede considerarse uno de los aspectos que más problemas puede generar, sobre todo para entornos industriales, donde un ataque de denegación de servicio puede provocar pérdidas millonarias.

Otro de los factores más importantes a tener en cuenta es la confidencialidad de los datos, tanto de los que residen en nuestros dispositivos, como la de los que se transfieren a servicios en la nube. Debe garantizarse la seguridad de las transmisiones y el almacenamiento durante toda la vida de los datos.

Para que los usuarios de estos dispositivos podamos ejercer nuestros derechos de la mejor manera posible existe tanto una legislación europea como una a nivel español. A nivel europeo, todo dispositivo que haga uso de nuestros datos ha de cumplir la normativa recogida en la GDPR[4] (Parlamento, 2018). Además, si ofrecen servicios en España, también han de regirse por la LOPD[5] (BOE, 2018).

Por todo lo anteriormente mencionado, la seguridad de los sistemas en los que intervienen dispositivos IoT es de vital importancia a día de hoy. Es nuestro deber proteger la información personal de los usuarios. Y de ahí también el porqué de la importancia de este trabajo. En él, se explicarán con detalle cuales son las debilidades comunes de los sistemas que coexisten en el ecosistema IoT y qué medidas existen para poder remediarlas de forma generalista.

Contraseñas débiles, información transferida sin ningún tipo de cifrado, comunicaciones con protocolos muy básicos en los que apenas existen posibilidades de bastionado, redes en las que conviven dispositivos electrónicos de todo tipo sin ningún tipo de aislamiento... todas estas afirmaciones son parte del día a día del ecosistema IoT. Al concluir este trabajo, se habrán estudiado muchos conceptos en materia de seguridad de este tipo de sistemas que nos permitirán tanto auditar posibles fallos de sistemas ya existentes, como proteger por defecto los que se creen a partir de la lectura del mismo.

[4] “General Data Protection Regulation” – https://gdpr-info.eu/
[5] “Ley Orgánica de Protección de Datos” – https://www.boe.es/buscar/pdf/2018/BOE-A-2018-16673-consolidado.pdf

Parte 1: Vectores de ataque

Estudio de vulnerabilidades con mayor impacto en sistemas IoT

A finales de 2020 se superaron los 21.000 millones de dispositivos IoT conectados alrededor del mundo. Estos dispositivos crean una red masiva de aplicaciones inteligentes, coches autónomos y sistemas empresariales completos que funcionan de forma interconectada. Las compañías basadas en la innovación y los desarrolladores de productos electrónicos que conviven ya en este futuro conectado, deben estar continuamente evaluando los riesgos de seguridad que aparecen en estas redes.

OWASP (Open Web Application Security Project), cada cierto tiempo publica un listado[6] (OWASP, 2018) con las vulnerabilidades que más impacto tienen en estos sistemas. Este listado debe estar actualizado porque es en el que se basan la mayor parte de equipos de seguridad para realizar las pruebas pertinentes necesarias, testeando entornos reales pensando en cada posible vector de ataque para mejorar la seguridad de los sistemas que conviven en entornos de IoT.

Por ello, entramos en detalle para explicar las 10 vulnerabilidades con mayor impacto, con un orden que va de mayor a menor número de apariciones:

Contraseñas hardcodeadas, débiles o fáciles de adivinar.

Muchos de los dispositivos IoT que se utilizan actualmente aún utilizan para la conexión tanto local como en la nube contraseñas por defecto. También es habitual que en estos dispositivos podamos encontrar passwords en texto plano o con alguna codificación básica en el propio código de la aplicación. Han de aplicarse las políticas de passwords[7] (OWASP, 2018) recomendadas para asegurar que esto no ocurra. Además, debemos ser conscientes de que actualmente existen aplicaciones basadas en OSINT, para buscar de forma más precisa posibles passwords atendiendo a la información que hay en la red sobre un usuario. Aunque se explicará más adelante en este trabajo, uno de los mayores ataques a redes de IoT se basa en esta vulnerabilidad, la botnet Mirai.

Este tipo de debilidad suele ser explotada con ataques por fuerza bruta a los servicios expuestos en los dispositivos, o incluso con ingeniería inversa a los propios firmwares instalados en ellos.

Mitigación: No hay una política perfecta para crear un password perfecto. Sin embargo, existen bastantes guías con recomendaciones para generar contraseñas seguras. En este caso, ha de hacerse referencia a la documentación MITRE ATT&CK, más concretamente a la sección de políticas de contraseñas[8] (MITRE, 2019). En esta sección se documentan diferentes

---

[6] "OWASP TOP 10 Internet of Things 2018" - https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project

[7] "OWASP Authentication Cheat Sheet" - https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html

[8] "MITRE ATT&CK Password Policies" MITRE, 6 de Junio de 2019- https://attack.mitre.org/mitigations/M1027/

técnicas para reducir el riesgo de posibles ataques a las contraseñas que quieran establecerse.

Servicios de red inseguros

Muchos de los dispositivos inteligentes tienen servicios de red innecesarios o expuestos por defecto. Por ejemplo, muchos disponen de puertos abiertos que proveen acceso al sistema operativo desde servicios que suelen tener agujeros de seguridad.

En los dispositivos inteligentes, cada puerto abierto es una nueva oportunidad de que un actor malicioso pueda ganar acceso al dispositivo, así que el objetivo debe ser mantener el mínimo número de puertos abiertos por defecto para reducir la superficie de posibles ataques.

Al escanear los puertos de dispositivos inteligentes, muchas veces podemos encontrar protocolos antiguos como telnet o servidores HTTP en plano, que tienen muchas vulnerabilidades que ponen en riesgo el dispositivo al completo.

La forma más común para atacar este tipo de vulnerabilidades es utilizando escáners que nos permitan conocer las versiones tanto de protocolos como de servidores instalados en estos dispositivos, y buscando exploits disponibles de dichas versiones. Al ser antiguas probablemente puedan encontrarse vulnerabilidades graves que no hayan sido parcheadas y hacer uso de ellas para tomar el control del dispositivo.

Mitigación: Evitar la apertura de puertos de comunicación siempre que sea posible. Podemos utilizar RDP o VPN. Si la apertura es estrictamente necesaria, no utilizar nunca puertos por defecto. Mantener actualizada la versión de software de los dispositivos para corregir posibles vulnerabilidades en versiones antiguas. Cifrar toda la comunicación que pase por esos puertos. En este caso, MITRE también ofrece una serie de guías[9] (MITRE, 2019) disponibles para la mitigación de posibles vulnerabilidades relativas a servicios de red.

Interfaces de ecosistema inseguras

Bastionar los dispositivos es solo la mitad del proceso. Una implementación segura de un entorno IoT requiere que la seguridad vaya más allá de los dispositivos que forman parte de él. Es necesario asegurar los servicios y los componentes con los que estos dispositivos se comunican. Esto incluye las aplicaciones móviles, web y API[10]'s (Slate, 2019) que permiten a los usuarios interactuar con estos dispositivos de forma remota.

Este vector de ataque es mucho más amplio. Será necesario vulnerar servicios o componentes, que atendiendo a la tecnología en la que estén desarrollados, podrán tener a su vez diferentes vectores de ataque, pero a grandes rasgos,

---

[9] "MITRE ATT&CK Behavior Prevention on Endpoint" MITRE, 11 de Junio de 2019- https://attack.mitre.org/mitigations/M1040/

[10] "¿Qué es una API?: Todo lo que necesitas saber" de Andrew Slate, 8 Julio 2019 - https://www.wrike.com/es/blog/que-es-una-api-necesitas-saber/

consiste en atacar posibles API's o aplicaciones que se conecten con el servicio. Si conseguimos el control de ellas, podremos modificar la comunicación con nuestros dispositivos.

Mitigación: Sin embargo, es posible diseñar API's seguras, e incluso establecer sistemas de prevención de comportamientos sospechosos en caso de que puedan surgir posibles intentos de ataque. Los ataques asociados a este tipo de vulnerabilidad, son los recogidos en la ID de MITRE M1040. Podemos recurrir a ella para verlos en detalle y ver cómo evitar posibles ataques.

Ausencia de mecanismos de actualización segura

Una de las mayores ventajas de los dispositivos conectados es que pueden ser actualizados de forma inalámbrica con actualizaciones de firmware OTA. Esto supone la libertad de mejorar su propio producto, aunque ya se encuentre en el mercado. Las actualizaciones permiten añadir características y solucionar bugs.

Sin embargo, esta misma característica puede convertirse en un vector de ataque importante si no se implementa de manera correcta. Estas actualizaciones deben ser realizadas de manera segura a través de canales cifrados y con la certeza de no dejar dispositivos inutilizados si la actualización no finaliza correctamente.

Mitigación: Si no se realiza una verificación correcta podríamos generar un ataque MitM que nos permita suplantar el servidor o el archivo original, de forma que la actualización que llegue al dispositivo sea distinta de la que se esperaba inicialmente. Por ello sería necesario verificar el servidor de actualización o el firmware con algún tipo de checksum para asegurar la integridad, prevenir el acceso a cuentas con muchos privilegios, y parchear en caso de posibles vulnerabilidades en este proceso. Podemos obtener información ampliada en la guía de actualización de software[11] (MITRE, 2020) de la matriz de MITRE.

Uso de componentes inseguros o no actualizados

La seguridad de la información es una carrera constante para mantenerse siempre al tanto de las nuevas vulnerabilidades descubiertas en las diferentes librarías de software que soportan productos y servicios. Es necesario parchear rápidamente los componentes vulnerables que forman parte de nuestro ecosistema. Dos casos de este tipo de vulnerabilidades fueron "Heartbleed" y "Shellshock" en 2014.

Al igual que en los servicios de red, un posible ataque utilizando este vector de entrada pasa por escanear los servicios para ver las versiones de las librerías o componentes no actualizados. Una vez las obtengamos, hemos de buscar

[11] "MITRE ATT&CK Update Software" MITRE, 9 de Junio de 2020- https://attack.mitre.org/mitigations/M1051/

vulnerabilidades asociadas a las mismas, o encontrar un 0-day en la propia librería para intentar vulnerar el sistema.

Mitigación: Al igual que en el punto anterior, para reducir la exposición ante este tipo de ataques, debemos establecer políticas de actualización adecuadas tanto para servicios como para el propio producto. En el caso de que un componente quede fuera de soporte, hemos de gestionar su actualización nosotros, o cambiar a otro que no se encuentre en este estado. Bastaría con seguir las guías de actualización anteriormente mencionadas (MITRE 1051).

Protección de privacidad insuficiente

La información personal no son solo datos. Si no se maneja de forma adecuada puede llegar a tener un impacto significativo en los usuarios de nuestros dispositivos o redes IoT. Estos dispositivos inteligentes recolectan muchos datos sobre las redes en las que conviven, y sobre los sujetos que los utilizan.

Es obligatorio seguir la legislación, si no puede suponer sanciones económicas importantes. En España estamos sujetos al GDPR y a la LOPD, que tienen que verificarse de forma independiente tanto en la transmisión de datos hagan los dispositivos de la red como en el almacenamiento tanto en dispositivo como en infraestructuras cloud.

La exfiltración de datos es el tipo de ataque más común ante este vector de entrada. Cualquier otra vulnerabilidad asociada al sistema puede permitirnos acceder a información personal si esta no está almacenada de forma correcta o se transmite de manera insegura.

Mitigación: En este caso es bastante más complejo establecer una guía para reducir la exposición a este tipo de ataques. Pueden ser de diversa índole, así que como tal no existe una guía específica para mitigar fugas de datos en MITRE, sin embargo, en este caso OWASP si ofrece una guía de control proactivo[12] (OWASP, 2018), que se basa en que los datos permanezcan la mayor parte del tiempo posible cifrados.

Almacenamiento y transferencia de datos inseguras

Cada vez que se recogen datos en un dispositivo inteligente, estos se envían a través de la red o se almacenan en el propio dispositivo. Esto hace que el potencial de un posible ataque para comprometerlos aumente. El almacenamiento en el dispositivo debe estar siempre cifrado con algoritmos seguros, y en el caso de las transmisiones, estas deberían usar protocolos de transmisión que permitan el cifrado de las comunicaciones como TLS[13] [14] (IONOS, 2020), (Rescorla, 2018). En el caso que nos aplica, si utilizamos comunicaciones mediante UDP, la mayoría de las veces es complicado utilizar TLS directamente, pues no permite el descifrado independiente y no tenemos

[12] "C8: Protect Data Everywhere" OWASP, 2018 - https://owasp.org/www-project-proactive-controls/v3/en/c8-protect-data-everywhere
[13] "TLS: cómo se encripta en internet" IONOS, 15 Julio 2020 - https://www.ionos.es/digitalguide/servidores/seguridad/tls-transport-layer-security/
[14] "The Transport Layer Security (TLS) Protocol Version 1.3" E. Rescorla, Agosto 2018 - https://tools.ietf.org/html/rfc8446

forma de asegurar el orden de llegada. Por ello suelen usarse implementaciones más avanzadas como DTLS.

Asociado con la privacidad en la mayoría de los casos, tanto el almacenamiento como la transferencia de los datos deben cifrarse correctamente. En el caso del almacenamiento un posible ataque pasa por utilizar vulnerabilidades que permitan exfiltrar los datos del sistema, tanto si estos están en plano, como si queremos realizar fuerza bruta al sistema posteriormente a la extracción. Por ello además de cifrados, deben estarlo con un algoritmo seguro. Con respecto a la transferencia es crucial que los datos vayan cifrados, porque si no, con un simple analizador de red podríamos conocer el funcionamiento del sistema y atacarlo modificando peticiones de red.

Mitigación: De la misma forma que con la privacidad, podemos seguir la guía de OWASP[13] que se menciona en la tipificación anterior, pero además han de mantenerse actualizados los protocolos y métodos de cifrado que se utilicen tanto en la transmisión, como en almacenamiento de datos.

Ausencia de gestión de los dispositivos

En muchas ocasiones, en IoT, utilizamos dispositivos que permanecerán conectados al sistema durante mucho tiempo. Es habitual que haya dispositivos de los cuales desconocemos su existencia, ya sea por pérdida de trazabilidad, porque han dejado de reportarnos datos útiles o porque hemos actualizado el dispositivo por otro mejor, es necesario un centro de control que nos permita mantener un control sobre los dispositivos que habitan en nuestro entorno.

La ausencia de gestión de estos dispositivos, los hacen vulnerables a posible software desactualizado o a la posibilidad de sustituirlo por un dispositivo malicioso sin que seamos conscientes de este ataque.

Mitigación: En todo tipo de entornos, pero mucho más en los que son extensos, hay que tener protocolos estrictos tanto para cuando se produzca el alta, como la baja de un nuevo dispositivo en la red. De esta forma, evitaremos en su mayor parte, que existan dispositivos desactualizados en la red. Mientras sea viable hacerlo, todos los dispositivos han de permanecer actualizados, y cuando estos dispositivos salgan de soporte, es conveniente aislarlos o sustituirlos por otros que sí tengan las medidas de seguridad que deseemos establecer. Aún con protocolos estrictos, MITRE ofrece una guía con técnicas para la detección de dispositivos no gestionados basándonos en la monitorización de la red[15] (MITRE).

Ajustes inseguros por defecto

Una gran mayoría de dispositivos se comercializan con ajustes de baja seguridad para hacer más sencilla la instalación para el usuario. Por ejemplo, hay muchos servicios locales funcionando en puertos por defecto y software

[15] “MITRE Shield: Network Monitoring” MITRE - https://shield.mitre.org/techniques/DTE0027/

ejecutándose con permisos de administración. Muchas veces incluso se nos permite acceder al dispositivo para desactivar ajustes de seguridad haciendo los dispositivos incluso menos seguros que cuando los obtuvimos.

Los ataques a través de este vector pueden ser de distinta índole, pero hay que ser conscientes de que una gran mayoría de usuarios no modifica los ajustes por defecto de los dispositivos que adquieren. Por tanto, la afectación de los ataques en redes a través de configuraciones inseguras puede llegar a ser crítica por la facilidad con la que podemos distribuir el ataque.

Mitigación: Para evitar este tipo de vector de ataque, deberían establecerse políticas de análisis de código tanto estático como dinámico, y realizar auditorías de seguridad cada cierto tiempo. A menudo es difícil detectar este tipo de fallos sin la ayuda de software o equipo especializado. OWASP dispone de una guía de referencia para el análisis de análisis de código fuente[16] (OWASP).

Ausencia de bastionado físico

Probablemente uno de los retos de seguridad más significativos a resolver a día de hoy. Una gran parte de dispositivos IoT se basan en microcontroladores básicos como ESP-32, o ESP8266. Para reducir costes, suelen adaptar módulos previamente fabricados en su propio producto, dejando generalmente descubiertas interfaces de comunicación que permiten leer la información que se transmite si conseguimos acceso físico al dispositivo, o incluso instalar un firmware malicioso que modifique las características iniciales del dispositivo o incluso que exfiltre información previamente al cifrado de ésta.

Un ataque común es buscar alguna interfaz de comunicación UART o JTAG con la intención de modificar el firmware original por uno modificado con características maliciosas.

Mitigación: En el caso de los fabricantes de hardware, han de dificultar el acceso no autorizado al dispositivo. Ha de evitarse dejar expuestos puertos de comunicación como UART o SPI. También es conveniente implementar mecanismos de protección contra escritura de firmwares sin firmar, y también mantenerlos cifrados, para evitar posibles volcados mediante una interfaz física.

[16] “Source Code Analysis Tools” OWASP - https://owasp.org/www-community/Source_Code_Analysis_Tools

Vulnerabilidades más relevantes

Estudio de algunos de los ataques y vulnerabilidades más importantes en sistemas IoT

Botnet Mirai

A finales de 2016, el ataque de la botnet Mirai, se convirtió en el mayor ataque de denegación de servicio que se ha lanzado en la historia. Afectó directamente al proveedor de servicios Dyn usando una botnet a través de dispositivos IoT. Sitios importantes como Reddit, Netflix, CNN, o Twitter tuvieron fallos en el servicio durante la ola de ataques. Una vez que los dispositivos son infectados con el malware Mirai, escaneaban internet buscando otros dispositivos vulnerables. Cuando los encontraban, usaban credenciales por defecto para acceder al dispositivo, propagando el malware Mirai. La mayoría de estos dispositivos eran cámaras IP y DVR (Digital Video Recorder).

Una botnet se caracteriza por ser un conjunto de dispositivos conectados a internet que están bajo el control remoto de un tercero. La peculiaridad de Mirai, es que se basa en un ataque simple. En vez de utilizar sistemas complejos de ataque, simplemente utilizaron estos dispositivos en conjunto para, el 12 de octubre de 2016 lanzar un ataque masivo de HTTP flooding contra los servidores de DNS de Dyn, dejando así a multitud de servicios web sin la posibilidad de ofrecer servicio, pues no podían resolverse sus direcciones IP.

El código de la botnet Mirai fue liberado, y previamente al día del ataque masivo hubo algunos intentos de ataque, e incluso de forma posterior, en dispositivos que no habían podido ser parcheados.

De este ataque se sacaron varias conclusiones clave:

- No deben comercializarse dispositivos que no puedan recibir actualizaciones en el firmware o el software.
- Cuando un dispositivo es activado por primera vez, debe ser obligatorio para el usuario cambiar las credenciales de acceso por defecto a cualquiera de los servicios que implemente.
- Las credenciales para los dispositivos IoT deben ser siempre únicas.
- Debemos asegurarnos que los dispositivos siempre disponen de las últimas versiones del firmware y del software para evitar intentos de ataque y las vulnerabilidades presentes en software desactualizado.

Marcapasos de St. Jude

Los ataques a dispositivos IoT no solo se han utilizado contra servicios informáticos. En agosto de 2016, Carson Block, fundador de uno de los grupos de investigación de ciberseguridad más importantes del mundo, publicó un artículo asegurando que los marcapasos instalados por el hospital St.Jude

podían ser hackeados. No fue hasta principios de 2017 cuando se asumió la posibilidad de esta vulnerabilidad[17] (Smith, 2017).

Dispositivos diseñados por laboratorios Abbott's, entre ellos marcapasos cardíacos implantables y dispositivos de terapia para la resincronización cardíaca se consideraron vulnerables. Utilizaban tecnología de radiofrecuencia. Son dispositivos que se instalan bajo la piel con cables dirigidos al corazón, usados por personas con bradicardias o que necesitan resincronización para tratar un fallo cardíaco. Estuvieron afectados los siguientes modelos de dispositivos:

- Accent
- Anthem
- Accent MRI
- Accent ST
- Assurity
- Allure

En agosto de 2017 se aprobó la primera actualización de firmware[18] (FDA, 2017) que resolvía esta vulnerabilidad para reducir el riesgo de los pacientes a un posible ataque de ciberseguridad. La actualización consistía en bastionar la transmisión de datos vía RF que permitía configurar estos dispositivos. Tras la actualización, cualquier intento de comunicación con los dispositivos debe tener autorización para hacerlo. No hay reportes de pacientes que hayan sufrido daños asociados a esta vulnerabilidad de entre los 465.000 usuarios que los tienen implantados en los Estados Unidos. Sin embargo, hay que ser conscientes de que el grupo de investigación en ciberseguridad (MuddyWaters), decidió no publicar la vulnerabilidad.

Monitores de bebés Owlet

Los investigadores de "prpl Foundation" descubrieron una vulnerabilidad en el monitor cardíaco para bebés. La vulnerabilidad permitía hackear el dispositivo de forma que podían modificarse los datos de salida. La vulnerabilidad residía en la capa de conectividad del dispositivo, que levantaba un punto de acceso Wi-Fi sin credenciales de acceso, haciendo posible la decodificación y posterior modificación de los datos que se generaban en la red. De esta forma, podían modificarse los reportes que se generaban en la aplicación que utilizaban estos monitores cardíacos, pudiendo hacer creer que un bebé sano estaba enfermo o viceversa. Aunque tampoco hay reportes relativos a ataques reales que hayan afectado a usuarios del sistema, no deja de ser importante proteger este tipo de sistemas IoT.

---

[17] "465.000 Abbott pacemakers vulnerable to hacking, need a firmware fix" Ms. Smith, 4 de Septiembre de 2017 - https://www.csoonline.com/article/3222068/465000-abbott-pacemakers-vulnerable-to-hacking-need-a-firmware-fix.html

[18] "Firmware Update to Address Cybersecurity Vulnerabilities Identified in Abbott's Implantable Cardiac Pacemakers" 18 de Octubre, 2017 - https://www.fda.gov/medical-devices/safety-communications/firmware-update-address-cybersecurity-vulnerabilities-identified-abbotts-formerly-st-jude-medicals

TRENDnet Webcams

Esta vulnerabilidad fue descubierta por un blogger cuyo pseudónimo es "someLuser". Publicó un artículo que describía como era capaz de encontrar cámaras vulnerables en internet usando el motor de búsqueda Shodan, que permite a los usuarios encontrar dispositivos conectados con búsquedas sencillas. Aunque los usuarios podían configurar un password para las cámaras, el servicio que exponía el stream de vídeo estaba disponible para cualquiera que conociera la dirección de red de la cámara, que consistía en la IP y una secuencia de 15 dígitos que siempre era la misma.

Días después de su publicación en el blog, sus lectores encontraron multitud de cámaras expuestas a internet, incluyendo cámaras situadas dentro de negocios y en habitaciones de niños. Cuando se encontraban estas cámaras, los usuarios empezaron a publicar capturas de las mismas y la localización en Google Maps para identificar las ubicaciones exactas de las mismas.

TRENDnet publicó mediante la BBC que la vulnerabilidad se había añadido al código en 2010, sin embargo, la primera vez que fueron conscientes de ello fue el 12 de enero de 2012. Fueron parcheados 26 modelos vulnerables de cámaras de fabricadas por esta empresa.

El post donde se publicó la vulnerabilidad[19] (someLuser, 2012) aún puede ser visitado a día de hoy, donde podemos ver el proceso detallado para la explotación de la misma, y todos los pasos que dio el Blogger "someLuser" para descubrirla.

Ataque a CANBUS de Jeep

En la conferencia Black Hat USA 2015, los investigadores Charlie Miller y Chris Valasek explicaron en detalle cómo habían conseguido hackear un Jeep Cherokee para la revista WIRED.

Su primer intento consistió en intentar hackear el sistema multimedia del Jeep a través de la conexión Wi-Fi. Descubrieron que era posible porque el password del punto de acceso que creaba el sistema de entretenimiento se generaba de forma automática basándose en el tiempo en el que el coche y su sistema multimedia se arrancaron por primera vez. Esto da bastantes combinaciones posibles, sin embargo, por la matrícula, es posible inferir de forma suficientemente precisa el año y el mes en los que se fabricó el vehículo, de forma que puedes reducir las combinaciones a 15 millones. Si supones que la primera vez que se arranca fue durante el día, esto las reduce aún más, exactamente hasta 7 millones de combinaciones.

Puede realizarse fuerza bruta a la contraseña en menos de una hora. Sin embargo, es bastante difícil mantener una conexión de una hora con un vehículo encendido, pero descubrieron que se basaba en la hora del sistema

[19] "Trendnet Cameras – I always feel like somebody's watching me" someLuser, 10 de enero de 2012 - http://console-cowboys.blogspot.com/2012/01/trendnet-cameras-i-always-feel-like.html

por defecto, más los segundos que tardaba en arrancar la unidad multimedia, por lo que redujeron aún más este tiempo a unos pocos segundos.

Una vez conectados a la red del vehículo, descubrieron que funcionaba sobre una versión de Linux que consiguieron vulnerar y obtener acceso completo al centro multimedia, con esto ya conseguían un mapa de localización del vehículo y un posible ataque a la ubicación del mismo.

Pero la clave de todo el ataque fue un controlador instalado en el vehículo, más concretamente el V850. Las unidades multimedia están aisladas generalmente del bus de comunicaciones CANBUS, por motivos de seguridad. En cambio, este controlador, estaba conectado tanto a este bus como al centro multimedia. Aunque por defecto el V850 estaba programado para ejecutar solo operaciones de lectura sobre CANBUS, este no tenía ningún tipo de verificación a la hora de modificar su firmware, así que, desde la propia unidad multimedia, consiguieron intercambiar el firmware por otro donde sí tenían permisos de lectura.

A partir de este momento los investigadores eran capaces de enviar todo tipo de comandos a través del bus de comunicaciones central del vehículo, pudiendo hacer que cada uno de los componentes del coche funcionara a su antojo. Tomaron control del volante, el motor, la transmisión y el sistema de frenado, y no solo eso, sino que fueron capaces de exponer el control a través de la red móvil, porque todas las unidades multimedia fabricadas por Chrysler, que se instalan en Jeep, llevan conexión directa a la red de Sprint.

Conclusiones sobre los vectores de entrada y la relevancia de estos ataques

Si analizamos las vulnerabilidades que se han detallado en el punto anterior, todas ellas pueden asociarse al Top 10 publicado por OWASP en 2018 y podrían haber sido evitadas estableciendo políticas de seguridad básicas en el desarrollo de estos productos.

Estamos viviendo el auge de los sistemas y los entornos de IoT con los que convivimos en mayor o menor medida, y hay que tener en cuenta la suma importancia de la seguridad en los mismos. Hasta ahora hemos hablado de dispositivos conectados en la mayoría de ocasiones pensando en dispositivos domóticos. Sin embargo, cada vez existen más empresas con sistemas que en su mayor parte forman parte de redes IoT, dispositivos médicos y vehículos conectados de cuya seguridad dependen grandes inversiones económicas e incluso la vida de las personas.

Se necesitan protocolos, normativas y personal especializado que sea capaz de aplicar este tipo de políticas sumamente necesarias a los productos que ya forman y que van a formar parte de nuestras vidas. Los vectores de ataque más utilizados pueden evitarse o reducir la superficie de exposición en productos, en su mayor con conocimientos técnicos relativamente sencillos.

La normativa ha de ser mucho más exigente, impidiendo la venta de productos con una seguridad deficiente y forzando a las empresas a pasar controles y auditorías no solo previamente al lanzamiento si no en todo el ciclo de vida del producto para mantener a los usuarios seguros.

Pruebas en entorno simulado

Preparación de un entorno IoT con conexión a Cloud

Para la preparación del entorno he elegido un VPS (Virtual Private Server) que funciona sobre Azure. El sistema es una versión de Ubuntu Server 18.04 LTS de 64 bits. Una vez tenemos la máquina funcionando, en este caso con un sistema Linux, es necesario abrir el puerto correspondiente al servidor MQTT para poder realizar la comunicación desde nuestro dispositivo domótico.

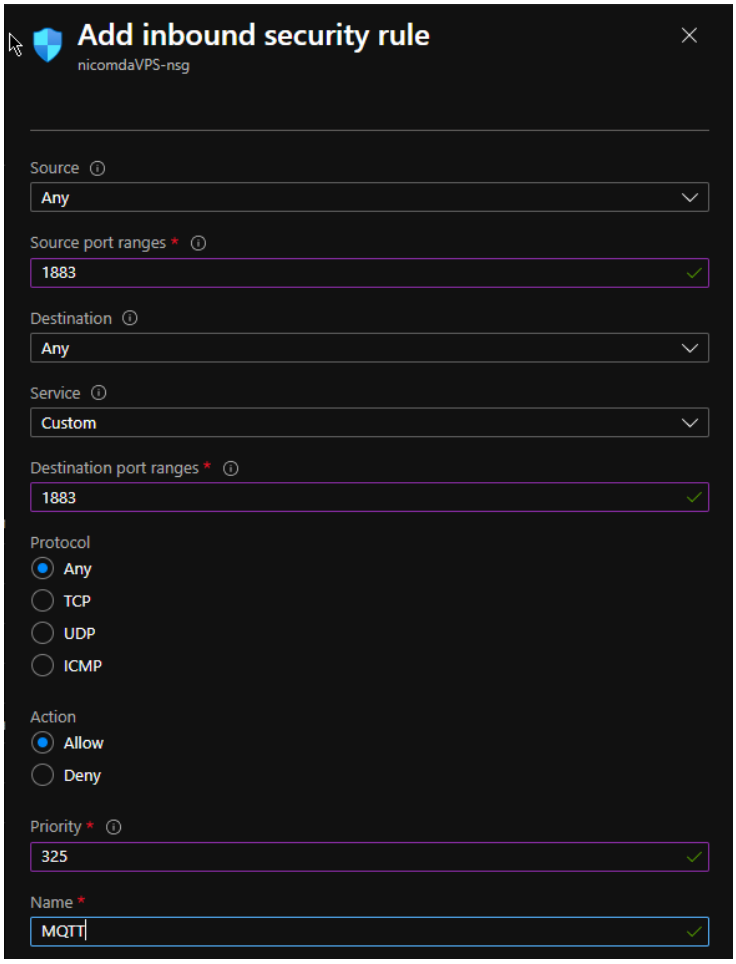


Ilustración 1: Configuración de puerto MQTT en Azure

Además de ello, se ha elegido un dispositivo domótico de la marca Sonoff, un módulo de 2 relés que permite controlar el motor de una persiana automática.

Ilustración 2: Dispositivo domótico con soporte MQTT

Este dispositivo está basado en un microcontrolador ESP-8266, que es uno de los microcontroladores más utilizados comercialmente en sistemas domóticos. Posee un firmware de código abierto llamado Tasmota[20] (Web21), que nos va a permitir modificar algunos parámetros que se necesitan para realizar las pruebas de concepto planteadas.

Pruebas de concepto de ataques simulados

Ataque a la interfaz web del dispositivo mediante fuerza bruta

Simulamos un dispositivo malicioso dentro de la web, y al lanzar un escáner de puertos abiertos hacia el dispositivo, vemos que tiene el puerto 80 sirviendo una interfaz web. Si intentamos acceder, se nos solicita el típico “Basic Authentication” vía HTTP. Sin embargo, al generar una URL inválida el servidor nos responde “File Not Found”. Es una respuesta bastante básica, pero si copiamos la petición en Google tal cual vemos que es la respuesta básica de la librería del ESP-8266 y el ESP-32 así que ya sabemos que el dispositivo que hay detrás debe tener un firmware de ese tipo. Podemos utilizar Burp para analizar la petición de login y ver si existe la posibilidad de un ataque por fuerza bruta.

Ilustración 3: Petición de Login capturada en Burp

Una vez tenemos la petición decodificamos esa autenticación que obtenemos en base64 y vemos que no es más que admin:1234, las credenciales que se han probado. Sabiendo el formato podemos generar un ataque por fuera bruta mediante credenciales genéricas a través del propio Burp. Enviamos la petición al apartado de “Intruder”, creamos el parámetro para el Payload, establecemos el formato y cargamos un diccionario de passwords por defecto.

[20] “Web oficial del firmware Tasmota” - https://tasmota.github.io/docs/

Ilustración 4: Selección de la posición de la carga útil

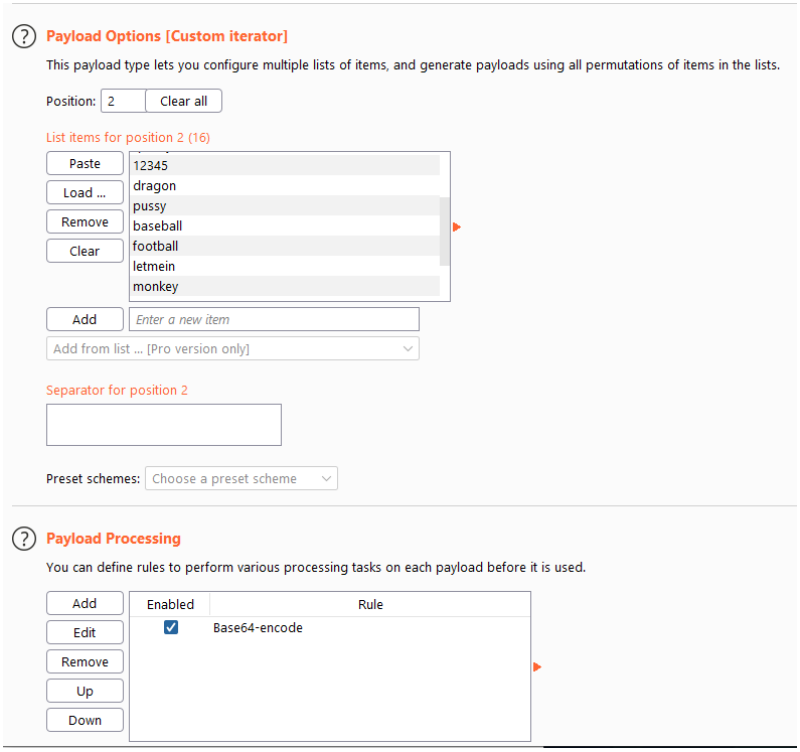


Ilustración 5: Configuración de los parámetros de la carga útil

Hemos simulado el ataque introduciendo un password correcto desde el que obtenemos acceso finalmente al servidor con la siguiente respuesta.

Ilustración 6: Respuesta válida del servidor web

Ataque de sniffing en red local

El servidor de Tasmota funciona por defecto mediante HTTP. Las credenciales viajan en texto plano desde el servidor al cliente, otro posible ataque, consiste en la posibilidad de ver el tráfico con un analizador de red en modo promiscuo. Simplemente hemos de esperar a que un cliente establezca una conexión con el servidor, y sus credenciales se enviarán. Si estamos escuchando en la red, al no llevar ningún tipo de cifrado, podemos verlas. Simplemente están codificadas en Base64.

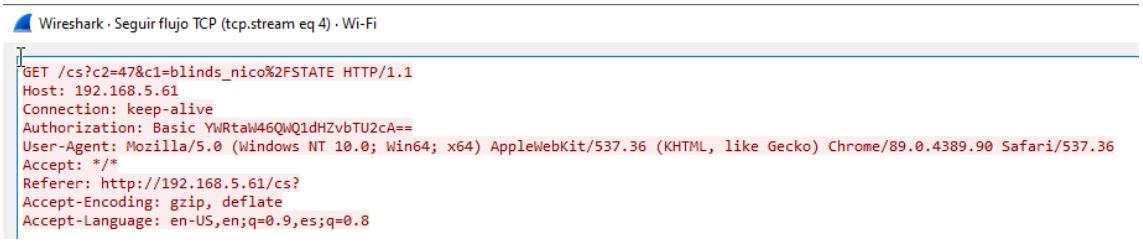


Ilustración 7: Petición HTTP del servidor de Tasmota con el password en texto plano

Podríamos partir de este tipo de ataque para hacer un MitM (Man in the Middle) en red local envenenando las tablas de ARP de los equipos pertinentes, para modificar peticiones antes de que lleguen desde o hacia el servidor, pudiendo combinarlo con ingeniería social para confundir al usuario y e incluso modificar ajustes sin que este lo sepa.

Ataque al servidor de MQTT

El servidor de MQTT que se instala en Linux por defecto, se configura sin credenciales. A no ser que nosotros personalmente las establezcamos en un archivo de configuración, la autenticación con el servidor puede ser anónima. En este caso visualizaremos como podemos obtener la información que llega al servidor desde un cliente local (simulamos el cliente en local para las pruebas). Si nos fijamos en el ataque anterior, obteníamos una captura de la petición HTTP, veamos la segunda parte de esta captura.

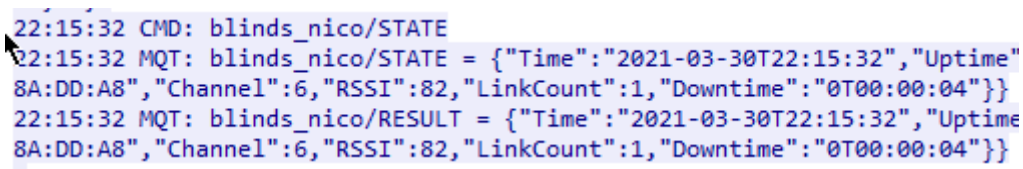


Ilustración 8: Petición HTTP con contenido de comandos y topic MQTT

A estas alturas ya sabemos el firmware que se está utilizando. Con toda esta información vemos que el topic asociado al protocolo MQTT es “blinds_nico”. Si estudiamos el funcionamiento de este firmware y vemos como maneja MQTT, sabremos que el topic se comparte entre todas las peticiones. Si tenemos una petición que devuelve el estado a través del topic “blinds_nico/STATE”, existirá también una petición que permite activar los relés del sistema con mensajes enviados al topic “blinds_nico/POWER1”.

Como el servidor permite autenticación anónima, podemos generar un mensaje desde nuestro cliente local simulado con un comportamiento malicioso. En la siguiente imagen veremos la información que recibimos si nos suscribimos al topic capturado mediante sniffing en red local.

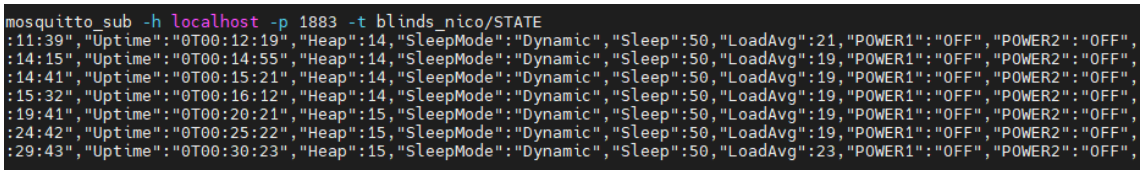


Ilustración 9: Cliente MQTT local en el servidor remoto

Una vez que vemos cómo funciona el servicio, podemos generar el siguiente mensaje malicioso para hacernos con el control de forma remota. Podemos conectarnos tanto en local, como desde cualquier IP, ya que la tenemos al haber conseguido acceso a la interfaz web local previamente. Simplemente debemos generar el siguiente mensaje malicioso, y publicarlo al topic correspondiente. Esto puede realizarse desde otro dispositivo, tal y como se muestra en la siguiente imagen, de esta forma, tendríamos el control del sistema de forma remota.

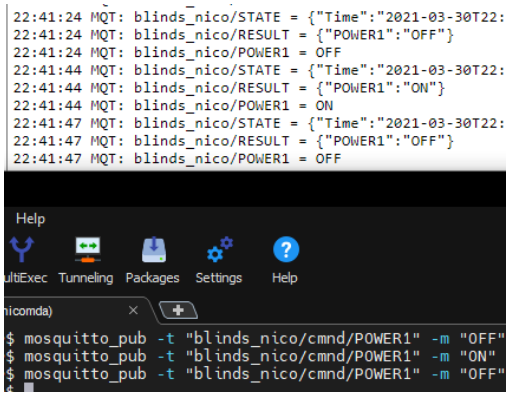


Ilustración 10: Envío de mensajes maliciosos desde cliente MQTT al servidor central

Conclusiones sobre la viabilidad y el desarrollo de las pruebas

MQTT es un protocolo muy potente que funciona de forma jerárquica, sin embargo, tal y como está planteada la implementación por defecto es poco seguro. El servidor no solo debe configurarse con credenciales, si no sobre TLS. Es posible realizar esto, aunque generalmente supone acciones de configuración adicionales que se omiten en la mayoría de los casos. Estas pruebas nos confirman lo vulnerable que puede llegar a ser un sistema domótico, aunque la gestión del mismo se haga a través de la nube. Hay muchos vectores de ataque posibles, aunque nosotros hayamos diseñado estas tres pruebas de concepto que demuestran la debilidad de estos sistemas.

Hay muchas pruebas que no han sido viables ya que estamos restringidos a la explotación de servidores propios, pues hacerlo en ajenos, aunque sea a nivel formativo es ilegal. Tampoco tenemos la posibilidad de atacar aplicaciones comerciales por el mismo motivo. Así que se ha diseñado un entorno de pruebas que puede existir realmente, con la intención de que este tuviera vulnerabilidades explotables para que se pudiesen mostrar riesgos reales de este tipo de implementaciones.

Todos los dispositivos IoT que conectamos a nuestra red, son susceptibles de ser atacados, e incluso de convertirse en atacantes dentro de nuestra propia red. Es por ello que se recomienda aislar a estos dispositivos en subredes adicionales o incluso mediante el uso de VLAN’s.

Además, hemos de elegir marcas que tengan un recorrido en este tipo de mercado, que probablemente se hayan encontrado con problemas similares. Existen multitud de marcas muy poco conocidas que utilizan tanto firmware

como software de baja calidad, que nos expondrá aún más si cabe a posibles ciberataques.

Durante el desarrollo se han dado a conocer las medidas básicas de seguridad que hemos de tener implementadas si desarrollamos o instalamos en algún momento un sistema de estas características.

Parte 2: Medidas de seguridad

Gestión de identidad y autenticación segura

Sistemas IAM

Un sistema de gestión de acceso e identidad (IAM) se encarga de asegurar que las personas de nuestra organización puedan acceder a lo que necesiten para hacer su trabajo. Este tipo de sistemas permiten a la organización gestionar los permisos y roles de los empleados en las aplicaciones. Sin embargo, en este caso no habrían de considerarse como “personas, o empleados” si no tal y como su nombre indica, han de considerarse identidades. Esto es necesario porque, aunque los IAM empezaron como sistemas de gestión de permisos para el personal, a día de hoy también forman parte de estos sistemas aplicaciones, bots y dispositivos robóticos e IoT.

Un IAM es necesario para mejorar dos puntos vitales de las identidades que lo conforman:

La productividad: Una vez que inicias sesión mediante el IAM, esta identidad no ha de volver a preocuparse por gestionar múltiples passwords o el nivel de acceso correcto a sus tareas. No solo permite obtener acceso a toda la suite de herramientas de trabajo, si no que este acceso puede ser gestionado a nivel de grupo o rol en vez de individualmente, reduciendo la carga de trabajo para el equipo de IT.

La seguridad: Tradicionalmente siempre existe el mismo punto inseguro, la contraseña. Si la contraseña o el email de un usuario se filtra, toda la organización se vuelve vulnerable a ataques. Los servicios IAM, consiguen reducir la posibilidad de que estos puntos débiles aparezcan, y permiten procesos de remediación en caso de que ocurra alguna incidencia.

Ya no es solo a nivel de seguridad, sino también a nivel legal, contractual y regulatorio. El estándar europeo, la GDPR, el HIPPA o la ley Sarbanes-Oxley en USA, son de obligado cumplimiento para mantener la seguridad de los datos. Con un IAM, tanto los usuarios como la organización pueden asegurar los estándares de seguridad más altos, hacer el seguimiento, y mantener la transparencia administrativa necesaria en las actividades que desempeñen.

Una solución IAM realiza por lo general, dos tareas:

- Gestión de Identidad: Confirma que el usuario, el software, o el hardware es quien dice ser comprobando sus credenciales en una base de datos. Las soluciones cloud IAM son más seguras y flexibles que la autenticación tradicional vía usuario y password.
- Gestión de Acceso: Asegura un nivel de acceso adecuado. En vez de un usuario y una contraseña para acceder a todo el software, permite gestionar credenciales distintas para cada rol y cada aplicación, aunque posteriormente estas sean transparentes al usuario.

Hasta hace unos años, la mayoría de los sistemas IAM que existían en el mundo empresarial se controlaban desde servidores físicos dentro de la organización, lo que se conoce como un servicio IAM “on-premise”. Sin embargo, a día de hoy, la mayoría de sistemas de gestión de identidad son controlados por un proveedor en la nube para evitar costes de mantenimiento de equipos físicos a la organización y asegurar la disponibilidad en sistemas redundantes. Actualmente ya ni siquiera es necesario tener un VPS (Servidor privado virtual), sino que se comercializan infraestructuras IAM como servicio, llamadas IDaaS.

Tipos de autenticación

SSO (Single Sign-On)

Son sistemas que permiten con solo iniciar sesión mediante una pareja de credenciales (usuario y contraseña), acceder a múltiples aplicaciones y cambiar entre ellas de forma transparente para usuarios, aplicaciones o hardware. Incrementa la productividad y reduce la fricción de uso de los sistemas corporativos. Detallaremos los tipos más comunes:

- Gestión de identidad federada (FIM): Consiste en la relación de confianza que se crea entre dos o más dominios o sistemas IAM, el más conocido es Active Directory Federation Services (Microsoft)
- OAuth: Es un framework específico que puede ser considerado parte de una arquitectura FIM. Parte de la confianza que un usuario tiene hacia un dominio, para que este comparta su información hacia otros.
- OpenID Connect (OIDC): Es una capa de autenticación que funciona sobre OAuth 2.0, para permitir la funcionalidad SSO
- Security Access Markup Language (SAML): Es un estándar abierto para proveer funcionalidad Single Sign-on
- Same Sign-On (SSO): Aunque a menudo suele confundirse con Single Sign-On, existen diferencias, pues no requiere de relaciones de confianza entre las entidades. Depende de la duplicación de las credenciales compartiéndolas entre ambos sistemas cuando es necesario. No es tan segura como las opciones anteriores. Pues estas no comparten directamente las credenciales si no que hacen uso de tokens.

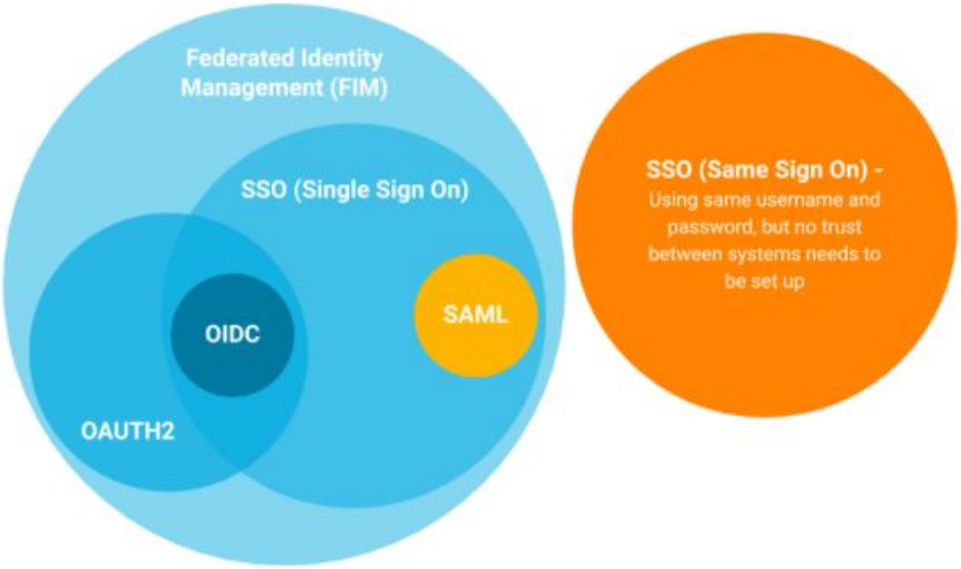


Ilustración 11: Diagrama de SSO. Fuente: OneLogin

MFA (Multifactor Authentication)

Es un sistema que permite añadir una capa de autenticación adicional de protección solicitando a los usuarios al menos dos credenciales de autenticación, además de un nombre de usuario para conseguir el acceso a aplicaciones. Por ejemplo, puede solicitarse el password y un código temporal enviado por correo electrónico o mensaje.

- OTP (One Time Password): Son passwords que pueden usarse una sola vez. Existen varias implementaciones dentro de OTP, pero las más comunes son TOTP y HOTP. En el caso de TOTP, el password rota cada vez que se sobrepasa un tiempo establecido, y en HOTP, solo cuando éste es usado.
- Códigos QR: Permiten autenticar al usuario escaneando un código QR. Pueden utilizar OTP u otros sistemas.
- FIDO (Fast Identity Online): Son llaves de seguridad hardware que, o bien se desbloquean al insertar una clave maestra, y contienen el segundo password de la autenticación, o se solicita la contraseña y la presencia del mismo.

Biometric Autentication

Puede ser utilizada como una de las credenciales para MFA, o como método de autenticación directo. Se basa en el escaneo de fuentes biológicas únicas, ya sea mediante huellas, retina, voz, caras o reconocimiento de presión sanguínea. Ofrecen autenticación muy segura, pero requieren de hardware adicional, como un escáner o sensores, y software de procesamiento.

Risk-based Authentication

Basándose en diferentes niveles de riesgo, permite cambiar de forma dinámica el tipo de MFA, o si se solicita doble factor o no.

Ejemplo: La primera vez que se inicia sesión en un equipo nuevo, se solicitan ambos, posteriormente solo uno de ellos.

Puede basarse en diferentes tipos de información, no solo el primer inicio de sesión. También existen sistemas basados en direcciones IP, nuevo software instalado, presencia de malware o varios de ellos al mismo tiempo.

Viabilidad en sistemas IoT

Como tal, aunque lo analizaremos más adelante, los protocolos estándar que se utilizan en domótica, no poseen características que permitan usar sistemas de autenticación IAM. La mayoría ni siquiera disponen de cifrado, por lo que la implementación de redes IoT en sistemas de gestión de identidad aún está empezando a aparecer.

Sin embargo, ya hay algunas empresas que han implementado capas en un nivel de abstracción más alto con respecto a los protocolos habituales, para añadir estas características de seguridad tan necesarias. Es el caso de Beviwise. Han desarrollado un bróker compatible con MQTT, llamado

MQTTRoute[21] (Hema, 2021), que en su versión 3.1 permite el desarrollo y la instalación de plugins para acceder mediante sistemas IAM.

El desarrollo es realmente reciente, por lo que para certificar su uso aún tendría que probarse en entornos controlados y pasar por los diversos organismos de estandarización. Además, creo que la tendencia ha de hacer que sean los propios protocolos los que permitan este tipo de autenticación, para no depender de productos, y pueda ser algo abierto a todo el mundo.

También hay disponible una integración de IAM en el Core IoT de AWS[22] (Amazon, 2021) (Amazon Web Services). Aunque no toda la funcionalidad de un sistema IAM está disponible, sí que existen políticas, tanto basadas en identidad como en recursos, autorización basada en etiquetas, y roles.

En este caso, es una capa que pertenece al core de IoT disponibles en los servicios de Amazon, y desarrollada encima de MQTT, así que, aunque muy reciente, a día de hoy es viable desplegar un entorno IoT gestionado por IAM en la nube.

Aunque siempre que se habla de sistemas IAM está relacionado con entornos empresariales, a día de hoy los dispositivos IoT obtienen tal cantidad de datos sobre las personas que los utilizan que estos sistemas de seguridad avanzados deben llegar a todos los usuarios.

[21] "MQTTRoute 3.1 – Custom Secure MQTT Authentication Released" Hema, 17 de Febrero 2021 - https://www.bevywise.com/blog/mqttroute-31-released-all-you-need-to-know-about-custom-authentication/

[22] "How AWS IoT works with IAM" - https://docs.aws.amazon.com/iot/latest/developerguide/security_iam_service-with-iam.html

Protocolos de cifrado seguro

TLS (Transport Layer Security)

Es el cifrado con mayor tasa de adopción diseñado para facilitar la seguridad de las comunicaciones y la privacidad en Internet. El caso de uso habitual más común el cifrado de comunicación entre servidores y aplicaciones web, como los navegadores. También puede usarse para cifrar otras comunicaciones como email, mensajería y VoIP. TLS fue propuesto por la IETF (Internet Engineering Task Force), un organismo de estándares a nivel internacional, cuya primera versión fue publicada en 1999. Sin embargo, la versión más reciente es TLS 1.3, que se publicó en 2018.

TLS es una evolución de un protocolo de cifrado llamado SSL (Secure Sockets Layer), desarrollado por Netscape. De hecho, la versión 1.0 de TLS aún se llamaba SSL 3.1. El cambio de nombre es debido a que previamente a su publicación se quiso que dejara de estar asociado a Netscape. Por eso, a veces, ambos acrónimos se usan de forma indistinta.

Sin embargo, sí que ha de hacerse una diferencia con respecto a HTTPS. En este caso, HTTPS es una implementación del cifrado TLS que funciona sobre el protocolo HTTP, por lo que podemos decir que cualquier web que utilice HTTPS, utiliza cifrado TLS. Aunque fue Chrome quien empezó esta tendencia, prácticamente ningún navegador actual permite comunicaciones sin cifrar mediante TLS. Hay 3 componentes principales de los que se encarga TLS:

- Cifrado: Oculta los datos cuando se transfieren de un sitio a otro
- Autenticación: Asegura que los sitios que intercambian información son quien dicen ser
- Verifica que los datos no han sido modificados

Para que una comunicación pueda usar TLS, debe tener un certificado instalado en el servidor de origen (Certificado SSL). Este certificado ha de ser emitido por una autoridad certificadora a la persona propietaria del dominio hacia o desde el que se genera la conexión. Este certificado contiene información sobre el propietario del dominio y la clave pública.

Una conexión TLS es iniciada por una secuencia de paquetes conocida como TLS Handshake. Cuando el usuario se conecta con un servidor protegido mediante TLS, se produce un intercambio de información en este saludo.

- Especifica la versión de TLS que usarán
- Especifican las suites de cifrado
- Autentica la identidad del servidor mediante el certificado
- Genera claves de sesión de cifrado para los mensajes entre los participantes

Cada sesión tendrá unas claves de cifrado distintas, estas claves pueden transferirse mediante un canal no cifrado gracias a la tecnología conocida como criptografía de clave pública[23] (CloudFlare).

Una vez que los datos están cifrados y autenticados mediante la clave pública, se firman con un código de autenticación de mensaje (MAC), que permite al receptor verificar la integridad de los datos.

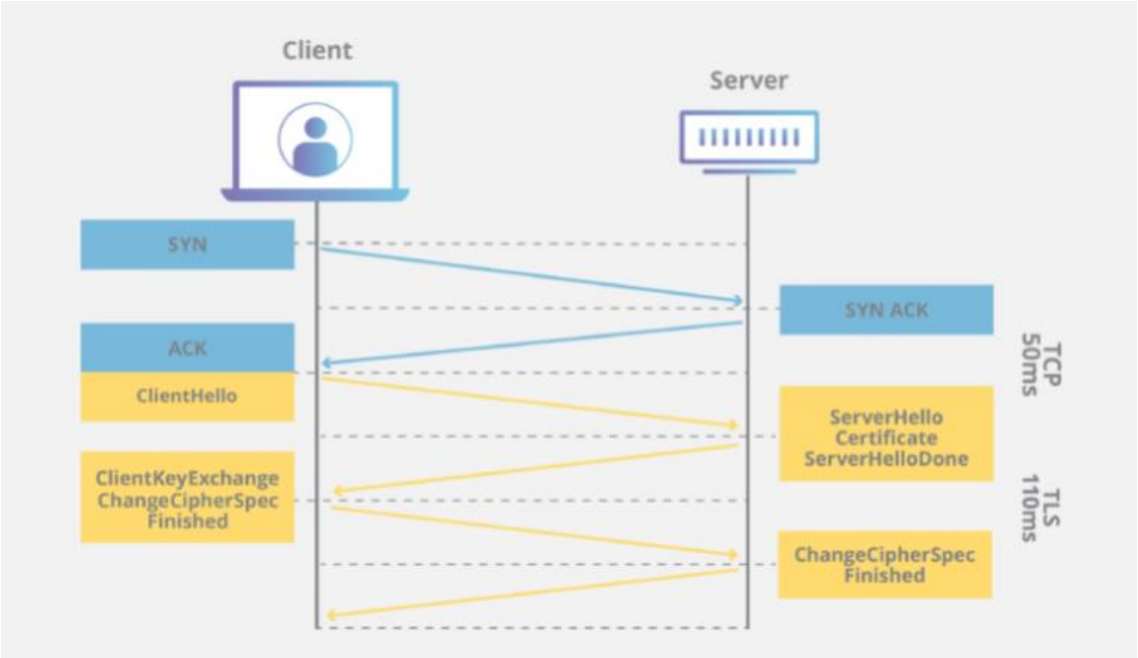


Ilustración 12: Ejemplo de saludo TLS. Fuente: CloudFlare

Las últimas versiones de TLS casi no afectan al rendimiento de la comunicación en un ordenador. Sin embargo, el cifrado que realiza TLS implica un complejo proceso que necesita potencia computacional. Ya no solo aumentan algunos milisegundos de carga, sino también la memoria necesaria para procesar la comunicación.

En las diferentes versiones de TLS, se ha intentado mitigar la latencia potencial que se crea en el saludo de TLS. En la versión 1.3, ya solo es necesario un viaje de ida y vuelta, en vez de los dos que eran necesarios previamente.

Viabilidad en hardware existente

En IoT, al contrario que en los ordenadores, tenemos ciertas limitaciones, por las que, en cierto modo, la expansión de las comunicaciones es muchas veces inexistente, o de lenta expansión. Los dispositivos IoT, generalmente son de hardware limitado, a veces ni siquiera tenemos un procesador como tal, sino simplemente un microcontrolador, que no siempre tiene la arquitectura necesaria para trabajar de forma eficiente con cifrados.

Los primeros Arduino, no eran lo suficientemente potentes como para cifrar en TLS, ni poseían hardware dedicado, así que, aunque ha habido algunos

[23] "How Does Public Key Encryption Work?" CloudFlare - https://www.cloudflare.com/learning/ssl/how-does-public-key-encryption-work/

intentos, prácticamente desperdiciaban toda la potencia de su microcontrolador, convirtiendo el cifrado en inviable. No fue posible un cifrado usable hasta la primera versión con un ARM M3-Cortex (Arduino Due), tanto por temas de memoria como por de potencia computacional. Sin embargo, el Arduino Due era demasiado caro como para usarse en productos a nivel comercial, al menos para el usuario de a pie.

Con la llegada del ESP8266 por parte de Espressif, también hubo intentos, algunos exitosos[24] (Internetofhomethings, 2017). Sin embargo, era necesario modificar algunas librerías del framework original porque la RAM estaba realmente limitada, y se producían reinicios continuos. Por lo que a nivel de producir hardware con este SoC (System on a Chip) y que el mismo estuviese cifrado, aún no era del todo viable.

Pero todo cambia cuando empieza a comercializarse el ESP32. En este caso, Espressif implementó un módulo de hardware para aceleración criptográfica, así que, a día de hoy sí que es viable conectar dispositivos IoT basados en ESP32 y utilizar en dicha conexión el cifrado TLS.

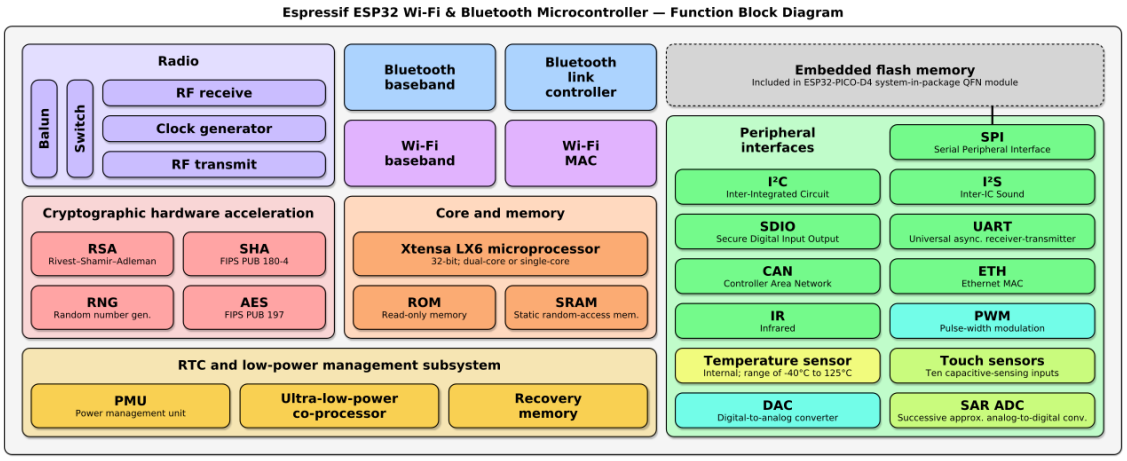


Ilustración 13: Diagrama de bloques del SoC ESP32

Implementación en MQTT con TLS en entorno Cloud

El servidor que utilizamos en las pruebas tiene soporte para MQTT sobre TLS. Sin embargo, hay que generar un CA, y certificados tanto para cliente como para servidor. Es el proceso de generación de certificados habitual.

Generar CA:

$ openssl req -new -x509 -days 365 -extensions v3_ca -keyout ca.key -out ca.crt

[24] "ESP8266 SSL/TLS MQTT Connection" Internetofhomethings, 26 de Abril 2017 - Internet of Home Things » ESP8266 SSL/TLS MQTT Connection

Generando certificados para el bróker:

$ openssl genrsa -out broker.key 2048

$ openssl req -out broker.csr -key broker.key -new

Firmando el certificado del bróker con nuestra CA

$ openssl x509 -req -in broker.csr -CA ../ca/ca.crt -CAkey ../ca/ca.key -CAcreateserial -out broker.crt -days 100

Generando certificados para el cliente:

$ openssl genrsa -out client.key 2048

$ openssl req -out client.csr -key client.key -new

Firmando el certificado del cliente con nuestra CA

$ openssl x509 -req -in client.csr -CA ../ca/ca.crt -CAkey ../ca/ca.key -CAcreateserial -out client.crt -days 100

Durante la generación del archivo csr, tanto para cliente como para servidor, y del ca.crt para la CA, se nos solicitarán datos relativos al dominio. Es importante que el Common Name sea el mismo, si no, no se establecerá la comunicación porque no puede autenticarse la veracidad del dominio del servidor.

Debemos cambiar el puerto y asignar los parámetros para que el bróker funcione a través de TLS.

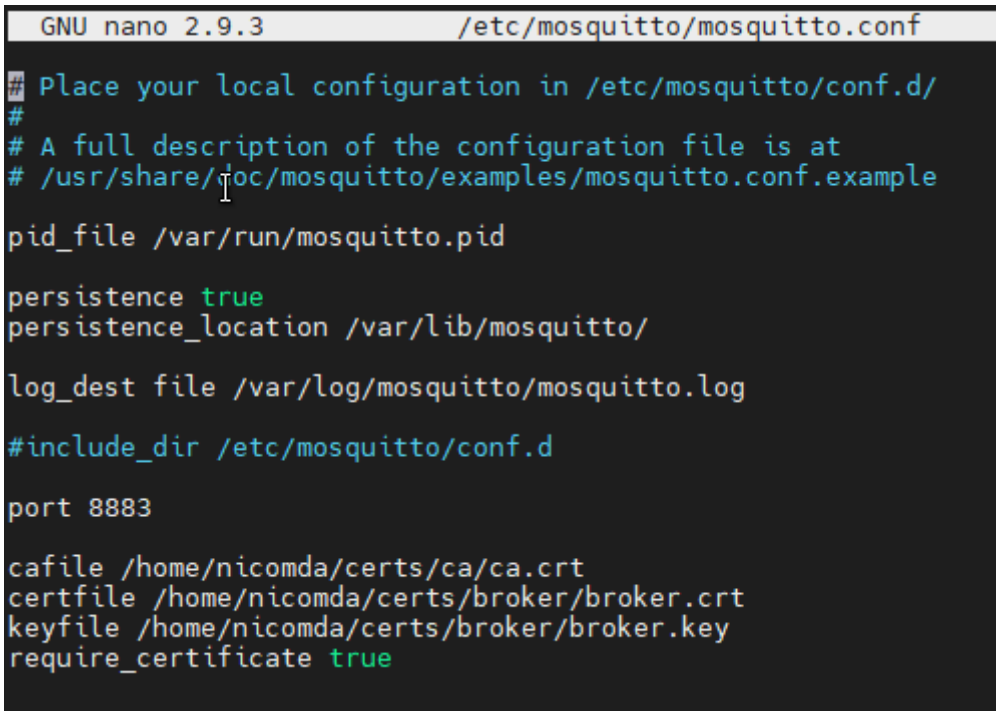


Ilustración 14: Configuración del broker Mosquitto en la VM de Azure

Una vez cambiada la configuración basta con reiniciar el servicio de MQTT del sistema para que cargue la nueva. Dicho esto, la conexión ahora es funcional. Ejecutamos un cliente que se suscribe a un topic y uno que envía uno de prueba mediante TLS.

Ilustración 15: Cliente que publica "HOLA_UOC" al topic /hello mediante TLS

Ilustración 16: Cliente que suscrito al topic /hello mediante TLS recibiendo el mensaje

Aunque en este caso hemos probado la funcionalidad en localhost, el puerto está abierto y la conexión se hace de la misma forma. Sin embargo, los dispositivos de pruebas en las builds del firmware Tasmota, no soportan por defecto TLS. Aunque recompilar el firmware para habilitar el soporte es totalmente viable y funciona correctamente.

Filtrado de Amenazas

Proxy Inverso

Un proxy inverso es un punto de conexión intermedio que se posiciona generalmente en el borde de una red. Recibe las peticiones HTTP que van direccionadas hacia los equipos de la red, actuando como el endpoint de esta.

Funciona de la siguiente forma:

- Recibe petición de conexión de un usuario
- Completa el saludo TCP, terminando la conexión.
- Se conecta con el servidor de origen y redirecciona la petición inicial.

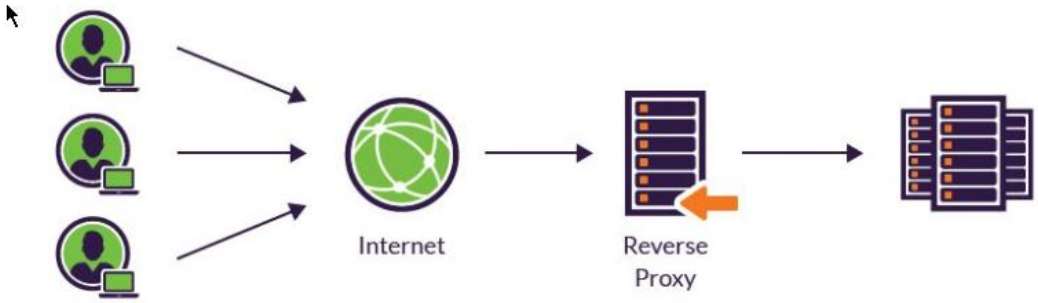


Ilustración 17: Funcionamiento de un proxy inverso

De esta forma, los usuarios que realizan las peticiones no tienen una conexión directa con el servidor, protegiéndolo de posibles ataques, permitiendo filtrar peticiones e incluso comprimir las peticiones que genera el servidor, para reducir los datos transferidos hacia los clientes.

Tienen varias utilidades básicas:

Cacheo de contenido

En las organizaciones pueden estar situados en diferentes localizaciones, donde hay copias de las web comprimidas y cacheadas, de tal forma facilita el contenido de forma más rápida basándose en la localización del cliente, reduciendo tiempos de carga y mejorando la experiencia de usuario.

WAF (Web Application Firewall)

Estos servidores se instalan delante de los servidores de backend, revisando las peticiones para evitar tanto posibles ataques DDoS como posibles paquetes maliciosos.

Los ataques DDoS se ven desviados distribuyendo la carga en una malla de proxies inversos para reducir el impacto. Además, a nivel de firewall, inspeccionan las peticiones con multitud de posibles filtros para evitar cualquier tipo de acción maliciosa contemplada en ellas, y en mucha parte permite ocultar información sobre las máquinas que hay detrás.

Enmascaramiento de IP

Cuando enrutas tráfico entrante a través de un servidor proxy inverso, las conexiones acaban en el proxy y posteriormente se reabren con el servidor de backend. Desde la perspectiva de los usuarios, las peticiones son resueltas por la IP del proxy.

Balanceador de carga

Como los proxies inversos son la puerta entre los usuarios y la aplicación en origen, pueden determinar dónde se envían sesiones HTTP individualmente. Podemos establecer varios servidores backend detrás de un solo proxy, y este redireccionará las peticiones equilibrando la carga de los mismos para crear un entorno de alta disponibilidad.

Debemos tener en consideración este tipo de servidores para la seguridad de sistemas IoT en Cloud, ya que con las utilidades que presentan, son recomendables tanto a nivel de escalabilidad colocando varios bróker detrás de un solo proxy, o incluso de varios si estamos en un entorno realmente grande, como a nivel de seguridad, ya que ocultarán las direcciones reales de los brokers que estamos utilizando, y en muchas ocasiones, incluso de las diferentes redes que formen parte del mismo ecosistema.

CASB (Cloud Access Security Broker)

Un CASB se establece como un punto de control entre el usuario y los recursos o servicios en la nube. Están diseñados por los propios proveedores “Cloud” y garantizan que solo usuarios autorizados tengan acceso a los servicios que se ofrecen. Generalmente son capaces de gestionar tareas como las siguientes:

- Asignación de credenciales
- Inicio de sesión
- Creación de perfiles de acceso
- Registros
- Tokenización
- Cifrado
- Detección de intrusos
- Detección de malware

Mediante Machine Learning, estos servicios son capaces de proteger las comunicaciones de los usuarios hacia servicios cloud como Office365, Salesforce o GSuite. Además, también implementan funcionalidades para proteger de posibles correos maliciosos, phishing, malware y filtraciones de datos.

Ilustración 18: Diagrama de funcionamiento de CASB

Protección en redes locales

Cuando se implementa una infraestructura IoT en una red local, debemos tener en cuenta desde el primer momento la seguridad que va a implementarse. Es mucho más sencillo bastionar un sistema si adoptamos un punto de vista priorizando la seguridad desde el primer momento.

Como primera medida básica, debe de existir una separación completa entre la red IoT, y el resto. Sin embargo, probablemente necesitemos acceso a internet para controlar nuestro sistema local desde la red externa. Pueden plantearse 2 VLAN:

- VLAN_1: Dispositivos habituales o no pertenecientes a la red IoT
- VLAN_2: Dispositivos de la red IoT

En la VLAN_2, residirían tanto sensores como actuadores, gobernados mediante MQTT por un bróker también interno a esta. Ni siquiera sería necesario que la VLAN_2 tuviera conexión a internet. Podríamos utilizar un dispositivo con doble interfaz de red, que perteneciese a ambas VLAN's de forma que sirviera de puente de seguridad desde el que controlaríamos todos los elementos internos de la 2, pero tendríamos acceso al sistema de forma indirecta mediante la 1.

Además, debería establecerse un filtrado por MAC. Es importante tener siempre inventariados los dispositivos de una red IoT, a nivel empresarial, pueden existir cientos de dispositivos conectados, y un dispositivo no controlado, desactualizado, y fuera del inventario puede suponer un posible vector de ataque al sistema. Este tipo de filtrado, nos fuerza a registrar cada una de las direcciones físicas de red de los dispositivos conectados.

Si nuestro sistema tiene acceso externo, a nuestra red, es conveniente establecer un proxy inverso delante de la interfaz del mismo, y cambiar los puertos por defecto en todos los servicios de comunicación que formen parte del sistema. Este proxy, evitaría la interacción directa desde internet con el

servidor donde tengamos alojado el sistema de gestión de la red, y puede servir como WAF parando posibles peticiones maliciosas.

Además, es conveniente levantar un sistema IDS, ya sea en la misma máquina donde reside el proxy inverso, o en otra, para poder tener la red monitorizada ante posibles comportamientos anómalos en la misma.

Todas las conexiones deben ser cifradas, incluso de forma interna a la red, debemos usar HTTP y MQTT con SSL, para mitigar posibles ataques “MitM”.

Protección en entornos cloud

Cuando el sistema IoT se implementa sobre entornos cloud, mucha parte de la seguridad ya viene establecida por defecto, pero tenemos algunas posibilidades adicionales.

Es recomendable no conectar los dispositivos directamente a la nube, si no a través de un bróker secundario. Además, podría establecerse una conexión de VPN entre el entorno físico y la nube. De esta forma, cualquier comunicación perteneciente a la VLAN ya saldría cifrada hacia la nube.

Si estamos en un entorno cloud, tal y como hemos visto durante el desarrollo de este trabajo, es conveniente utilizar sistemas de autenticación MFA, disponibles mediante dichos servicios cloud y actualmente compatibles con algunos brokers específicos, como el core IoT de AWS.

En este trabajo se han estudiado los diferentes brokers que permiten integración con IAM. Es conveniente establecer políticas de seguridad que permitan autenticación mediante diferentes sistemas de autenticación, y que permitan establecer contraseñas por dispositivo.

Además, tenemos la posibilidad de añadir un sistema CASB, que analizará mediante machine learning los comportamientos anómalos que sucedan en la red, aumentando la monitorización y en última instancia la seguridad de la misma.

Seguridad en protocolos IoT estándar

MQTT

La seguridad en este protocolo[25] (IBM) se divide en diferentes capas para mitigar diferentes tipos de ataque. El protocolo en sí mismo especifica mecanismos de seguridad bastante básicos. Recae sobre las propias implementaciones tanto de servidores como de clientes utilizar estándares de seguridad como SSL/TLS o certificados de autenticación x509.

Estas capas de seguridad pueden dividirse en tres niveles:

- Nivel de red
  Recae en el uso seguro de la propia red. Ya sea mediante VLAN’s o VPN para la comunicación entre los clientes y los brokers. Esta solución

[25] “Seguridad de MQTT” IBM - https://www.ibm.com/docs/es/ibm-mq/7.5?topic=ssfksj-7-5-0-com-ibm-mm-tc-doc-tc00150--htm

puede implementarse mediante routers o directamente en los brokers, si estos permiten conexión vía VPN.

- Nivel de transporte

  SSL/TLS permite el cifrado de la comunicación entre cliente y servidor. Existe una verificación de identidad, pero el hardware a día de hoy aún es bastante limitado

- Nivel de aplicación

  El propio protocolo permite la identificación de clientes, y credenciales mediante usuario y contraseña para autenticar los dispositivos. En este nivel también es posible utilizar el cifrado de la propia carga útil, sin necesidad de cifrado en transporte.

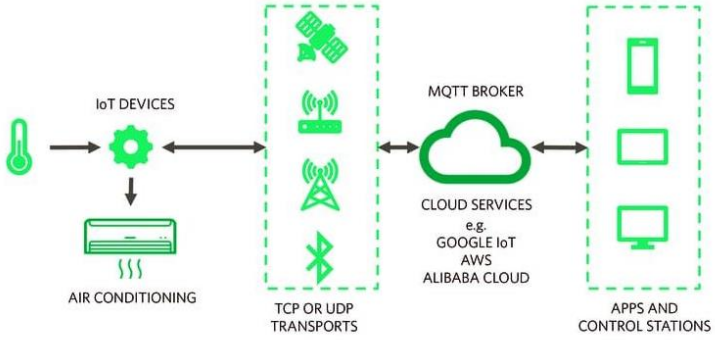


Ilustración 19: Diagrama de un sistema IoT

Aunque muchos clientes llevan un sistema operativo embebido, hay que proteger no solo los tramos de conexión sino también estos sistemas, con más énfasis en los servidores sobre los que funcionan los brókers. Una mala configuración en un servidor puede provocar el compromiso completo del sistema.

Zigbee

La seguridad en la comunicación en este caso es uno de los puntos a favor de Zigbee. Su modelo de seguridad sigue el estándar IEEE 802.15.4[26] (Kivinen, 2017). Este modelo[27] (INCIBE, 2016) proporciona mecanismos de control, cifrado e integridad, asegurando que las tramas transmitidas no sufren manipulación.

Por defecto, el protocolo se basa en el uso de criptografía de clave simétrica. Utiliza 3 tipos distintos de claves, según la red, el grupo de dispositivos o el enlace entre dos de ellos.

- Master Key

[26] "IEEE 802.15.4 Information Element for the IETF" T.Kivinen, mayo de 2017 - https://datatracker.ietf.org/doc/html/rfc8137
[27] "Seguridad en comunicaciones ZigBee" INCIBE, 26 de abril de 2016 - https://www.incibe-cert.es/blog/seguridad-comunicaciones-zigbee

Clave que permite generar las diferentes claves de enlace. Esta ha de obtenerse mediante medios seguros, ya sea mediante preinstalación o un centro de confianza.

- Link Key
  Cifran punto a punto en el nivel de aplicación. Solo es conocida por los elementos que intervienen en el enlace. Se comparte entre dos elementos, y cada emparejamiento dentro de la red tiene una distinta. De esta forma consigue minimizarse el riesgo al no ser necesario intercambiar la clave maestra.

- Network Key
  Se utiliza a nivel de red y es conocida por todos los elementos que pertenecen a la misma. Sirve para comunicaciones de más de dos elementos de forma simultánea.

A pesar de que, en este caso, Zigbee tiene más opciones de seguridad que MQTT por defecto, cae sobre el equipo de implementación la correcta gestión de estas configuraciones para mantener el entorno del sistema IoT seguro.

Z-Wave

Este protocolo a diferencia de los anteriores, no es abierto. Comprende tanto software como hardware y ambos han de estar certificados por la empresa proveedora, en este caso, Silicon Labs.

La implementación de seguridad en este protocolo se basa en las tres siguientes premisas, integridad de los mensajes, confidencialidad y actualidad de los datos.

Las características de seguridad disponibles en el sistema son las siguientes:

- Seguridad en el nivel de aplicación de punto a punto
- Intercambio de claves de red con temporalidad
- Cifrado simétrico por bloques mediante claves de 128-bit
- Nodos seguros y no seguros pueden coexistir en la misma red
- No hay implementación de seguridad ni en la capa dos, ni en la tres del modelo OSI.
- Los nodos no seguros pueden actuar como repetidores para los nodos seguros
- No hay soporte de multicast.

Para mantener la compatibilidad con nodos de las primeras versiones del protocolo, se permite la interacción entre nodos seguros y no seguros. Depende de la implementación de cada fabricante decidir qué comandos deberían tener encapsulación segura.

Conclusiones

Como en cualquier sistema, hemos de entender que la seguridad ha de adaptarse a la actividad, al uso, y atendiendo a la criticidad de los procesos en los que intervenga dicho sistema. Una mayor seguridad requiere generalmente de mayor conocimiento técnico a la hora de desplegar o implementar el sistema en nuestro entorno.

A pesar de esto, actualmente aún dependemos de implementaciones adicionales a los propios protocolos para que nuestros sistemas sean seguros. Más allá del cifrado y la autenticación, sobre todo si nos referimos a entornos empresariales, la seguridad no es suficiente. En cambio, la nube nos permite disponer de capas de seguridad adicionales a las que estos protocolos incorporan por defecto. De esta forma, aunque no directamente desarrollado sobre los protocolos estándar de domótica, la nube nos ofrece soluciones que amplían las características llegando a permitir sistemas que pueden gestionarse mediante IAM.

A nivel empresarial son estas soluciones las que deben utilizarse para mantener la seguridad a un nivel suficiente. AWS, en su IoT core, dispone de implementaciones IAM, que permiten autenticación por dispositivo, e incluso autenticación multifactorial. Sin embargo, es necesario mejorar la seguridad que estos protocolos implementan por defecto para no depender directamente de un servicio ajeno al propio protocolo. Es el propio protocolo el que ha de soportar este tipo de mecanismos de seguridad.

Durante los próximos años, tanto usuarios como empresas vamos a convivir con estos sistemas en los que residen multitud de datos que requieren tanto almacenamiento como transporte seguro. Es cierto que aún son protocolos relativamente recientes, pero las empresas que los desarrollan han de prestar especial atención a esta parte de la implementación, ya que permitirá que estos sistemas se instalen de forma aún más fehaciente si cabe en el día a día de nuestros hogares y nuestras empresas.

Estos entornos IoT requieren disponer ya de la posibilidad de una mayor monitorización y gestión de cada uno de los dispositivos que forman parte de ellos y de la propia red a la que se encuentran conectados.

En el desarrollo del trabajo he adquirido conocimientos más extensos relativos a la seguridad de los sistemas IoT, he desplegado por primera vez un bróker en la nube y tras aplicar la configuración pertinente, he conseguido establecer comunicaciones cifradas. También me ha permitido desarrollar diferentes pruebas de concepto y entender cómo puedo mejorar significativamente bastionar mi propio sistema domótico. En un nivel más alto, me ha hecho entender de forma más profunda la importancia que la ciberseguridad y la privacidad tienen en todos los sistemas que permiten interactuar con el usuario, en este caso, desde el punto de vista de IoT.

Aunque una mayor investigación a nivel de pruebas reales se salía del alcance de este trabajo, quedo satisfecho al ver que prácticamente todos objetivos que

se habían planteado en un principio. Incluso, a lo largo del mismo tuve que rediseñar varias secciones porque durante la realización del trabajo he ido aprendiendo nuevas y mejores formas de organizar algunas de ellas.

Creo que durante todo el desarrollo de este trabajo se ha seguido la planificación de forma coherente, a excepción de algunos momentos puntuales que son comprensibles dada la magnitud y la necesidad del cálculo de la planificación previamente a saber realmente cuánto tiempo podían llevarme las secciones más complejas, sobre todo en las que intervenía más parte técnica que teórica. Los cambios han sido pocos, y simplemente basados en reestructuración en la disposición dentro del trabajo.

Líneas de trabajo futuro

Personalmente creo que el mundo de los sistemas IoT tiene aún muchísimos campos en los que integrarse, y ello conlleva investigaciones y desarrollos extensos para que esta integración pueda hacerse de forma correcta. Ya no solo a nivel local del usuario, si no en la nube y en entornos empresariales aún hay bastante margen de mejora que recorrer.

- Estándares en los protocolos que permitan integración con las medidas de seguridad disponibles en la nube (IAM, MFA, CASB…) por defecto. No son las diferentes implementaciones comerciales quienes deberían dar acceso a este tipo de medidas, sino directamente deberían abarcarse desde el protocolo, para que la expansión de la seguridad dentro de estos sistemas se haga de forma mucho más sencilla.
- Software específico para monitorización empresarial de IoT, incluyendo Honeypots, IDS e incluso EDR para detectar conductas anómalas e intentos de ataque.
- Frameworks y analizadores de políticas de privacidad que cumplan tanto con la GDPR como con la LOPD y que puedan servir como punto de partida para cualquier empresa o particular que pretenda implementar un producto o sistema, pueda ser consciente de estas normativas durante todo el ciclo de desarrollo y producción del mismo.
- Sistemas que permitan establecer restricciones de seguridad a la hora de implementar un entorno domótico. Podría investigarse sobre protocolos para la aplicación automática de políticas de seguridad en los dispositivos que gobiernan la red, ya sea en brokers, routers o switches para mantener unos requisitos de obligado cumplimiento si el usuario quiere utilizar estos dispositivos, en caso contrario, no podría realizarse la conexión.

Glosario

“0-day”: Es una vulnerabilidad nueva para la que no existe aún mitigación disponible.

API: Es el acrónimo de “Application Programming Interface”. Permite la comunicación entre dos aplicaciones.

Backend:

Botnet: Conjunto de ordenadores infectados que son controlables de forma remota o autónoma.

Burp Suite: Herramienta de PortSwigger escrita en Java que hace las funciones de servidor proxy, repetidor y escáner de vulnerabilidades.

CANBUS: Bus de comunicaciones utilizado en vehículos desarrollado por Bosch, interconecta microcontroladores y dispositivos sin necesidad de un host.

Checksum: Es una función matemática que permite detectar cambios en una secuencia de datos para proteger su integridad. Ejemplos: MD5, SHA1.

Contraseñas hardcodeadas: Passwords que permanecen en texto plano ya sea en el código, en la interfaz o en la configuración de un sistema.

DoS: Ataque a un sistema informático que causa que el servicio o recurso que soporta sea inaccesible a los usuarios legítimos del mismo.

EDR: Herramienta que proporciona monitorización y análisis continuo para identificar, detectar y prevenir amenazas.

Endpoint: Se refiere a las URL que reciben o devuelven información a través de una API.

Framework: Es un esquema de trabajo, generalmente utilizado en software, que permite asegurar buenas prácticas y consistencia en los desarrollos. Generalmente suele distribuirse como conjuntos de librerías.

Honeypot: Es un sistema señuelo. Se instala en una red para obtener información sobre posibles ataques informáticos. Algunos hasta permiten simular sistemas.

HTTP Flooding: Es un ataque DDoS que consiste en saturar la aplicación mediante muchas visitas desde lugares distintos. Llega un momento en el que el sistema se sobrecarga e impide el acceso al servidor o la red.

IDaaS: Identity as a Service. Cuando los sistemas IAM o MFA aparecen en la nube con características ampliadas y se ofertan como servicio, estos se consideran IDaaS.

IDS: Es una aplicación de software diseñada para detectar dispositivos o accesos no autorizados en una red.

JTAG: es una interfaz de conexión definida en la norma IEEE 1149.1 que sirve para establecer comunicación con placas PCB, ya sea para la reprogramación o la depuración de aplicaciones embebidas.

LTS: Es un término utilizado para designar a versiones de software que han sido diseñadas para tener soporte durante un tiempo más largo del que suele ser habitual.

MitM: Acrónimo de ataques tipo Man in the Middle.

MITRE: Es una organización dedicada a estudios sobre ciberseguridad y tecnología.

MQTT Broker: Programa que se encarga de recepcionar y distribuir los mensajes enviados por los clientes a través del protocolo MQTT.

MQTT: Protocolo de domótica que funciona de forma jerárquica mediante “topics”, que se implementa sobre la pila TCP/IP.

OSINT: Hace referencia a aplicaciones y métodos de recopilación de información mediante fuentes disponibles, generalmente en internet.

OTA: Actualización de firmware de un dispositivo que se realiza de forma inalámbrica.

OWASP: Es el proyecto abierto de seguridad de aplicaciones web más famoso del mundo actualmente.

Payload: Es el conjunto de datos transmitido que excluye cabeceras y metadatos.

RDP: Protocolo de acceso remoto a equipos, generalmente implementado en sistemas Windows, ya que está desarrollado por Microsoft.

Sniffing: El conjunto de técnicas de escucha en aplicaciones y redes de forma generalmente ilícita.

SPI: Estándar de comunicaciones para la transferencia de información entre circuitos integrados.

UART: Dispositivo que controla los puertos de serie. Se encuentra integrado generalmente en la placa base o en una tarjeta que permite adaptar dispositivos.

VLAN: Método para crear redes lógicas independientes dentro de una misma red física.

VPN: Tecnología que permite conectarse a una red privada a través de internet, generalmente estableciendo una comunicación cifrada.

VPS: Es una partición virtual dentro de un servidor físico en la nube. Tienes acceso mucho más amplio para instalar software y realizar configuraciones diversas que en un hosting. Literalmente tienes disponible un sistema operativo “restringido” a tu partición.

WAF: Firewall de aplicaciones web, que sirve para bloquear o redireccionar tráfico HTTP hacia y desde una aplicación web.

Wearable: Dispositivos electrónicos que son “vestibles”, podemos encontrar desde Smartbands hasta pantalones, o mochilas.

Zigbee: Protocolo de comunicación inalámbrico creado bajo el estándar IEEE 802.15.4.

Z-Wave: Estándar internacional tanto hardware como software para sistemas de control domóticos en viviendas.

Bibliografía

Amazon. 2021. How AWS IoT works with IAM. [En línea] 2021. [Citado el: 10 de Abril de 2021.] https://docs.aws.amazon.com/iot/latest/developerguide/security_iam_service-with-iam.html.

BOE. 2018. Ley Orgánica de Protección de Datos. [En línea] 2018. [Citado el: 15 de Marzo de 2021.] https://www.boe.es/buscar/pdf/2018/BOE-A-2018-16673-consolidado.pdf.

CloudFlare. How Does Public Key Encryption Work? [En línea] [Citado el: 12 de Abril de 2021.] https://www.cloudflare.com/learning/ssl/how-does-public-key-encryption-work/.

FDA, EEUU. 2017. Firmware Update to Address Cybersecurity Vulnerabilities Identified in Abbott's Implantable Cardiac Pacemakers. [En línea] 18 de Octubre de 2017. [Citado el: 27 de Marzo de 2021.] https://www.fda.gov/medical-devices/safety-communications/firmware-update-address-cybersecurity-vulnerabilities-identi.

FORTIFY, HP. 2016. Informe: "Internet of Things Research Study". [En línea] 2016. [Citado el: 10 de Marzo de 2021.] http://fortifyprotect.com/HP_IoT_Research_Study.pdf.

Hema. 2021. MQTTRoute 3.1 – Custom Secure MQTT Authentication Released. [En línea] 17 de Febrero de 2021. [Citado el: 9 de Abril de 2021.] https://www.bevywise.com/blog/mqttroute-31-released-all-you-need-to-know-about-custom-authentication/.

IBM. Seguridad de MQTT. [En línea] [Citado el: 3 de Mayo de 2021.] https://www.ibm.com/docs/es/ibm-mq/7.5?topic=ssfksj-7-5-0-com-ibm-mm-tc-doc-tc00150--htm.

INCIBE. 2016. Seguridad en comunicaciones ZigBee. [En línea] 26 de Abril de 2016. [Citado el: 13 de Mayo de 2021.] https://www.incibe-cert.es/blog/seguridad-comunicaciones-zigbee.

Internetofhomethings. 2017. ESP8266 SSL/TLS MQTT Connection. [En línea] 26 de Abril de 2017. [Citado el: 11 de Abril de 2021.] https://internetofhomethings.com/homethings/?p=1820.

IONOS. 2020. TLS: cómo se encripta en internet. [En línea] 15 de Julio de 2020. [Citado el: 6 de Abril de 2021.] https://www.ionos.es/digitalguide/servidores/seguridad/tls-transport-layer-security/.

Kivinen, T. 2017. IEEE 802.15.4 Information Element for the IETF. [En línea] Mayo de 2017. https://datatracker.ietf.org/doc/html/rfc8137.

MITRE. 2019. MITRE ATT&CK Behavior Prevention on Endpoint. [En línea] 11 de Junio de 2019. [Citado el: 20 de Marzo de 2021.] https://attack.mitre.org/mitigations/M1040/.

—. MITRE ATT&CK MATRIX. [En línea] [Citado el: 13 de Marzo de 2021.] https://attack.mitre.org/.

—. 2019. MITRE ATT&CK Password Policies. [En línea] 6 de Junio de 2019. [Citado el: 20 de Marzo de 2021.] https://attack.mitre.org/mitigations/M1027/.

—. 2020. MITRE ATT&CK Update Software. [En línea] 9 de Junio de 2020. [Citado el: 22 de Marzo de 2021.] https://attack.mitre.org/mitigations/M1051/.

—. MITRE Shield: Network Monitoring. [En línea] [Citado el: 26 de Marzo de 2021.] https://shield.mitre.org/techniques/DTE0027/.

OWASP. 2018. C8: Protect Data Everywhere. [En línea] 2018. [Citado el: 5 de Abril de 2021.] https://owasp.org/www-project-proactive-controls/v3/en/c8-protect-data-everywhere.

—. 2018. OWASP Authentication Cheat Sheet. [En línea] 2018. [Citado el: 19 de Marzo de 2021.] https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html.

—. 2018. OWASP TOP 10 Internet of Things 2018. [En línea] 2018. [Citado el: 16 de Marzo de 2021.] https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project.

—. 2018. OWASP Top Ten. [En línea] 2018. [Citado el: 9 de Marzo de 2021.] https://owasp.org/www-project-top-ten/.

—. Source Code Analysis Tools. [En línea] [Citado el: 27 de Marzo de 2021.] https://owasp.org/www-community/Source_Code_Analysis_Tools.

Parlamento, Europeo. 2018. General Data Protection Regulation. [En línea] 25 de Mayo de 2018. [Citado el: 15 de Marzo de 2021.] https://gdpr-info.eu/.

Rescorla, E. 2018. The Transport Layer Security (TLS) Protocol Version 1.3. [En línea] Agosto de 2018. [Citado el: 6 de Abril de 2021.] https://tools.ietf.org/html/rfc8446.

Slate, Andrew. 2019. ¿Qué es una API?: Todo lo que necesitas saber. [En línea] 8 de Julio de 2019. [Citado el: 21 de Marzo de 2021.] https://www.wrike.com/es/blog/que-es-una-api-necesitas-saber/.

Smith, Ms. 2017. 465.000 Abbott pacemakers vulnerable to hacking, need a firmware fix. [En línea] 4 de Septiembre de 2017. [Citado el: 28 de Marzo de 2021.] https://www.csoonline.com/article/3222068/465000-abbott-pacemakers-vulnerable-to-hacking-need-a-firmware-fix.html.

someLuser. 2012. Trendnet Cameras – I always feel like somebody's watching me. [En línea] 10 de Enero de 2012. [Citado el: 28 de Marzo de 2021.]

http://console-cowboys.blogspot.com/2012/01/trendnet-cameras-i-always-feel-like.html.

Web oficial del firmware Tasmota. [En línea] [Citado el: 28 de Marzo de 2021.] https://tasmota.github.io/docs/.