plain]

plain]

[

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Tècnica Superior d'Enginyeria
de Telecomunicació de Barcelona

Deployment of a security testbed for IoT

Treball de Fi de Grau
presentat a l'Escola Tècnica Superior
d'Enginyeria de Telecomunicació de Barcelona
de la Universitat Politècnica de Catalunya
per

Joel Otero Masplà

En compliment parcial
dels requisits per a l'obtenció del
Grau en Enginyeria de Tecnologies i Serveis de Telecomunicació

Director/a: Olga León Abarca
Ponent: *{Nom del ponent (si s'escau)}*

Barcelona, Juny de 2025

Resum

Cada exemplar del Treball de Fi de Grau (TFG) ha de contenir un Resum, que és un breu extracte del TFG. En termes d'estil, el Resum hauria de ser una versió reduïda del projecte: una introducció concisa, un compendi dels resultats i les principals conclusions o arguments presentats en el projecte. El Resum no ha de superar les 150 paraules i cal que estigui traduït al català, castellà i anglès.

Resumen

Cada ejemplar del Trabajo de Fin de Grado (TFG) debe incluir un Resumenque es un breve extracto del TFG. En cuanto al estilo, el Resumen debería ser una versión reducida del proyecto: una introducción breve, un resumen de los resultados principales y las conclusiones o argumentos principales presentados en el proyecto. El Resumen no debe exceder las 150 palabras y debe estar traducido al catalán, castellano e inglés.

Summary

Each copy of the Bachelor's Thesis (TFG) must include a Summary, which is a concise abstract of the TFG. In terms of style, the Summary should be a condensed version of the project: a brief introduction, a summary of the main results, and the conclusions or key arguments presented in the project. The Summary should not exceed 150 words and must be translated to catalan, spanish and english.

Podeu incloure una pàgina de Dedicatòries just abans de la pàgina d'Agraïments, però no és un requisit.

Agraïments

És apropiat, però no obligatori, declarar l'extensió de l'ajuda aportada per persones de l'*staff*, companys/companyes d'estudis, tècnics/ques o altres en la col·lecció de dades, disseny i construcció del prototip, l'anàlisi de dades, l'execució dels experiments i la preparació del projecte (incloent l'ajuda editorial). A més a més, és apropiat reconèixer la supervisió i la direcció donada pel tutor/a.

Historial de revisió i aprovació

Revisió	Data	Autor(s)	Descripció
1.0	dd/mm/yyyy	AME	Creació del document
1.1	dd/mm/yyyy	AME, JPV	Correcció d'errors
2.0	dd/mm/yyyy	AME, MLO	Versió revisada
4.0	dd/mm/yyyy	AME	Versió final

LLISTA DE DISTRIBUCIÓ DEL DOCUMENT

Rol	Cognom(s) i Nom
[Estudiant]	
[Director del projecte]	
[Director 2 (si aplica)]	

Escrit per:		**Revisat i aprovat per:**	
Data	dd/mm/yyyy	Data	dd/mm/yyyy
Nom	Xxxxxxx Yyyyyyy	Nom	Xxxxxxx Yyyyyyy
Rol	Autor del projecte	Rol	Director del projecte

Índex

Índex de figures

Índex de taules

Sigles

[heading=chapter*, display=all, include=abbrev, name=Sigles i acrònims,]

[

CAPÍTOL 1

Introducció

Una Introducció que estableix clarament la justificació de la tesi que inclogui:

1. Objectius del treball.
2. Requisits i especificacions.
3. Mètodes i procediments, citant si aquest treball és una continuació d'un altre projecte o utilitza aplicacions, algoritmes, programari o maquinari desenvolupat anteriorment per altres autors.
4. Pla de treball amb tasques, fites i un diagrama de Gantt.
5. Descripció de les desviacions del pla inicial i incidències que poden haver ocorregut.

Els capítols mínims que aquest document de TFE hauria de tenir es descriuen a continuació; no obstant això, poden tenir noms diferents i es poden afegir més capítols.

1.1 Objectius del treball

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1.2 Requisits i especificacions

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

1.3 Mètodes i procediments

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

1.4 Pla de treball

Normalment les figures i taules es col·loquen en els entorns `\figure` i `\table`, que poden flotar lliurement en el document. Pots identificar cada flotant amb un `\label`

CAPÍTOL 2

Estat de l'art

2.1 Internet of Medical Things (IoMT)

L'Internet of Medical Things (IoMT) és una branca específica de l'Internet de les Coses (IoT) aplicada a l'àmbit sanitari. Consisteix en una xarxa creixent de dispositius mèdics intel·ligents que poden recopilar, processar i transmetre dades clíniques amb l'objectiu de millorar la qualitat assistencial, facilitar el monitoratge de pacients i optimitzar els processos hospitalaris.

Aquest ecosistema connectat inclou una gran varietat de dispositius, que poden ser tant portables com fixes, i que cobreixen des del seguiment de signes vitals fins al control automatitzat de tractaments. Alguns exemples habituals són:

- **Monitors cardíacs:** permeten controlar l'activitat del cor de manera contínua.
- **Pulsòmetres i termòmetres intel·ligents:** ofereixen dades precises i fàcilment accessibles.
- **Inhaladors i bombes d'insulina intel·ligents:** poden registrar l'ús i ajudar a ajustar el tractament.
- **Implants mèdics connectats:** permeten registar l'estat d'un pacient a temps complert. Per exemple marcapassos o neuroestimuladors.
- **Sistemes de dosificació automàtica de medicació:** especialment útils en pacients amb malalties cròniques.
- **Equips hospitalaris intel·ligents:** com llits monitoritzats o sistemes de seguiment de pacients dins d'unitats de cures intensives.

El creixement de l'IoMT s'ha vist impulsat per diversos factors, com la miniaturització dels sensors, el progrés tecnològic en dispositius mèdics, l'augment de la digitalització sanitària, i la necessitat creixent de models d'atenció centrats en el pacient i orientats a la prevenció i el seguiment continuat. A més, la pandèmia de la COVID-19 va accelerar

l'adopció de solucions de monitoratge remot, que han consolidat l'ús d'aquest tipus de dispositius fora dels entorns hospitalaris convencionals.

L'ús generalitzat d'aquesta tecnologia permet una atenció més personalitzada i basada en dades, alhora que facilita la detecció precoç de complicacions i una millor gestió dels recursos sanitaris. També contribueix a reduir la necessitat de desplaçaments i hospitalitzacions, millorant l'accessibilitat a l'atenció mèdica, especialment en zones rurals o amb menys infraestructures.

Amb una adopció creixent tant en entorns clínics com domèstics, s'espera que l'IoMT sigui una peça clau en la transformació digital del sistema sanitari en els pròxims anys, aportant beneficis tant per als pacients com per als professionals de la salut. [8]

2.2 Seguretat en entorns IoMT

Donat el creixement de l'us de dispositius IoMT, aquesta mateixa expansió comporta un augment significatiu de la superfície d'exposició a ciberatacs. A més a més, s'espera que a mesura que avança la seva adopció, aquests dispositius siguin més determinants en les tasques mèdiques, la qual cosa pot impilcar una major criticitat en cas de ciberatac.

A diferència dels sistemes informàtics convencionals, els dispositius IoMT sovint operen en entorns amb recursos computacionals limitats (processador, memòria, energia), i moltes vegades han estat dissenyats amb una orientació funcional, no pas de seguretat. Això els fa especialment vulnerables a atacants que poden es poden aprofitar de configuracions per defecte, manca d'actualitzacions, credencials febles o vulnerabilitats en els protocols de comunicació. A més, la connexió d'aquests dispositius mitjançant xarxes Wi-Fi o altres canals sense fils exposa el sistema a atacs com l'escolta (sniffing), suplantació de dispositius (spoofing), atacs de denegació de servei (DoS) entre altres.

Un dels aspectes més crítics del risc en entorns IoMT és la naturalesa de les dades que gestionen. Les dades mèdiques són altament sensibles i personals. Un accés no autoritzat pot vulnerar drets fonamentals com la privacitat i tenir conseqüències legals greus per a les institucions sanitàries. En aquest context, la ciberseguretat en l'àmbit IoMT no es pot considerar un afegit posterior al desplegament dels sistemes, sinó un requisit fonamental des de la fase de disseny. Això, és especialment rellevant en entorns on les conseqüències d'un atac poden tenir un impacte directe sobre la salut i la seguretat física dels pacients.

Però, és important destacar que la protecció dels sistemes IoMT també ha de ser escalable i adaptable. L'amenaça no és estàtica, i els vectors d'atac evolucionen constantment.

Davant d'aquesta realitat, la recerca en ciberseguretat per a l'IoMT s'està orientant cada cop més cap a solucions dinàmiques, com ara IDS/IRS basats en aprenentatge automàtic que permetin detectar patrons anòmals de comportament i actuar de forma proactiva. En aquest sentit, la generació de datasets reals que simulin tant trànsit legítim com maliciós en entorns IoMT esdevé una peça clau per entrenar i validar aquestes solucions emergents. Aquestes solucions han estat tractades en artícles com [2]. També la caracterització de

vulnerabilitats connegudes ha estat tractada en artícles com [18] o [16] que han estat utlitzats com a referència per a la recopliació d'atacs i la generació de datasets.

2.3 Message Queuing Telemetry Transport (MQTT)

El Message Queuing Telemetry Transport (MQTT) és un protocol de missatgeria lleuger dissenyat per a la comunicació entre dispositius amb recursos limitats en xarxes poc fiables o amb amplada de banda reduïda. Aquest protocol s'ha convertit en un estàndard de facto en moltes aplicacions IoT, inclòs l'àmbit de l'Internet of Medical Things (IoMT), per la seva eficàcia, simplicitat i facilitat de desplegament.

Desenvolupat originalment per IBM l'any 1999, MQTT segueix un model de comunicació publish/subscribe, que afavoreix la desconnexió temporal dels nodes i la minimització de l'ús de la xarxa, dos requisits habituals en xarxes IoT.

En una arquitectura MQTT, el component central és el broker, un servidor que actua com a intermediari entre els dispositius que publiquen dades (publishers) i els que les reben (subscribers). Els dispositius no es comuniquen directament entre ells, sinó que ho fan a través del broker, que rep els missatges publicats en un tema determinat (tòpic) i els redirigeix als clients que s'han subscrit a aquest tema. Aquesta arquitectura desacoblada simplifica el disseny de sistemes escalables i resilients. A l'àmbit IoMT, aquesta estructura és especialment útil per gestionar sensors mèdics que generen dades de manera periòdica, com ara nivells de glucosa, senyals d'electrocardiograma (ECG), o mesures de tensió arterial. Aquests sensors poden publicar lectures de manera eficient al broker MQTT, i altres components del sistema (com bases de dades, aplicacions clíniques o sistemes d'alerta) poden consumir aquesta informació segons les seves necessitats.

El protocol MQTT opera habitualment sobre TCP/IP, utilitzant el port 1883 per a connexions no segures i el port 8883 quan es fa servir TLS (Transport Layer Security) per protegir la transmissió. Entre les característiques tècniques més destacades d'MQTT, podem ressaltar:

- **Qualitat del servei (QoS):** MQTT ofereix tres nivells de abilitat en el lliurament de missatges, cosa que permet ajustar el comportament segons els requisits de l'aplicació.
- **Sessions persistents:** Un missatge es pot marcar com a retained perquè quedi emmagatzemat al broker i sigui enviat automàticament als nous subscriptors del topic. Això permet garantir que les dades més recents estiguin disponibles en tot moment, encara que el dispositiu que les va enviar originalment ja no estigui actiu.
- **Protocol lleuger:** Amb una capçalera mínima de només 2 bytes, MQTT genera molt poca sobrecàrrega, cosa que el fa extremadament ecient per dispositius amb CPU limitada, poca memòria RAM o connexions de xarxa inestables o intermitents.
- **Model desacoblat (publish/subscribe):** Els clients no necessiten conèixer ni l'adreça ni l'estat dels altres dispositius. Això facilita l'escalabilitat i la fexibilitat

del sistema, ja que els rols de publicador i subscriptor poden canviar dinàmicament.

- **Jerarquia de temes (topics):** Els topics MQTT segueixen una estructura jeràrquica cosa que permet l'ús de comodins ("+","#"), fet que proporciona una gran fexibilitat, però també pot ser explotat maliciosament si no es controla adequadament.

Malgrat aquests avantatges, el protocol MQTT no està pensat amb la seguretat com a objectiu principal, cosa que el fa vulnerable en entorns crítics com l'IoMT si no s'hi afegeixen mecanismes de protecció. Les principals limitacions de seguretat inclouen:

- **El broker com a punt crític:** El broker MQTT és un únic punt de fallada. Si és compromès o queda saturat, tota la infraestructura de comunicació es veu afectada.

- **Flooding i sobrecàrrega:** Un ús malitencionat del QoS i grans volums de dades, poden causar sobrecàrregues en el broker i saturar el sistema.

- **Control d'accés deficient:** En moltes implementacions, si no es configuren polítiques d'ACL (Access Control List), qualsevol client pot publicar o subscriure's a qualsevol tema.

- **Manca d'autenticació forta:** MQTT deneix només un sistema bàsic d'autenticació mitjançant username i password, sense mecanismes d'autenticació mútua ni suport nadiu per a protocols d'identitat moderna (com OAuth 2.0). Si el canal de comunicació no es protegeix amb TLS/SSL, tant les dades com les credencials es transmeten en text pla.

- **Lack of message integrity:** Si no s'utilitza TLS, tampoc hi ha garanties que els missatges no hagin estat modificats durant el trànsit.

Donada la seva extensió en entorns IoT i les seves característiques adaptades a dispositius amb recursos limitats, MQTT s'ha triat com a protocol principal per a la simulació de trànsit en aquest treball. El seu ús permet generar escenaris tant de comunicació legítima com maliciosa, en els quals es poden observar comportaments anòmals mitjançant eines d'anàlisi i detecció. Això facilita la creació de datasets realistes per a l'entrenament d'IDS basats en IA.

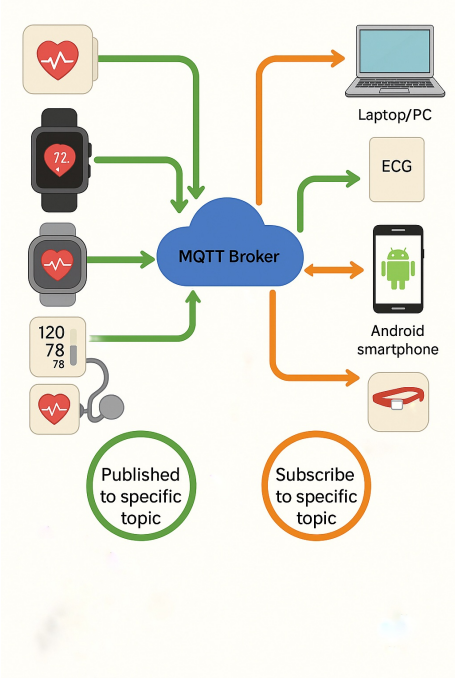

Figura 2.1: *Protocol MQTT en un entorn IoMT. Imatge extreta de [10] i adaptada amb intel·ligència artificial.*

2.4 Altres protocols en l'entorn IoMT

Pel que fa a altres protocols, tot i que MQTT és el protocol principal emprat en aquest treball, també es considera l'ús del protocol Constrained Application Protocol (CoAP) com a alternativa o complement en la generació de trànsit. CoAP és un protocol pensat específicament per a dispositius amb recursos limitats en xarxes IoT. Funciona sobre UDP, cosa que li proporciona una latència molt baixa i un comportament lleuger, tot i que això també comporta certes limitacions pel que fa a la fiabilitat de la transmissió.

CoAP segueix un model client-servidor similar a HTTP però optimitzat per a entorns embeguts. Utilitza mètodes com GET, POST, PUT i DELETE, i permet observar recursos mitjançant un sistema d'actualitzacions automàtiques (observe). A diferència de MQTT, que és orientat a un model (publish/subscribe), CoAP és més adequat per a interaccions puntuals o consulta de recursos puntuals. En aquest treball, l'ús de CoAP es contempla per generar variabilitat en els escenaris de comunicació i per comparar comportaments de trànsit entre protocols amb estructures diferents. Això pot enriquir el dataset resultant i millorar la capacitat de generalització del sistema d'IA per a la detecció d'intrusions.

En l'entorn mèdic, també són utilitzats altres protocols d'aplicació com HTTP/HTTPS o bé Extensible Messaging and Presence Protocol (XMPP) Pel que fa a protocols de capa física, també es fa servir Bluetooth Low Energy (BLE), Near Filed Communication (NFC) o bé dades cel·lulars com NB-IoT que no seran usats en aquest treball.

CAPÍTOL 3

Metodologia / desenvolupament del projecte

En aquest capítol es detallarà la metodologia emprada en la realització del treball. Té com a objectiu oferir un compte detallat de les aproximacions i tècniques utilitzades, assegurant la replicabilitat i el rigor acadèmic. No només cobrirà els mètodes de recerca i tècniques de mesurament emprats, sinó que també aprofundirà en les especificitats del desenvolupament de programari i maquinari. Tant si el projecte implica anàlisi qualitativa, mesuraments quantitatius, modelatge computacional com prototipatge físic, aquest capítol hauria d'elucidar com contribueix cada component als objectius generals.

A més de descriure els mètodes en si mateixos, el capítol també proporcionarà justificacions per què es van escollir mètodes particulars enfront d'altres. Per exemple, podria explicar la tria d'un llenguatge de programació específic, prova estadística o configuració experimental. El capítol també abordarà les limitacions de la metodologia i com aquestes s'han mitigat o tingut en compte. Els lectors haurien de sortir amb una comprensió clara de com s'ha dut a terme el desenvolupament del projecte, per què s'han escollit determinades opcions i com aquests mètodes serveixen per complir els objectius establerts inicialment.

3.1 Escenari utilitzat

L'escenari presentat en aquest treball està inspirat en el que s'utilitza en el Treball: *"«MIoTTA-UPC: Testbed MIoT Configurable para la Evaluacion de Algoritmos de Detección de Ciberataques Basados en Inteligencia Artificial» "* [21] . L'objectiu principal és poder generar un banc de dades que contingui paquets benignes i maliciosos de diferents atcas per a entrenar un sistema de detecció d'intrusions (IDS) basat en intel·ligència artificial que classifiqui si aquest tràfic és benigne o maliciós. La programació, desplegament i entrenament d'aquest IDS no formen part d'aquest treball. Per a generar aquest testbed és necessari simular atacs coneguts per a una infraestructura com utilitzada de manera realista i amb un escenari que reprodueixi adequadament unes condicions reals.

3.2 Topologia general

Per a la generació i captura del tràfic de xarxa en un IoMT (Internet of Medical Things), s'ha dissenyat i desplegat un escenari experimental que simula una habitació d'hospital interconnectada amb un servidor central i altres dispositius de suport clínic. Aquest entorn busca reproduir amb fidelitat un ecosistema típic d'atenció sanitària digitalitzada, incloent-hi sensors mèdics, passarel·les de comunicació, equips de monitoratge i possibles actors maliciosos.

L'escenari està basat en una xarxa LAN Wi-Fi irradiada amb un Access Point, en la qual s'hi connecten tots els dispositius, encara que també pot contenir trams Ethernet si és necessari. Entre els dispositius utilitzats es considera:

- **Clients MQTT:** Aquests clients són sensors com ara oxímetres, monitors de ritme cardíac, bombes d'infusió, sensors de glucosa, tensiòmetres o altres sensors utilitzats en l'entorn mèdic. Aquests clients poden actuar tant transmetent informació a altres dispositius com esperant rebre'n o ambdues a la vegada. Una part d'aquests dispositius, s'implementarà de forma simulada i injectada a la xarxa a través d'un node comú, ja que es tracta de dispositius amb gran cost econòmic. En aquest treball s'utilitzarà Docker com és explicat en apartats posteriors.

- **Broker MQTT:** Connectat a la xarxa, es disposa d'un servidor o broker MQTT amb el el qual s'hi connecten sigui per enviar o rebre informació tots els clients de la xarxa. Actua com un organisme central de la informació i un punt crític de la xarxa.

- **Atacant:** Dintre aquesta xarxa Wi-Fi, se suposa que es connecta un dispositiu atacant, el qual realitza diversos atacs cap als altres components de la xarxa.

- **Monitor de trànsit:** S'hi connecta un monitor que captura tot el trànsit dins la xarxa visible des de la seva posició. Aquest actua de forma passiva escoltant tot el trànsit que circula i recopilant tota la informació possible per tal generar el *testbed*, que és el principal objectiu del treball.

- **Ordinadors I servidors:** Se suposa que en aquesta xarxa hi poden haver connectats altres dispositius que no són especialment utilitzats en l'entorn IoMT com per exemple altres servidors hospitalaris o ordinadors.

Dintre aquest escenari, el meu treball se centra en l'apartat de la generació de trànsit simulat així com en l'elaboració d'atacs des de la perspectiva de desplegar clients simulats i fer-los interactuar amb la xarxa real. L'objectiu principal del treball no ha estat la implementació d'aquesta xarxa.

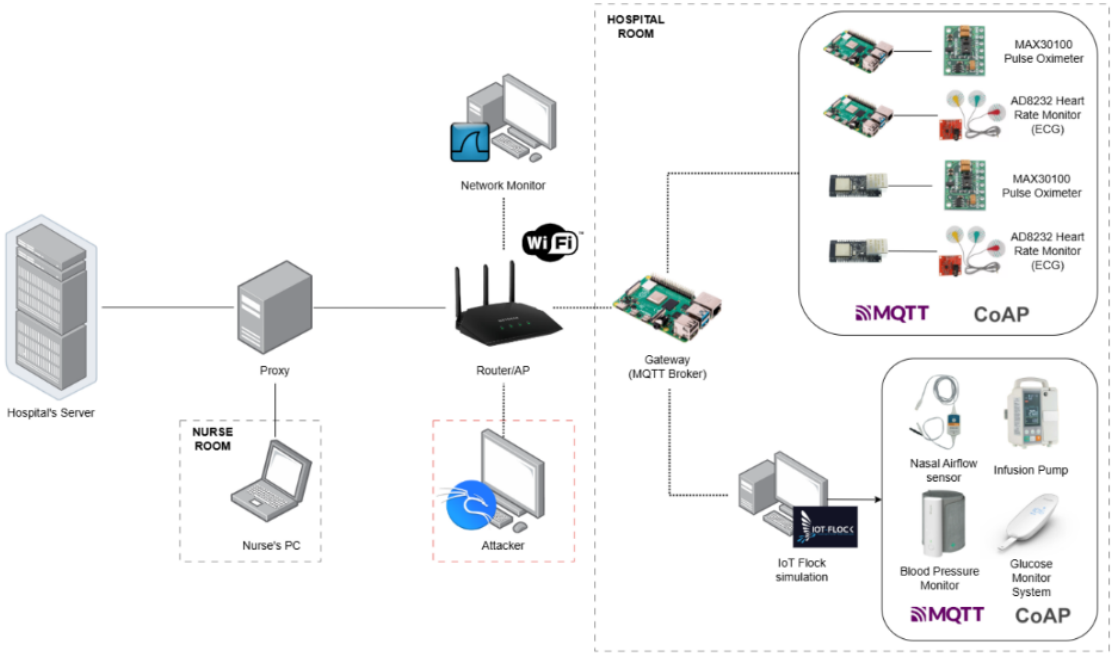

Figura 3.1: *Esquema de l'arquitectura utilitzada. Imatge extreta del treball [21].*

3.3 Ús del protocol MQTT

Primer de tot, l'elecció del protocol MQTT (Message Queuing Telemetry Transport) està fonamentada en el fet que és un dels protocols més utilitzats en l'àmbit IoT en general i en entorns IoMT en específic.

És un dels protocols amb més rellevància i eficiència per a dispositius amb recursos limitats, com és el cas d'aquest treball, on l'apartat de les comunicacions no és la seva funció principal. Consta d'una arquitectura *publisher – subscriber* centralitzada en un únic servidor, la qual cosa fa que tota la informació estigui centralitzada en un sol dispositiu, aquest fet el fa vulnerable i, per tant, cal prendre les mesures de seguretat adequades. A més a més, és un protocol altament configurable, ja que s'hi poden configurar mesures de seguretat com ara limitacions de trànsit, Access Lists (ACL) o encriptat TLS.

Dintre aquest projecte del grup ISG-UPC també es contempla el protocol CoAP, més enfocat a una arquitectura client -servidor semblant a protocols HTTP, però en aquest treball no serà utilitzat.

Mosquitto és una de les implementacions més conegudes del protocol MQTT. Es tracta d'un broker MQTT lleuger, de codi obert i altament configurable i compatible amb les especificacions MQTT 3.1 i 5.0 que permet la configuració de les funcionalitats bàsiques del protocol com definir tòpics, limitacions en l'ús de recursos, ACLs I encriptat TLS. També permet desplegar clients MQTT mitjançant mosquitto-clients i poder realitzar el procés de subscriure's a un tòpic en un broker concret (mosquitto-sub) on publicar

missatges en aquest tòpic (mosquitto-pub). Adicionalment, disposa de les configuracions bàsiques de MQTT com els paràmetres de QoS, retain o presistance. [11]

En l'entorn acadèmic, la seva simplicitat fa que sigui una excel·lent opció per a desplegar laboratoris d'IoMT.

3.4 Ús de Docker per al desplegament de dispositius

Per al desplegament dels dispositius simulats (clients) i del servidor MQTT (broker), s'ha optat per l'ús de contenidors Docker en lloc de màquines virtuals (VMs). Aquesta decisió s'ha pres tenint en compte diversos criteris tècnics, pràctics i de rendiment, que fan que Docker sigui una opció més adequada per als objectius del projecte.

Docker et permet desplegar contenidors seguint una imatge comuna i configurable. D'aquesta manera, podem desplegar els clients simulats o el servidor en qualsevol entorn i sistema que compleixi uns requisits mínims de hardware I software. També permet mantenir una eficiència de recursos òptima, ja que utilitza el propi kernel del sistema hipervisor.

Alhora, és un sistema aïllat del sistema operatiu principal, per això, podem executar proves de penetració sense veure compromesa realment la seguretat dels nostres equips i amb una gran facilitat de reproduir aquest atac diverses vegades sense haver de configurar novament tot el dispositiu vulnerat, ja que aquests contenidors són fàcilment renovables per còpies idèntiques prèvies a l'atac.

A través del seu orquestrador Docker Compose, podem realitzar desplegaments múltiples de dispositius. Amb aquesta eina, podem desplegar en un sol dispositiu físic una gran quantitat de dispositius simulats que comparteixin unes característiques comunes entre ells.

Dintre dels motius pels quals s'ha escollit aquesta tecnologia, està l'ús de volums, els quals et permeten compartir espai en memòria entre el dispositiu hipervisor i els contenidors. Aquesta funcionalitat ens permet agilitzar la transferència d'arxius entre contenidors, com ara fitxers de configuració, scripts per executar tasques determinades o atacs coordinats (en el cas de contenidors desplegats per l'atacant).

També he utilitzat l'arquitectura de xarxa de Docker Compose per a poder crear infraestructures de xarxa simulades senceres, mantenint una lògica i rigorositat en les adreces de cada contenidor, d'aquesta manera, per a alguns atacs m'és possible simular una arquitectura com ara una gran quantitat de contenidors connectats a un switch o bé com un seguit de serveis del host per fer una arquitectura de microserveis dintre un mateix dispositiu.

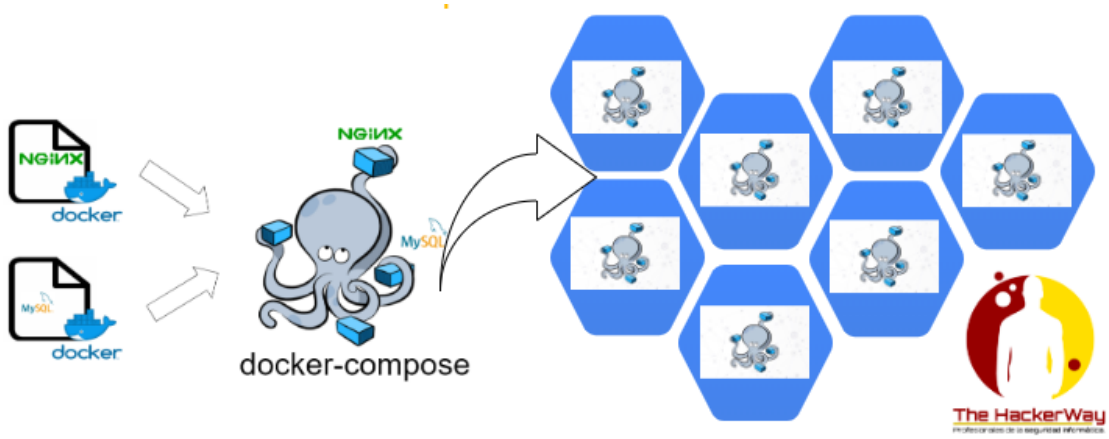

Figura 3.2: *Esquema de la generació de contenidors amb Docker Compose. Imatge extreta de The Hacker Way [21].*

3.5 Eines utilitzades per a la generació d'atacs

Per a la realització d'aquest treball s'han emprat diverses eines de codi obert, escollides pel seu suport ampliat en entorns de xarxa, la seva flexibilitat i la possibilitat d'automatitzar proves i captures de trànsit en entorns simulats. Algunes d'aquestes eines tenen un enfocament general i són àmpliament utilitzades en proves de penetració en xarxes IP tradicionals, mentre que d'altres presenten característiques específiques que les fan especialment adequades per a entorns IoT o IoMT.

Totes aquestes eines han estat utilitzades en un entorn de Kali Linux, que és l'entorn base per a la realització d'aquest treball.

Kali Linux és una distribució basada en Debian orientada específicament a seguretat informàtica i proves de penetració (pentesting). Mantinguda per l'equip d'Offensive Security (OFSEC), Kali proporciona un entorn completament equipat amb centenars d'eines preinstal·lades per a l'auditoria de xarxes, anàlisi de vulnerabilitats, enginyeria inversa, sniffing, spoofing, explotació i forense digital. [19]

Aquesta distribució té certes avantatges respecte a altres sistemes operatius, les mes destacables són:

- **Gran nombre d'eines incloses:** Kali inclou eines com Nmap, Masscan, Wireshark, tcpdump, Scapy, aprscan i moltes més, facilitant la realització de proves diverses sense necessitat d'instal·lació addicional.
- **Entorn controlat i configurable:** Kali pot executar-se de manera virtualitzada (en aquest treball s'ha utilitzat en contenidors Docker), fet que permet recrear escenaris controlats i aïllats per a la simulació d'atacs sense posar en risc cap sistema real.
- **Actualitzacions contínues i suport actiu de la comunitat:** Es tracta d'una

distribució mantinguda amb freqüència, compatible amb la majoria d'arquitectures, i àmpliament utilitzada tant en àmbits acadèmics com professionals.

- **Automatització i scripting:** El seu entorn Unix-like facilita l'ús d'scripts en bash o python per automatitzar atacs, recollir trànsit o llançar seqüències repetitives d'accions

A continuació es descriuen les principals fetes servir en aquest projecte:

3.5.1 Nmap

Nmap (Network Mapper) és una eina de network reconnaissance i auditoria de xarxes molt utilitzada en l'àmbit del pentesting. Permet identificar dispositius connectats a una xarxa, descobrir serveis oberts, detectar sistemes operatius i obtenir informació detallada mitjançant scripts del motor NSE (Nmap Scripting Engine). Nmap pot utilitzar diferents tècniques d'escaneig (TCP SYN, UDP, ping sweep, etc.), i també permet la detecció de versions de serveis, cosa que facilita la identificació de vulnerabilitats específiques en dispositius o serveis actius. És una eina molt completa per a la fase de reconeixement, però pot resultar relativament lenta quan s'aplica a xarxes amb un gran nombre d'hosts o rangs amplis d'adreces IP.

Masscan, en canvi, és una eina especialitzada en escaneig de ports oberts a gran escala i amb una velocitat molt superior a la de Nmap. La seva arquitectura li permet enviar milions de paquets per segon, fet que la fa ideal per a fer un descobriment ràpid d'hosts actius en xarxes grans. No proporciona tants detalls com Nmap (com versions de serveis o sistema operatiu), però és extremadament útil com a pas previ per detectar ràpidament quins dispositius responen en determinats ports.

En el context d'aquest treball, Masscan ha resultat especialment útil per a detectar dispositius IoT actius dins segments de xarxa amb centenars d'IPs possibles, abans d'aplicar anàlisis més profundes amb Nmap. Així, s'ha pogut optimitzar el temps d'escaneig i enfocar els esforços de reconeixement detallat només sobre aquells nodes que realment presentaven algun servei obert.

3.5.2 TCPDump

TCPDump és una eina de línia de comandes per a la captura i anàlisi de paquets a nivell de xarxa. Es tracta d'una eina fonamental en entorns de recerca i pentesting, ja que permet registrar amb precisió tot el trànsit que circula per una interfície de xarxa en temps real. [6]

TCPDump permet capturar trànsit benigne i maliciós entre dispositius de la xarxa i crear arxius de captura amb extensió pcap. És una eina similar a Wireshark, però més lleugera, utilitzada a través de terminal i amb capacitat de ser feta servir en automatitzacions.

3.5.3 ArpSpoof i Bettercap

ARPSpoof és una eina clàssica inclosa dins la suite dsniff, utilitzada per dur a terme atacs de tipus ARP spoofing o ARP poisoning. Aquest tipus d'atac consisteix a enviar respostes ARP falses dins una xarxa local per tal d'enganyar els dispositius i fer-los creure que l'atacant és el un altre dispositiu de la xarxa. Això permet interceptar el trànsit entre dos nodes, amb la finalitat de capturar dades sensibles o manipular el contingut dels paquets. ARPSpoof és una eina senzilla i directa, útil per entendre els fonaments d'aquest tipus d'atacs.

BetterCAP, per la seva banda, és una eina molt més avançada i modular, dins d'ella s'inclouen funcionalitats per a realitzar atacs de tipus ARP spoofing més elaborats i personalitzables que amb ArpSpoof.

3.5.4 Mitmproxy

MITMProxy és una eina de tipus proxy intermediari que permet interceptar, analitzar, modificar i registrar el trànsit entre clients i servidors de manera interactiva. A diferència d'altres eines centrades únicament en la captura passiva, MITMProxy ofereix una interfície potent per veure i modificar les peticions i respostes en temps real.

Per a la realització d'aquest treball MITMProxy ha estat especialment útil perquè permet utilitzar scripts de python per a modificar el trànsit de forma dinàmica, així com per a automatitzar atacs de tipus MITM. [1]

3.5.5 Mqtt Malaria i MQTTSA

MQTTSA (MQTT Security Assistant) és una eina especifica per a la generació de trànsit MQTT permetent realitzar atacs de denegació de servei (DoS) personalitzables com Low-Rate DoS explicat a [16]. També permet altres atacs de força bruta i la generació de reports en PDF. [20]

També s'ha utilitzat MQTT Malaria, una eina enfocada en atacs DoS implementada en Python que ens permet un ús més simple i directe que MQTTSA. [9]

3.5.6 MQTT-PWN

MQTT-PWN és una eina específica per l'auditoria de seguretat del protocol MQTT. És especialment útil en diversos atacs que engloben tècniques de força bruta. [5]

L'eina MQTT-PWN permet realitzar una sèrie de proves automatitzades contra brokers MQTT, com ara la detecció de tòpics disponibles, la publicació i subscripció no autoritzada mitjançant identificació, la identificació d'usuaris i contrasenyes per defecte, i la comprovació de si el canal de comunicació està obert sense cap tipus d'autenticació ni xifratge. En resum, és una eina molt útil per atacs basats en força bruta.

3.5.7 Zeek

Zeek (anteriorment conegut com Bro) és una plataforma d'anàlisi de trànsit de xarxa en profunditat (NSM), que també pot ser utilitzada com a IDS. Està orientada a la generació de registres estructurats a partir de captures de paquets (.pcap). [3]

Zeek genera automàticament fitxers de registre específics com conn.log, dns.log, mqtt-subscriber.log, wired.log entre altres. Aquests fitxers inclouen informació útil com connexions establertes, ports i IPs implicades, consultes DNS, sessions, ús de TLS, autenticacions sospitoses, i molt més.

Si bé l'anàlisi de trànsit amb Zeek no és l'objectiu principal d'aquest treball, s'ha utilitzat per a poder entendre millor el funcionament dels atacs i evaluar la seva efectivitat.

CAPÍTOL 4

Desenvolupament d'un entorn IoMT simulat

Per al desplegament d'un entorn IoMT simulat seguint l'arquitectura explicada a **??** he elaborat un entorn a través de contenidors Docker i l'orquestrador per defecte Docker Compose.

Per a la realització dels clients, que representen el trànsit benigne, he utilitzat una imatge d'Ubuntu, aquesta ha estat modificada (i renombrada com a mqttclient) on mitjançant el gestor de paquets APT s'ha instal·lat mosquitto-clients 2.0.20 ref imatge. Amb aquesta aplicació podem fer actuar aquest contenidor d'Ubuntu com un client MQTT i tenir les seves funcions principals com subscriure's i publicar a un tòpic d'un broker concret i utilitzar totes les funcionalitats descrites. ref mosquitto

També he instal·lat Python 3.13.2 i Paho-mqtt 2.0.0 [12] per a poder generar paquets de forma personalitzada. Amb aquesta llibreria de Python podem modificar aspectes molt més concrets de les nostres connexions MQTT i paquets, canviant els valors de les dades o la freqüència d'enviament de les publicacions. Gràcies a aquesta eina, al ser una llibreria de Python, podem córrer els clients de manera automatitzada utilitzant les eines pròpies del llenguatge.

També he instal·lat les eines net-tools i iputils per tal de poder monitoritzar l'estat dels contenidors i fer comprovacions de connectivitat.

Pel que fa al Broker, he utilitzat l'imatge oficial de mosquitto anomenada eclipse-mosquitto (versió 2.0.21) [14]. Aquesta permet l'ús del contenidor com a broker MQTT en les seves versions 3.1 i 5.1.1 amb totes les funcionalitats de les quals disposa la versió local **??**.

Respecte a configuració de Docker, he mapejat els ports 1883 i 8883 perquè en establir una connexió TCP a un d'aquests ports de l'hipervisor et dirigeixi al mateix port d'aquest contenidor en concret. També he generat 3 volums compartits amb l'hipervisor:

- **Config:** on s'ubica el fitxer de configuracions mosquitto.conf
- **Data:** on opcionalment s'emmagatzemen les dades rebudes en format txt o dintre una base de dades

- **Log:** on es guarden els logs dels errors ocasionats durant el seu funcionament en fitxers .log

Pel que fa al monitor de trànsit, partint de l'image base Ubuntu (renombrada com a sniffer), s'ha instal·lat tcpdump 4.99.5 i wireshark 4.4.3. Com ha estat explicat a **??**, la seva funcionalitat és emmagatzemar el trànsit per tal de generar el dataset, que aquest és enregistrat amb tcpdump, però, el mateix contenidor també ha estat utilitzat per a l'elaboració i l'estudi de l'impacte dels diferents atacs, és per això que s'utilitza Wireshark, que perquè pugui ser visualitzat s'ha fet servir el socket de X11 de l'hipervisor i així aquest contenidor pugui fer ús de GUI.

Per l'atacant, parteix de la imatge kalilinux/kali-last-release [15] (renombrada com a atacker) amb l'instal·lació dels paquets kali-linux-headless i que conté les eines més utilitzades de kali linux i depenent de l'atac s'ha instal·lat altres eines de pentesting.

Tots aquests contenidors, s'han integrat mitjançant un fitxer .yaml de Docker-Compose que ens permet definir volums i fer ús de configuracions de xarxa personalitzades. El nombre de clients utilitzats és variable, però tots disposen d'un sol volum vinculat a l'hipervisor.

S'ha creat una xarxa personalitzada (bridge) on es disposen tots els contenidors i tenen connectivitat entre ells com si es tractés d'una xarxa LAN privada. Per utilitzar un model híbrd entre dispositius simulats i dispositius reals (o bé dispositius simulats en hosts diferents), s'ha utilitzat la configuració MACVLAN que permet connectar els contenidors a la xarxa física amb adreces IP i MAC diferent (cal evitar conflictes amb altres adreces de la xarxa física), amb un efecte similar al que tindríem connectant un switch a la xarxa amb tots els seus contenidors. Amb aquesta configuració m'he coordinat amb altres membres del grup ISG-UPC per a poder integrar el meu treball al projecte. Finalment, en diversos atacs, he configurat l'entorn amb el mode IPVLAN (L2) on cada contenidor té una adreça IP diferent. El host realitza NAT col·locant la seva adreça MAC i responent a peticions ARP. Això és mès compatible en xarxes WiFi WPA2.

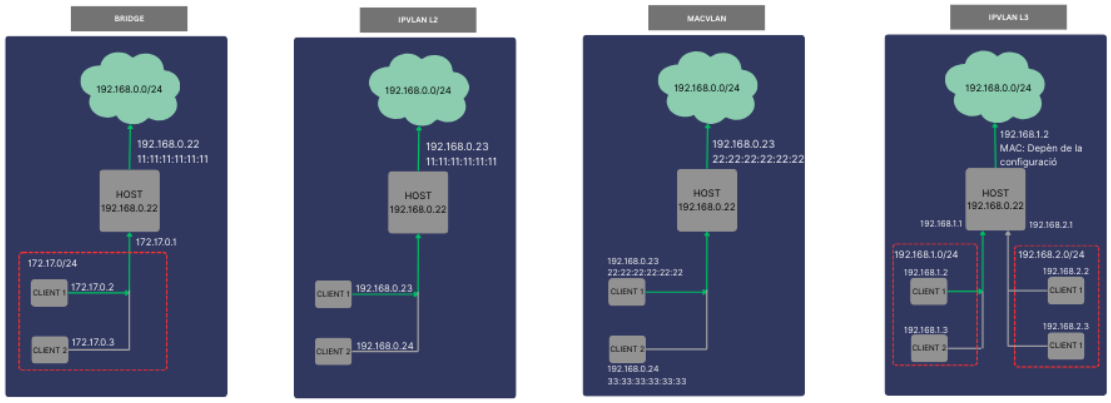

Figura 4.1: *La figura mostra el comportament dels diferents drivers de xarxa de Docker (excloent el mode host on directament actua en nom de l'hipervisor).*

CAPÍTOL 5

Elaboració d'atacs per a la generació de trànsit maliciós

En aquest capítol es presenten els atacs que s'han implementat per a generar trànsit maliciós en l'entorn IoMT descrit a **??**. Aquests atacs tenen com a objectiu comprometre la seguretat dels dispositius connectats o del broker intentant ser fidels a la realitat d'un escenari real.

S'han utilitzat les eines descrites a **??** així com diversos blocs de codi elaborat en Python i funcionalitats pròpies de Linux.

Per tal de generar dades que poden ser fetes servir per a l'entrenament de models de detecció d'intrusions basats en *machine learning*.

5.1 Atacs de descobriment de dispositius

El primer atac que se sol dur a terme en un entorn IoT és el descobriment de dispositius. Aquest se centra en identificar els dispositius connectats a la xarxa i obtenir informació sobre ells, com ara adreces IP, ports oberts i serveis disponibles. Això permet als atacants conèixer la topologia de la xarxa i identificar possibles vulnerabilitats.

Primer de tot, s'ha implementat un atac per aconseguir descobrir l'adreça IP del broker MQTT, ja que és l'element central de la comunicació entre els dispositius IoMT. Per a això, s'ha utilitzat l'eina *nmap* **??** per escanejar la xarxa sencera. Mitjançant Nmap, s'ha aplicat la detecció de ports oberts 1883 i 8883 (en el cas d'ús de TLS), ja que el broker MQTT, per rebre connexions dels clients ha de tenir aquests ports oberts.

També, una estratègia interessant és la detecció de serveis mitjançant l'opció -sV en Namp. Això ens permet descobrir si en el port 1883 o 8883 escanejat, s'està executant un servei de MQTT com per exemple Mosqitto 3.1.1.

Una execució utilitzada és:

```
nmap -sV -p 1883,8883 192.168.0.0/24
```

Aquest atac realitzarà una gran quantitat de peticions ARP a totes les IP del rang especificat i, en cas que contestin intentarà iniciar una connexió TCP als ports especificats (1883 i 8883), recopilant la informació de les respostes. Seguidament, mitjançant scripts propis de Nmap amb diverses peticions TCP intentarà esbrinar quins serveis s'estan executant.

capt wireshark

Però aquesta execució no és del tot realista, ja que és fàcilment detectable. Per això per a poder generar trànsit maliciós de forma més realista, s'han d'utilitzar estratègies menys sorolloses, és a dir on es generi menys trànsit i sigui menys detectable.

Una opció és utilitzar inicialment un escaneig general arp o icmp més lent per tal de recopilar les IPs actives, i llavors fer un escaneig més específic a les IPs que han respost. Així, es redueix el trànsit generat i es fa més difícil la detecció de l'atac.

Per a fer el primer escaneig més silenciós es pot utilitzar l'opcio -T per ajustar la velocitat (un valor -T2 per una xarxa /24 pot trigar uns 10 minuts). També podem afegir l'opció -n per evitar la resolució de noms DNS, ja que al ser una xarxa local no és necessari.

Llavors, per a fer el segon escaneig més específic, es pot utilitzar l'opció -Pn per evitar el ping (ja que ja sabem que el dispositiu està actiu gràcies al primer escaneig). També es pot afegir -sS per evitar completar el handshake TCP i així ser menys detectable. Ara podem afegir els ports específics i la detecció de serveis. Un exemple seria:

```
nmap -sn -n -T2 192.168.0.0/24   #primer escaneig.
nmap -Pn -sV -sS -p 1883,8883 IPs-actives.txt  #segon escaneig amb la
llista d'IPs actives obtinguda del primer escaneig.
```

capt wireshark

Per a fer l'esaneig més ràpid, també he utilitzat masscan per a la detecció del broker, però no és tant eficient i és menys sigilós.

```
masscan 192.168.0.0/24 -p1883,8883 --rate 1000 -oG masscan.txt
```

capt wireshark?

Per a la detecció dels clients IoMT, amb el primer escaneig ja veiem els dispositius actius a la xarxa, però no podem saber si realment són dispositius IoMT, per això podem utilitzar la detecció de *fingerprint* dels dispositius, referit a extreure tota la informació possible que descriu i diferència un dispositiu.

El fingerprint es pot aconseguir mitjançant l'opció *-o* de Nmap que detecta el sistema operatiu o també scripts NSE com *"banner"* o *"fingerprint-settings"*, encara que, són mèto-

des molt intrusius i, per tant, molt detectables. També es pot optar per a utilitzar datasets públics d'adreces MAC conegudes de fabricants com ara Philips Health, Siemens Healthcare o també Raspberry Pi i altres marques dispositius embeguts que siguin usualment utilitzats com a clients IoMT.

Registry	Assignment	Organization Name
MA-S	70B3D5B91	Cardinal Health
MA-S	8C1F6448D	Health Care Originals, Inc.
MA-S	8C1F64C79	Hills Health Solutions
MA-S	8C1F6430C	Hills Health Solutions
MA-S	001BC50B0	Miracle Healthcare, Inc.
MA-S	70B3D5FC4	PHYZHON Health Inc
MA-S	8C1F64B64	Sensus Healthcare
MA-S	8C1F64804	Siemens Healthcare Diagnostics
MA-S	8C1F6454A	Sound Health Systems

Taula 5.1: *Petit extret del fitxer "MAC Address Block Large (MA-S)"de [13] filtrat per a visualitzar només fabricants IoMT on es veuen parelles IniciMAC - Fabricant*

5.2 Atacs de descobriment d'informació del broker MQTT

En aquest apartat es descriuen diferents atacs que permeten descobrir diferent informació interessant per a l'atacant explotant vulnerabilitats del broker MQTT. Per a realitzar aquests atacs, s'ha suposat coneguda l'adreça IP i port del broker MQTT així com altres dades que poden ser trobades mitjançant els atacs utilitzats en l'apartat anterior (**??**).

5.2.1 Atacs de descobriment de credencials

Una de les mesures de seguretat més comunes per als brokers MQTT és l'autenticació mitjançant nom d'usuari i contrasenya que queden enregistrats en una ACL i limita per cada tòpic quins usuaris amb la seva respectiva contrasenya poden accedir a cada tòpic. Per tant, un atac comú és intentar descobrir aquestes credencials per accedir als tòpics utilitzats.

Una opció és el sniffing de credencials mitjançant l'ús d'eines com TCPDump (**??**), ja que si no s'utilitza encriptat TLS, les credencials es transmeten en clar.

Però, també es poden utilitzar eines especialitzades com ara MQTT-SA (**??**) que permeten realitzant d'igual manera una tècnica de sniffing, elaboren llistes dels usuaris i credencials

recopilats i les validen contra el broker MQTT. Un exemple d'execució és:

```
mqttsa ...
```

Un altre atac possible és el descobriment de credencials mitjançant força bruta. Aquest procés es basa en comprovar una gran quantitat de combinacions de nom d'usuari i contrasenya per tal d'accedir al broker MQTT i recopilant la seva validesa en un fitxer de sortida.

Per aquest atac, s'ha utilitzat l'eina MQTT-PWN (**??**) que permet mitjançant un fitxer de diccionari, provar totes aquestes combinacions i recopilar els resultats. Un exemple d'execució és:

```
mqttpwn ...
```

Aquest atac genera grans quantitats de paquets MQTT connect en direcció al broker. **capt wireshark**

5.2.2 Subscripció a tòpics d'administració

Una característica dels brokers MQTT, és l'ús de tòpics d'administració anomenats **$SYS** que permeten als clients obtenir informació sensible sobre l'estat del broker i dels clients.

Per a la realització d'aquest atac, s'ha utilitzat un script de NMAP anomenat mqtt-subscribe que recopila tota aquesta informació subscrivint-se a tots els tòpics d'administració possibles. Una execució d'aquest atac per a un broker amb IP 192.168.0.22 és la següent:

```
nmap -Pn --script mqtt-subscribe -p 1883 -oG info_broker.txt
192.168.0.22
```

Amb aquest atac a un broker mosquitto com l'utilitzat en aquest treball, s'obté informació com ara:

- Informació de la versió del broker i configuració general del broker.
- Nom d'usuari, estat de la connexió i keep alive dels clients.
- Nombre màxim de clients simultanis amb els quals pot treballar el broker.
- Mètriques de rendiment: bits enviats per segon, bits rebuts per segon, latència, etc
- Estadístiques de missatges enviats i rebuts.
- Llista de tòpics publicats i subscrits.
- Diversos errors i advertències del broker.

En aquest atac, el kali linux l'atacant genera un gran nombre de paquets MQTT subscribe per a subscrire's a aquests tòpics.

capt wireshark

5.3 Atacs de denegació de servei (DoS)

En aquest apartat es descriuen diferents atacs de denegació de servei (DoS) que poden ser realitzats contra el broker MQTT. Aquests atacs tenen com a objectiu saturar el broker amb peticions, fent que no pugui atendre les peticions legítimes dels clients, d'aquesta manera, aconseguim com el seu nom indica, denegar el servei a tots els clients que intenten connectar-se o enviar dades al broker.

Per a la realiztació d'aquest atac, s'ha suposat conneguda l'adreça IP i port del broker MQTT mitjançant els atacs utilitzats en l'apartat anterior (**??**).

5.3.1 Denegació de servei MQTT Publish

Inicialment, he implementat un atac de denegació de servei en el qual s'envia un gran nombre de missatges MQTT publish a un broker concret desde un client maliciós. He configurat un client MQTT mitjançant kali linux dintre un contenidor Docker com s'explica en **??**.

Amb aquest client, s'envien peticions MQTT Publish de forma contínua mitjançant un script de bash en format bucle infinit. Amb aquest atac, en cas de protecció nula del broker, aquest es satura ja que no té un sistema de preferències per a gestionar les peticions i no pot atendre les peticions legítimes dels clients.

Una actualització d'aquest atac va ser l'eleboració d'un script de Python que, mitjançant la llibreria Paho-MQTT, permet connectar el client al tòpic especificat i enviar un gran nombre de missatges MQTT Publish de forma contínua però amb camps de dades diferents cada vegada. L'ús d'aquesta llibreria en comptes de mosquitto-clients millora la eficiència ja que permet generar trànsit amb mès velocitat, això ho aconsegueix generant un gran nombre de threads diferents, els quals envien simultaniament missatges al broker en un bucle infinit.

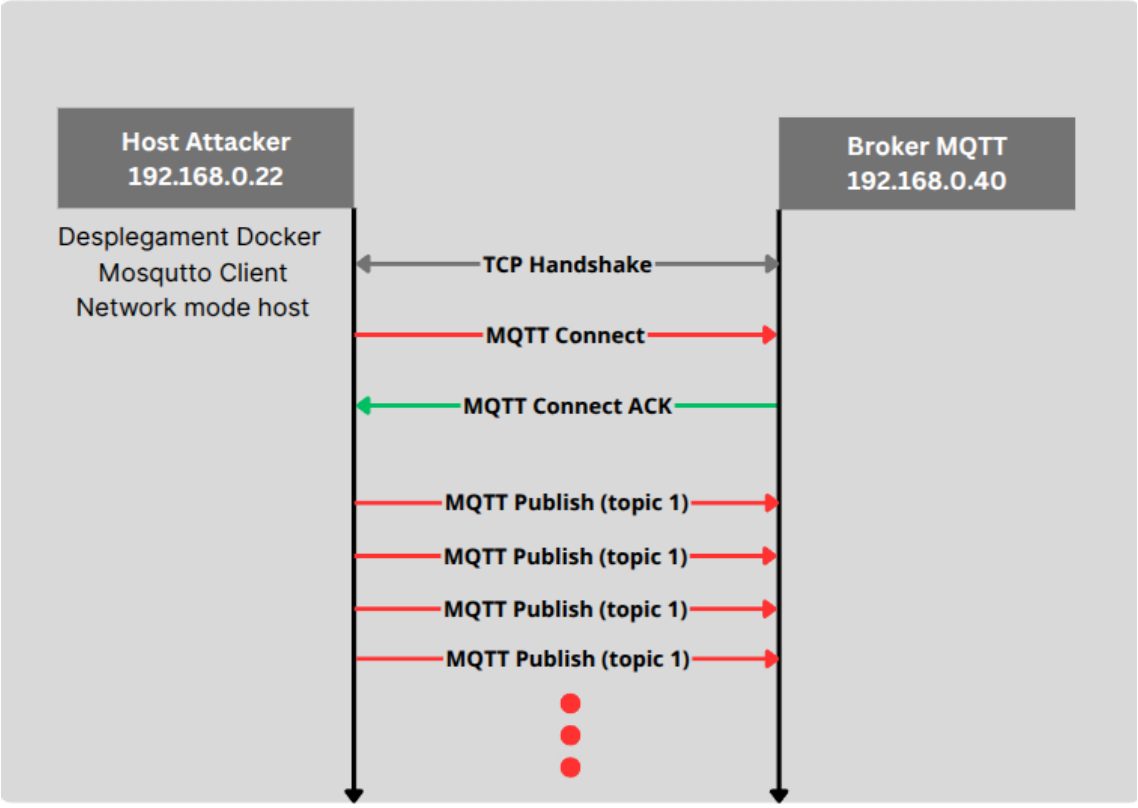

Figura 5.1: *Esquema del trànsit generat per l'atac de denegació de servei MQTT Publish explicat anteriorment.*

5.3.2 Denegació de servei Distribuïda (DDoS)

Per a la realització d'atacs de denegació de servei en MQTT, és important entendre la presistència de les sessións en MQTT, explicada a **??**. En l'atac DoS, si utilitzem sesions no presistents, cada vegada que l'atacant vol publicar un missatge, ha de tornar intercanviar el handshake TCP i el handshake MQTT Connect. Els paquets intercanviats d'aquesta manera son lleugers i no generen una gran quantitat de trànsit, i, per tant, l'atac perd eficiència. En canvi si s'utlitzen sesions presistents, els handshakes TCP i MQTT Connect només s'intercanvien una vegada i després es poden enviar missatges de forma contínua sense necessitat de tornar a establir la connexió.

L'atac proposat inicialment no és del tot realista, ja que la major part dels brokers MQTT, tenen configurats paràmentres per defecte per tal de limitar el nombre de missatges rebuts per segon des d'un client concret, com es pot veure a **??**. Per aquest motiu, vaig modificar l'script per tal que canviés el Client ID dels missatges en cada connexió.

El client ID és un identificador únic per a cada client MQTT que es connecta al broker. Si s'utilitza un Client ID diferent per a cada connexió, el broker no pot aplicar les mateixes restriccions de taxa de missatges, ja que cada connexió es veu com un client diferent. Aquest s'assigna a l'hora d'intercanviar els missatges MQTT Connect, per tant no es pot

fer l'atac amb connexións presistents, com he explicat anteriorment, es perd eficiecia comparat amb l'atac anterior, això es pot veure comparant les quantitats de trànsit generades pels 2 atacs.

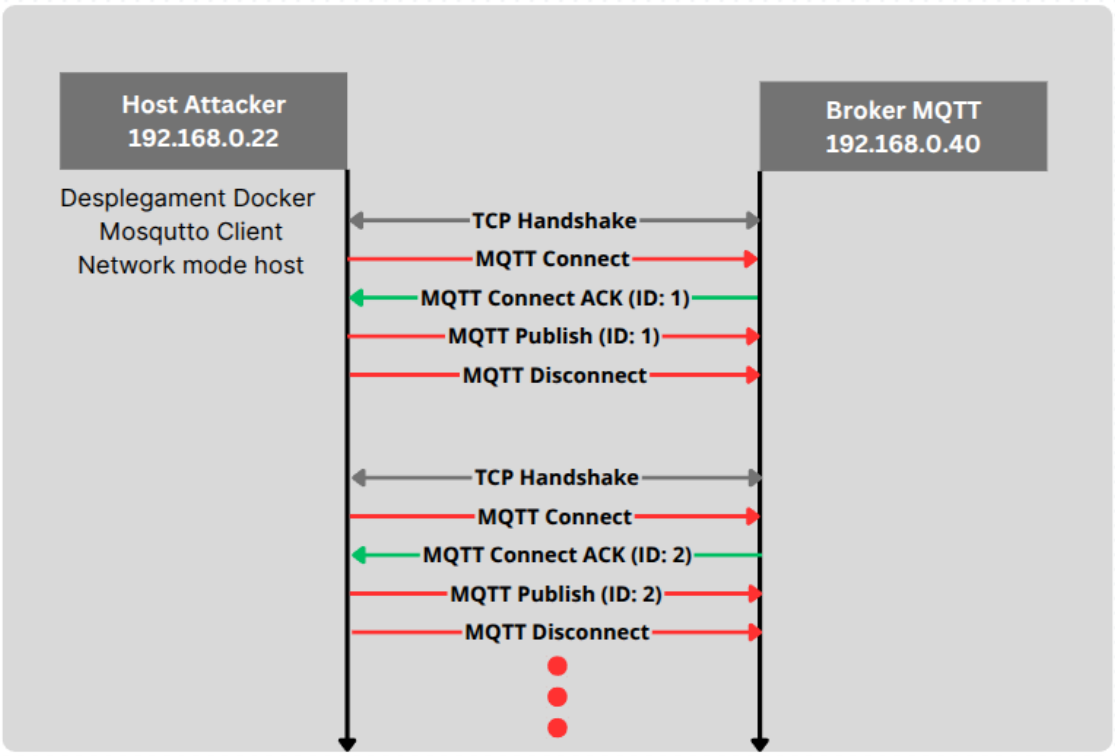

Figura 5.2: *Esquema del trànsit generat per l'atac de denegació de servei MQTT Publish amb modificació del paràmetre Client ID.*

El darrer mètode per a realitzar l'atac de denegació de servei distribuït amb Client ID diferents és mitjançant arquitectures de xarxa personalitzades amb Docker, concretament una arquitecture MACVLAN, on cada contenidor té una adreça IP diferent **??**. En aquesta versió de l'atac DDoS, generem un gran nombre de contenidors Docker i en cadascún despleguem un client MQTT. Aquest clients estableixen una connexió presistent amb el broker (amb Client ID diferents entre ells) i realitzen l'ataca de denegació de servei com els anteriors. Aquesta estratègia té una complexitat mès elevada però permet generar un gran nombre de connexións diferents però mantenint la presistència. D'aquesta manera, es s'aconsegueix un efecte similar a un DDoS amb diversos clients reals però utilitzant eines de virtualització com Docker.

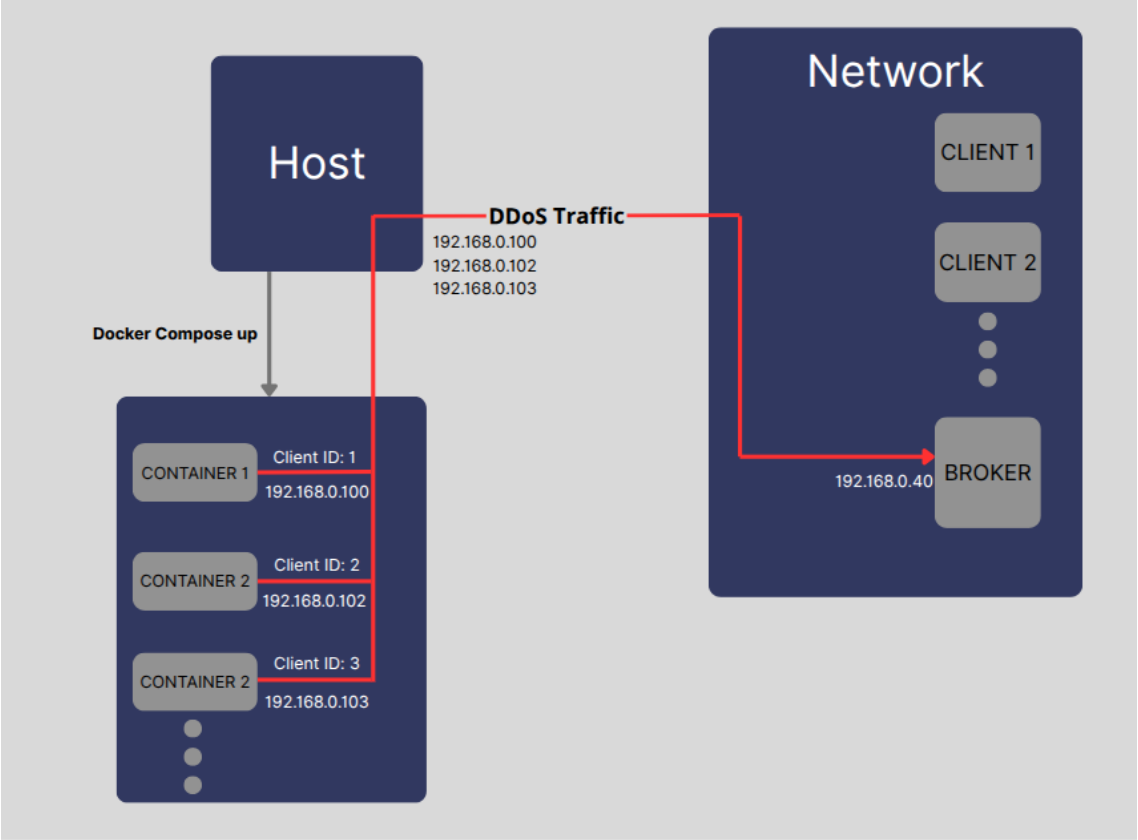

Figura 5.3: *Esquema del trànsit generat per l'atac de denegació de servei Distribuït. El DDoS Traffic es refereix a trànsit generat amb una estructura similar al de la figura* **??**.

Un cop elaborat l'atac, he passat a utilitzar eines ja existents com ara MQTT-PWN (**??**) i MQTT Malaria que ajuden a obtenir una eficiència més elevada al realitzar atacs DoS amb un codi més optimitzat. Aquestes tenen un funcionament similar al script de Python utilitzat anteriorment.

També és molt important escollir el valor de QoS adequat per a l'atac, l'ús de **QoS 0** permet enviar missatges sense esperar confirmació, per tant, permet saturar la xarxa amb més facilitat, però, amb QoS 1, el processat que ha de realitzar el broker per a cada missatge és més elevat i, per tant, millra l'eficiència en quant a saturació del broker. Si s'utilitza QoS 2, aquest processat és encara més elevat i ja comporta una diferència molt notable respecte a QoS 0. Per tant, QoS 2 es la opció més recomananada per a realitzar l'atac de denegació de servei.

5.3.3 Low-Rate DDoS

Tal i com s'explica a l'article [16], els atacs de denegació de servei distribuïts (DDoS) poden ser realitzats amb un trànsit baix, és a dir, enviant un nombre reduït de missatges per segon. Aquest tipus d'atac pot ser més difícil de detectar i pot passar desapercebut per

les mesures de seguretat del broker MQTT. Per a compensar la baixa taxa de missatges, s'utilitza un gran nombre de contenidors diferents, augmentant el nombre de clients.

Per acabar.

5.4 Atacs de suplantació d'identitat

En aquest apartat s'expliquen atacs de suplantació d'identitat que poden ser realitzats contra una xarxa MQTT. Aquests atacs tenen com a objectiu suplantar la identitat d'un client legítim per tal d'enviar missatges al broker MQTT o rebre'n, fent que el broker no pugui distingir entre clients legítims i clients maliciosos. També el fet de modificar informació legítima a través de sistemes de Man In The Middle.

Primer de tot, s'ha estudiat el funcionament d'atacs de ARP Spoofing, per entendre'ls és necessari conèixer el protocol ARP en profunditat.

El Protocol de Resolució d'Adreces (ARP) és un mecanisme fonamental en xarxes IP que permet als dispositius d'una xarxa local associar adreces IP amb adreces MAC corresponents. Quan un dispositiu necessita enviar un paquet a una adreça IP dins de la seva mateixa xarxa local, emet una petició ARP a través de difusió (broadcast), sol·licitant quina adreça MAC està associada a aquella IP. El dispositiu propietari d'aquesta adreça IP respon amb la seva adreça MAC, i aquesta informació queda temporalment registrada a la taula ARP del dispositiu que ha fet la sol·licitud. Tot i la seva simplicitat i eficiència, ARP no incorpora cap mecanisme d'autenticació, la qual cosa el fa vulnerable a diversos tipus d'atacs, entre els quals destaca l'ARP spoofing.

L'ARP spoofing, també conegut com a ARP poisoning, és una tècnica d'atac que aprofita la manca de verificació en el protocol ARP per introduir entrades falses en les taules ARP dels dispositius de la xarxa. L'objectiu principal és enganyar aquests dispositius perquè associïn una adreça IP legítima (habitualment la del gateway o la d'una víctima específica) amb l'adreça MAC de l'atacant. Això permet a l'atacant interceptar el tràfic que, en condicions normals, es dirigiria directament al gateway o a un altre dispositiu. D'aquesta manera, es crea una situació de tipus Man-in-the-Middle (MitM), en què l'atacant pot monitoritzar, modificar o redirigir el tràfic entre dispositius.

El funcionament bàsic d'un atac ARP spoofing es pot resumir en els passos següents:

- L'atacant envia respostes ARP falsificades a la víctima, fent-se passar pel gateway de la xarxa.
- Simultàniament, envia respostes ARP falsificades al gateway, fent-se passar per la víctima.
- Tant la víctima com el gateway actualitzen les seves taules ARP amb les associacions falses proporcionades per l'atacant.
- A partir d'aquest moment, el tràfic entre ambdós dispositius es redirigeix a través de l'atacant, que pot actuar com a passarel·la transparent (forwarder) o bé manipular

els paquets segons el seu objectiu.

En aquest treball, inicialment s'ha utilitzat l'eina arpspoof del paquet dsniff per realitzar l'atac ARP spoofing. Aquesta eina permet enviar respostes ARP falsificades a la víctima i el broker o el gateway, fent que ambdós dispositius actualitzin les seves taules ARP amb les associacions falses proporcionades per l'atacant tal i com s'ha explicat anteriorment.

Un exemple d'execució utilitzada on client real víctima té l'adreça IP 192.168.0.41 i és:

```
arpspoof -i wlp42s0 192.168.0.41
```

Amb aquesta execució aconseguim que el missatges que envia el broker MQTT al client real es redirigeixin a l'atacan. D'aquesta manera, si el client està subscrit a un tòpic concret, podem fer que l'atacant rebi aquests missatges, com per exemple podrien ser mesures de ritme cardíac o pressió sanguínia que deixen d'arribar a un monitoritzador mèdic.

Però, aquest atac és fàcilment detectable. Per això, cal implementar un atac bidireccional, fent que tant el broker com el client enviin els missatges a l'adreça MAC de l'atacant, generant una situació de MITM. Ho podem aconseguir amb una execució similar a la següent on s'afageix l'adreça IP del broker MQTT 192.168.0.40 amb el paràmetre -r:

```
arpspoof -i wlp42s0 -t 192.168.0.41 -r 192.168.0.40
```

Per al perfeccionament de l'atac APR Spoofing, he utilitzat l'eina better

CAPÍTOL 6

Resultats

Aquest capítol ha d'incloure l'anàlisi de les vostres dades i els resultats obtinguts. A més, incloeu-hi taules, figures i citacions pertinents per donar suport als vostres resultats i interpretacions. Aquí teniu una llista suggerida de temes a tractar:

6.1 Experiments i proves

Descriviu els experiments realitzats per provar el rendiment del vostre projecte. Expliqueu com heu recopilat i processat les dades.

6.2 Visualització de les dades

Creeu representacions visuals dels resultats (per exemple, gràfics de dispersió, diagrames de barres). Interpreteu les visualitzacions i relacioneu-les amb les preguntes de recerca.

6.3 Limitacions

Reconeixeu qualsevol limitació en les dades o l'anàlisi. Expliqueu com aquestes limitacions podrien haver afectat els resultats.

CAPÍTOL 7

Anàlisi de sostenibilitat i implicacions ètiques

Des del curs 2023-24, la normativa de TFG de l'ETSETB demana la inclusió d'un informe de sostenibilitat a la memòria del treball. Aquesta anàlisi consisteix en una valoració dels impactes ambientals, socials i econòmics, i les possibles implicacions ètiques que ha comportat la realització del TFG. En el cas que el TFG plantegi un producte/servei/sistema/edifici/etc., que podria arribar a implementar-se, l'anàlisi també ha de realitzar-se sobre els impactes que tindria la proposta en la seva execució durant les diferents etapes del seu cicle de vida.

A la plataforma ATENEA trobareu un document separat amb les instruccions detallades de què ha de contenir i com cal confeccionar l'informe de sostenibilitat.

IMPORTANT: Noteu que l'antic capítol de «Pressupost del projecte» ara queda integrat en l'anàlisi de sostenibilitat, concretament en les cel·les «Econòmic/-Desenvolupament del TFG» i «Econòmic/Execució del projecte».

CAPÍTOL 8

Conclusions i Línies Futures

8.1 Conclusions

- Resumiu els resultats principals del vostre treball.
- Discutiu el grau d'assoliment en relació amb els objectius marcats a l'inici del treball.
- Destaqueu les contribucions del vostre treball al camp d'estudi.

8.2 Línies Futures

- Identifiqueu àrees per a futures investigacions o desenvolupament basades en el vostre treball.
- Discutiu possibles vies per ampliar o millorar el projecte.
- Considereu les preguntes que han quedat sense resposta i les oportunitats per a futures exploracions.

Bibliografia

El sistema *biblatex* simplifica la gestió de la bibliografia en treballs científics, proporcionant automatització i personalització en el format de les citacions. Això permet a l'autor del document enfocar-se en el contingut sense haver de preocupar-se per l'estil de les referències, estalviant temps i reduint errors.

La base de dades de referències bibliogràfiques és al fitxer «TFG.bib» i és allà on heu d'afegir les vostres referències. Consulteu el manual del `biblatex`, secció «Database Guide», per conèixer els tipus de referències i camps disponibles.

Podeu modificar (o suprimir) aquesta nota editant la macro `\defbibnote` al fitxer «TFG.tex».

---

[1] Thomas Kriechbaumer Aldo Cortesi Maximilian Hils. *mitmproxy: A Free and Open Source Interactive HTTPS Proxy*. 2025. URL: https://docs.mitmproxy.org/stable/ (cons. 31-05-2025).

[2] Héctor Aláiz-Moretón Ángel Luis Muñoz Castañeda José Antonio Aveleira Mata. «Characterization of threats in IoT from an MQTT protocol-oriented dataset». A: *Complex Intelligent Systems* 9.01000 (2023), pàg. 5281 - 5296. DOI: 10.1007/s40747-023-01000-y.

[3] Carlos Cilleruelo. *¿Qué es Zeek?* 2014. URL: https://keepcoding.io/blog/que-es-zeek/ (cons. 31-05-2025).

[4] Newtowk Chuck. *Docker Tutorials - NetworkChuck*. 2020. URL: https://www.youtube.com/watchv=eGz9DSaIeY&list=PLIhvC56v63IJlnU4k60d0oFIrsbXEivQo&index=2 (cons. 09-05-2025).

[5] Moshe Zioni Daniel Abeles. *Welcome to MQTT-PWN!* 2018. URL: https://mqtt-pwn.readthedocs.io/en/latest/ (cons. 31-05-2025).

[6] TCPDump Developers. *TCPDump: Manual and Documentation*. 2023. URL: https://www.tcpdump.org/ (cons. 08-05-2025).

[7] Daniel Echeverri. *Docker Stack: Cómo desplegar servicios de Docker Compose en Docker Swarm*. 2021. URL: https://thehackerway.es/2021/11/22/docker-stack-como-desplegar-servicios-de-docker-compose-en-docker-swarm/ (cons. 21-05-2025).

[8] Alexandria University Elsevier BV on behalf of Faculty of Engineering. *The internet of things healthcare monitoring system based on MQTT protocol*. 2024. URL: https://www.sciencedirect.com/science/article/pii/S1110016823000881 (cons. 08-05-2025).

[9] eTactica. *mqtt-malaria*. 2021. URL: https://github.com/etactica/mqtt-malaria (cons. 31-05-2025).

[10] Mehdi Delrobaei Fatemeh Ghorbani Mohammad Kia. *Evaluating the Possibility of Integrating AugmentedReality and Internet of Things Technologies to HelpPatients with Alzheimer's Disease*. Figura adaptada i modificada amb IA. 2024. URL: https://www.researchgate.net/figure/MQTT-protocol-operation-in-AAL-system_fig1_339906098.

[11] Eclipse Foundation. *Mosquitto MQTT Broker Documentation*. 2024. URL: https://mosquitto.org/documentation/ (cons. 08-05-2025).

[12] Python Software Foundation. *Paho MQTT Python Packaging Survey*. 2025. URL: https://pypi.org/project/paho-mqtt/ (cons. 23-05-2025).

[13] IEEE. *IEEE Registration Authority: Assignments*. 2015. URL: https://regauth.standards.ieee.org/standards-ra-web/pub/view.html#registries (cons. 01-06-2025).

[14] Docker Inc. *Docker Hub: eclipse-mosquitto*. 2025. URL: https://hub.docker.com/_/eclipse-mosquitto (cons. 23-05-2025).

[15] Docker Inc. *Docker Hub: kalilinux/kali-last-release*. 2025. URL: https://hub.docker.com/r/kalilinux/kali-last-release (cons. 23-05-2025).

[16] Qasem Abu Al-Haija Mustafa Al-Fayoumi. «Capturing low-rate DDoS attack based on MQTT protocol in software Defined-IoT environment». A: *Sciencie Direct Journal* 19.100316 (2023), pàg. 10. DOI: 10.1016/j.array.2023.100316.

[17] Nmap Project. *Nmap Reference Guide*. 2024. URL: https://nmap.org/book/man.html#man-description/ (cons. 08-05-2025).

[18] Raphael Ferreira Sajjad Dadkhah Euclides Carlos Pinto Neto. «CICIoMT2024: Attack Vectors in Healthcare devices-A Multi-Protocol Dataset for Assessing IoMT Device Security». A: *Preprints.org* 1.0898 (2024), pàg. 30. DOI: 10.20944/preprints202402.0898.v1.

[19] Offensive Security. *Kali Linux Documentation*. 2024. URL: https://www.kali.org/docs/introduction/what-is-kali-linux/ (cons. 11-05-2025).

[20] Security Trust. *MQTTSA*. 2024. URL: https://github.com/stfbk/mqttsa (cons. 31-05-2025).

[21] Josep Pegueroles Valles. «MIoTTA-UPC: Testbed MIoT Configurable para la Evaluacion de Algoritmos de Detección de Ciberataques Basados en Inteligencia

Artificial». A: *The Bell System Technical Journal* 27.3 (1948), pàg. 379 - 423. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

APÈNDIX A

Un apèndix

Es poden incloure apèndixs al TFG però no és obligatori.