Mestrado em Cibersegurança e Informática Forense

Cybersecurity Detection System for IoT

Luís Miguel Silva Canuto

Leiria, Março de 2020

Mestrado em Cibersegurança e Informática Forense

Cybersecurity Detection System for IoT

Luís Miguel Silva Canuto

Dissertação de Mestrado realizada sob a orientação do Professor Doutor Carlos Rabadão, Professor Coordenador da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria e co-orientação do Professor Doutor Leonel Santos, Professor Adjunto da Escola Superior de Tecnologia e Gestão do Instituto Politécnico de Leiria.

Leiria, Março de 2020

Dedicatória

Aos meus queridos pais.

Agradecimentos

A realização deste projeto proporcionou uma experiência enriquecedora, que foi encarada com muita dedicação e possibilitou um engrandecimento pessoal. Não poderia deixar de agradecer a várias entidades que me ajudaram de alguma forma, seja ela direta ou indireta, na realização deste projeto. Deste modo queria deixar os meus sinceros agradecimentos a todas elas.

A minha primeira palavra de agradecimento é dedicada à minha família que sempre me apoiou e me incentivou durante esta jornada académica.

Das várias pessoas que me ajudaram, gostaria de agradecer em especial aos meus orientadores, Leonel Santos e Carlos Rabadão, por toda a ajuda prestada, desde as conversas aos conselhos dados durante as reuniões, mas principalmente por toda a sua disponibilidade e apoio durante todo o percurso deste projeto.

Gostaria de agradecer igualmente aos meus colegas que participaram na realização do projeto, em especial ao colega Roberto Leal pelo apoio e ajuda durante todo este período, tornando-se numa grande ajuda.

Por último, gostaria de agradecer ao colega João Pedrosa pelas várias horas de diálogo sobre o projeto, ajudando na realização do mesmo.

Nota Prévia

Este projeto tem como objetivo apresentar uma *framework* que possibilita a deteção de intrusões em sistemas *Internet of Things (IoT)* a fim detetar, em tempo útil, intrusões com origem interna ou externa, com principal enfoque nas intrusões que afetem as camadas de rede e aplicação da arquitetura IoT.

A realização deste projeto incluiu o desenvolvimento de trabalhos de um estudante de doutoramento e dois estudantes de mestrado. O trabalho base foi realizado pelo Professor Doutor Leonel Santos e tem o nome de "Sistema de deteção de intrusões para a Internet das Coisas". [1] Neste trabalho foram realizados testes ao funcionamento da *framework* ao nível do funcionamento, desempenho e segurança.

Em sintonia com este projeto, existe um outro, desenvolvido pelo aluno Roberto Leal com o nome *"Cybersecurity Detection System for IoT"*, onde define especificações para o protocolo *Message Queue Telemetry Transport (MQTT)* e, realiza testes de forma a validar a *framework*, tendo em consideração um protocolo baseado em *Transmission Control Protocol (TCP)*.

Deste modo, este projeto é parte integrante dos trabalhos anteriormente abordados, onde é pretendido analisar, definir e avaliar as especificares relativas ao tráfego gerados pelo aplicativos *Constrained Application Protocol (CoAP)* e *Constrained Application Protocol over DTLS (CoAPS)*, tendo em consideração um protocolo baseado em *User Datagram Protocol (UDP)*.

Resumo

A próxima geração da Internet aponta para um cenário onde milhares de milhões de utilizadores terão acesso a objetos, em qualquer momento e em qualquer lugar. Um dos pilares desta nova geração é a Internet das Coisas, ou *IoT*. Este conceito é um paradigma emergente na nossa sociedade, onde é pretendido integrar vários tipos de objetos do mundo físico no mundo digital. Esses objetos estão dispersos pelos vários setores económicos, como o setor da indústria, o setor da saúde, o setor de transportes, entre outros, permitindo moldar uma rede de objetos interligados. Conceptualmente é a possibilidade de conectar o mundo físico com o mundo digital permitindo assim registar dados ligados às nossas ações, utilizando posteriormente essas informações de forma a integrar processos, serviços e aplicações. Deste modo, os vários setores económicos conseguem beneficiar deste paradigma utilizando informação de contexto relevante durante a sua execução. Contudo, o avanço deste advento traz consigo importantes vulnerabilidades sociais e materiais. Se, por um lado, a Internet expõe os seus utilizadores a novas situações de risco e a outras que, embora já existam no mundo físico, potenciam-se no mundo digital, fruto da maior exposição e alcance que as tecnologias proporcionam, por outro lado, devido às gritantes limitações para os dispositivos IoT, estes estão facilmente suscetíveis a falhas de segurança. Estas limitações levam a que a comunicação na maioria das vezes seja efetuada através de uma ligação insegura, comprometendo em muitas ocasiões os dados e os dispositivos em relação à sua confidencialidade, integridade e disponibilidade.

De forma a tentar superar estes desafios, foram introduzidas contra medidas de segurança convencionais, nomeadamente *firewall*, cifragem de dados e comunicações, auditorias e mecanismos de autenticação e autorização. Mais uma vez, devido às limitações destes dispositivos, nem sempre é viável a utilização destas contra medidas pois ou não se enquadram ou não são o suficientemente tolerantes para as caraterísticas dos sistemas IoT. No entanto, uma das contra medidas que pode ser implementada neste tipo de redes são os sistemas de deteção de intrusões, ou *Intrusion Detection System (IDS)*. Para isso, um IDS deve respeitar a arquitetura IoT, isto é, deve ter em consideração a sua escalabilidade, a heterogeneidade dos dispositivos e respetivos meios de comunicação, a privacidade, entre outros.

As soluções existentes de IDS para IoT ainda apresentam muitas limitações, pelo que é perentório melhorar essas lacunas, sejam elas através de novas abordagens, nomeadamente as metodologias de deteção utilizadas, à quantidade de intrusões detetas e em diferentes camadas IoT, à interoperabilidade emergente nestes sistemas, ou simplesmente, melhorando as limitações já documentadas na literatura sobre o tema.

De formar a mitigar estas limitações, este projeto propõe especificações para a *framework* proposta com a finalidade da deteção de intrusões em sistemas IoT, cujo o objetivo principal visa detetar em tempo útil intrusões com origem interna ou externa que afetem todas as camadas da arquitetura IoT. A *framework* proposta visa recolher e agregar as comunicações IoT para, posteriormente, analisá-las de forma a detetar anomalias e intrusões. Relativamente à análise, esta é efetuada através de uma arquitetura baseada num sistema de deteção de intrusão de rede com uma localização híbrida, tendo como base a recolha de dados distribuídos e uma análise centralizada, recorrendo posteriormente a uma metodologia de deteção baseada em especificações de tráfego normal através de fluxos.

Em relação às especificações utilizadas, estas foram criadas com base em tráfego gerado num cenário de testes. Numa fase inicial, apenas foi gerado tráfego normal para a criação dos registos de fluxos de tráfego normais e respetivas especificações. Numa segunda fase, foi gerado tráfego anómalo, isto é, tráfego com ataques e ameaças, de forma a criar registos de fluxos de tráfego anómalos e respetivas especificações. Por último, para os fluxos classificados como tráfego anómalo, foram criadas especificações de forma a ser possível detetar o tipo de ataque, nomeadamente ataques de *Net Scan*, ataques visando URI válidos e inválidos, incluindo os ataque em rajada, utilizando os métodos do protocolo aplicacional *CoAP*.

De modo a validar a *framewok* foi desenvolvido um protótipo que foi posteriormente analisado e avaliado mediante os resultados da implementação de um plano de testes.

Os testes realizados confirmam a validação da *framework* proposta, atingindo valores bastantes satisfatórios.

Posto isto, é seguro afirmar que, sem fortes alicerces de segurança, os ataques e falhas nos sistemas IoT irão superar qualquer um dos seus benefícios.

Palavras-chave: Internet das Coisas; Sistemas de deteção de intrusão; CoAP; Ataques de sistemas; Segurança de sistemas.

Abstract

The next-generation of the Internet points to a scenario where billions of users access objects, anytime, anywhere. One of the pillars of this new generation is the Internet of Things, or *IoT*. This concept is an emerging paradigm in our society, where it is intended to integrate various types of objects from the physical world into the digital world. These objects are dispersed among the various economic sectors, such as the industrial sector, the health sector, the transport sector, among others, allowing to shape a network of interconnected objects. Conceptually, it is the possibility of connecting the physical world with the digital world, thus allowing the recording of data related to our actions, later using this information in order to integrate processes, services and applications. In this way, the various economic sectors are able to benefit from this paradigm using relevant context information during its execution. However, the advancement of this advent brings important social and material vulnerabilities. If, on the one hand, the Internet exposes its users to new risk situations and to others that, although they already exist in the physical world, are enhanced in the digital world, as a result of the greater exposure and reach that technologies provide, on the other hand , due to the glaring limitations for ac IoT devices, they are easily susceptible to security breaches. These limitations mean that communication is most often carried out through an insecure connection, often compromising data and devices in relation to their confidentiality, integrity and availability.

In order to try to overcome these challenges, conventional security measures were introduced, namely firewall, data and communications encryption, audits and authentication and authorization mechanisms. Again, due to the limitations of these devices, it is not always feasible to use these countermeasures because they either do not fit or are not tolerant enough for the characteristics of IoT systems. However, one of the countermeasures that can be implemented in this type of networks is the intrusion detection systems (IDS). For this, an IDS must respect the IoT architecture, that is, it must take into account its scalability, the heterogeneity of the devices and their means of communication, privacy, among others.

The existing IDS solutions for IoT still have many limitations, so it is essential to improve these gaps, whether through new approaches, namely the detection methodologies used, the number of intrusions detected and in different IoT layers, the emerging interoperability in these systems , or simply improving the limitations already documented in the literature on the topic.

In order to mitigate these limitations, this project proposes specifications for the proposed framework for the purpose of detecting intrusions in IoT systems, whose main objective is to detect intrusions with internal or external origin in a timely manner that affect all layers of the IoT architecture. The proposed framework aims to collect and aggregate IoT communications to later analyze them in order to detect anomalies and intrusions. Regarding the analysis, this is done through an architecture based on a network intrusion detection system with a hybrid location, based on the collection of distributed data and a centralized analysis, subsequently using a detection methodology based on traffic specifications. normal through flows.

Regarding the specifications used, these were created based on traffic generated in a test scenario. In an initial phase, only normal traffic was generated for the creation of registers of normal traffic flows and respective specifications. In a second phase, anomalous traffic was generated, that is, traffic with attacks and threats, in order to create records of anomalous traffic flows and their specifications. Finally, for flows classified as anomalous traffic, specifications were created in order to be able to detect the type of attack, namely Net Scan attacks, attacks targeting valid and invalid URIs, including burst attacks, using the methods of the Constrained Application Protocol.

In order to validate the framework, a prototype was developed, which was subsequently analyzed and evaluated based on the results of the implementation of a test plan.

The tests carried out confirm the validation of the proposed framework, reaching quite satisfactory values.

That said, it is safe to say that without a strong security foundation, attacks and failures in IoT systems will outweigh any of their benefits.

Keywords: Internet of things; Intrusion detection systems; CoAP; Systems attacks; Systems security.

Lista de Figuras

Lista de Tabelas

Lista de Siglas

ACL Access Control List

AES Advanced Encryption Standard

AMQP Advanced Message Queuing Protocol

ASM Authenticated security mode

BLE Bluetooth Low Energy

CE Consumer Electronics

CoAP Constrained Application Protocol

CoAPS Constrained Application Protocol over DTLS

CSMA/CA Carrier-sense multiple access with collision avoidance

DCPS Data-Centric PublishSubscribe

DDoS Distributed Denial of Service

DDS Data Distribution Service

DHCP Dynamic Host Configuration Protocol

dIDS Distributed IDS

DIO Information Object

DIS DODAG Information Solicitation

DLRL Data-Local Reconstruction Layer

DNS Domain Name System

DNS-SD Domain Name System - Service Discovery

DoS Denial of Service

DTLS Datagram Transport-Layer Security

ECC Elliptic Curve Cryptography

EPC Electronic Product Code

HIDS Host-based Intrusion Detection System

HTTP Hypertext Transfer Protocol

IDS Intrusion Detection System

IE Information Elements

IEEE Institute of Electrical and Electronics Engineers

IETF Internet Engineering Task Force

ISM Industrial, Scientific and Medical

IoT Internet of Things

IP Internet Protocol

IPS Intrusion Prevention System

IPSO IP for Smart Objects

JSON JavaScript Object Notation

KDC Key Distribution Center

LTE-A Long-Term Evolution Advanced

MAC Medium Access Control

mDNS DNS multicast

MIC Message Integrity Code

MQTT Message Queue Telemetry Transport

MTU Maximum Transmission Unit

M2M Machine-to-Machine

NFC Near Field Communication

NIDS Network-based Intrusion Detection System

NTP Network Time Protocol

OASIS Organization for the Advancement of Structured Information Standards

OFDMA Orthogonal Frequency Division Multiple Access

OMG Object Management Group

OWL Ontology Web Language

PAN Personal Area Network

PHY Physical Layer

PKI Public Key Infrastructure

PRB Physical Resource Block

PSM Preinstalled secure mode

QoS Quality of Service

RAN Radio Access Network

RDF Resource Description Framework

REST REpresentational State Transfer

RFCOMM Radio Frequency Communication

RFID Radio Frequency Identification

RPL Routing over Low Power and Lossy Networks

RTOS Real Time Operating System

SCTP Stream Control Transmission Protocol

SHF Super High Frequency

SOA Service-Oriented Architecture

SSL Secure Socket Layer

TCP Transmission Control Protocol

TCP/IP Transmission Control Protocol/Internet Protocol

TLS Transport Layer Security

UDP User Datagram Protocol

UHF Ultra High Frequency

URI Uniform Resources Identifiers

XML Extensible Markup Language

XMPP Extensible Messaging e Presence Protocol

YAF Yet Another Flowmeter

WAN Wide Area Network

WPAN Wireless Personal Area Network

WSN Wireless Sensor Networks

6LoWPAN IPv6 over Low-Power Wireless Personal Area Networks

Índice

Capítulo 1

Introdução

Desde há muitos anos que se imaginam e implementam soluções baseadas no conceito de ligação de dispositivos à Internet, muito para além dos tradicionais computadores, que foram os primeiros meios de acesso. Assim, o que há alguns anos era um conceito é hoje uma realidade em concretização, com muitos meios, pessoas e ideias envolvidas. Na Era do IoT, uma grande quantidade de dispositivos inteligentes passaram a estar conectados à Internet através de redes com e sem fios. Este crescimento veio revolucionar não só a indústria da computação mas também a vida quotidiana das pessoas, ajudando a melhorar a performance dos resultados obtidos, poupando tempo e dinheiro a ambos. Inegavelmente, o IoT veio para ficar e a sua aceitação do mercado traduziu-se numa exponencial adesão por parte dos demais fabricantes, o que veio oferecer uma diversidade de dispositivos disponíveis ao consumidor. Todavia, embora a definição geral do IoT esteja quase madura, definindo-a como uma rede de informações conectando objetos virtuais e físicos, existe ainda uma falta consistente de consenso em torno de soluções técnicas e regulamentações.

O crescimento da utilização deste tipo de dispositivos originou a um crescimento de novas vulnerabilidades, surgindo assim, novas ameaças. O facto deste tipo de dispositivos, serem predominantemente qualificados como dispositivos limitados, tanto pelo seu baixo recurso de computação como pela sua baixa capacidade de energia, têm vindo a ser um obstáculo no que concerne à segurança e comunicação dos mesmos. De forma a tentar superar estas questões, a comunidade científica tem prestado especial atenção à eficiência dos recursos, tendo em consideração futuros problemas de escalabilidade, originando a que a implementação de recursos de segurança possa ser complexa.

A seleção de um protocolo para a transmissão de mensagens tem sido igualmente uma tarefa desafiadora, pois depende da natureza do sistema IoT. Protocolos construídos sobre *UDP*, são os mais indicados devido às limitações dos dispositivos IoT,

embora não seja o suficiente para garantir uma segurança mais eficaz, pois esses mesmos protocolos também têm menos segurança. O protocolo aplicacional CoAP é um protocolo leve que foi desenvolvido especialmente para sistemas IoT. Este é baseado em UDP e é definido pela RFC 7252. [2]

Estas limitações têm vindo a ser um grande obstáculo na implementação de segurança em dispositivos IoT. Investigadores tem realizado esforços neste sentido, tendo já realizado abordagens convencionais, implementando recursos de segurança, como por exemplo, criptografia [3], verificação de antivírus e deteção de intrusões nos dispositivos, embora esta abordagem geralmente tem aplicabilidade limitada devido aos recursos limitados nos dispositivos IoT.

Este impasse, tem conduzido a uma desconfiança em torno da garantia de segurança e privacidade dos dados que devem ser tratados com prioridade. [4]

De forma a superar problemas em torno da deteção de intrusões têm sido propostas várias soluções de IDS por vários investigadores. Algumas soluções apenas conseguem detetar com eficiência um ataque, nomeadamente o ataque *Sinkhole* [5], outras soluções apresentadas são muito restritas, não sendo recomendáveis para sistemas IoT [6] e, outras soluções em que têm uma elevada taxa de falsos positivos, baixa precisão de deteção, conseguindo apenas detetar um único ataque [7].

Outras soluções propostas centravam-se em IDS distribuídos [5] ou centralizados [8] aliados por metodologias de deteção com base em anomalias [9, 10], assinaturas [6], especificações [11, 12], entre outros.

A complexa tarefa de implementação de segurança nos dispositivos IoT e a continuidade de superar este desafio, demonstra que ainda não há resposta para muitas das abordagens utilizadas pela comunidade cientifica e, a prova disso é que apesar do IDS ser uma possibilidade, ainda não existe uma solução ideal para sistemas IoT compostos por dispositivos limitados.

Coloca-se então um conjunto de questões relacionado com a segurança dos dados e comunicações. Deste modo e, tendo este desafio em mente, este projeto visa responder a algumas questões relativamente à deteção de intrusões através de uma análise de registo de fluxos de tráfego IP e consequente resultados aos testes efetuados.

1.1 Motivação

Com a emergente tendência para a utilização de dispositivos IoT, tentando adaptar e conectar vários objetos do nosso dia-a-dia à Internet, com o intuito de os tornar mais interativos, fáceis de utilizar e inteligentes, é primordial garantir a segurança não só dos dispositivos, como dos utilizadores e respetivos dados. Dos aspetos mais importantes a ter em conta neste novo mundo é que estes equipamentos estão presentes em muitos dos aspetos do nosso quotidiano e geralmente estão ligados à rede global necessitando de ser protegidos.

Atendendo a este facto, muitos investigadores já tentaram dar o seu contributo visando melhorar questões ao nível da energia, da segurança, aplicando contra medidas, da adoção de protocolos a utilizar nestes sistemas, entre outros. Apesar disso, as soluções de segurança existentes para sistemas IoT ainda pecam pelo facto de não se conseguirem adaptar à arquitetura IoT. Soluções propostas ao nível da deteção de intrusões acabam por revelar fragilidades na adaptação a estes sistemas, quer ao nível do aparecimento de novas ameaças quer ao nível de ataques internos e externos, não conseguindo igualmente detetar ataques nas várias camadas constituintes da pilha protocolar para sistemas IoT.

Com o desafio de contribuir para a mitigação dos problemas existentes, definiu-se como principal objetivo deste projeto, a criação de especificações para uma *framework* proposta visando a detenção de intrusões para sistemas IoT com o intuito de detetar, em tempo útil, intrusões oriundas da rede interna ou externa. A deteção passa por uma análise a registos de fluxos de tráfego IP gerados num cenário de testes que servirá igualmente para validar a *framework* proposta.

Assim, esta contribuição pretende auxiliar na dura batalha que é manter os sistemas IoT seguros para que todos possam usufruir e tirar partido do melhor que estes dispositivos podem oferecer.

1.2 Objetivos

Na tentativa de dar resposta face aos problemas existentes na deteção de intrusões em sistemas IoT, este projeto visa contribuir para a melhoraria da segurança neste tipo de sistemas. Assim, com a realização do presente trabalho pretende-se atingir os seguintes

objetivos:

i. Análise das principais vulnerabilidade de segurança associadas aos sistemas IoT;

ii. Análise às soluções propostas para a deteção de intrusões para sistemas IoT;

iii. Análise aos registos de fluxos de tráfego gerados em ambiente controlado;

iv. Criação de especificações para a classificação de registos de fluxos de tráfego normais e anómalos, classificando-os posteriormente por tipo de ataque;

v. Realização de testes para validação, aprovação e performance da *framework* proposta.

1.3 Metodologia

Como ponto de partida para este projeto, começou-se por uma pesquisa cuidada, consultando RFCs, documentos de identidades oficiais, projetos e artigos publicados pela comunidade científica, procurando mais informações relativamente à temática IoT, à segurança que os mesmos possuem nos dias atuais e informações sobre sistemas de deteção de intrusões, privilegiando soluções de sistemas de deteção de intrusões para sistemas IoT.

Foi também definido o plano e cenário de testes que visa validar as especificações definidas para a *framework* proposta. Numa fase inicial, apenas foram gerados registos de fluxos de tráfego normal, de forma a que fosse possível criar as especificações para este tipo de tráfego de uma forma "limpa", isto é, sem ataques para não "contaminar" os registos de fluxos de tráfego normais. Numa segunda fase, foram gerados registos de fluxos de tráfego anómalos, de forma a que fosse possível criar as especificações para este tipo de tráfego visando conseguir especificar os registos de fluxos de tráfego por tipo de ataque. Posteriormente, foram realizados testes para detetar qual a taxa de deteção conseguida através das especificações declaradas. Todo este processo é sucintamente explicado no capítulo 4 e 5 deste relatório.

1.4 Estrutura do Documento

Este documento encontra-se dividido por seis capítulos, por forma a possibilitar um melhor manuseamento/compreensão do mesmo.

No presente capítulo são apresentadas as motivações, desafios e abordagem ao presente relatório.

No capítulo 2 são descritos os conceitos gerais sobre a temática IoT, protocolo aplicacional a ser utilizado neste projeto e uma abordagem às questões de segurança para sistemas IoT nas várias camadas.

No capítulo 3 são descritos os conceitos gerais sobre um IDS e respetiva classificação. É também abordado as soluções propostas de IDS para IoT.

No capitulo 4 é apresentada a *frameqork* proposta, juntamente com um cenário de testes de forma a validar a *framework*. É também apresentado as respetivas especificações para os registos de fluxos de tráfego normais e anómalos, tanto para CoAP como para CoAPS, aliado de um plano de testes.

No capítulo 5 são apresentados e discutidos os resultados dos testes realizados, tendo em consideração o plano de testes proposto.

Por último, no capitulo 6 é feita uma abordagem geral das conclusões finais retiradas ao longo do projeto.

Capítulo 2

Internet of Things

> "The Internet of Things has the potential to change the world, just as the Internet did. Maybe even more so."Kevin Ashton, 2009

O conceito Internet das Coisas ou IoT foi introduzido por Kevin Ashton em 1999 num contexto de classificar eletronicamente os produtos da empresa, de forma a simplificar a logística da linha de produção. [13] O autor descreve o IoT como um amplo ecossistema, onde os dispositivos e serviços interligados conseguem recolher e trocar dados, a fim de se adaptar dinamicamente a um contexto específico.

O IoT forneceu uma oportunidade promissora para a construção de sistemas e aplicações industriais preponderantes, impulsionados pela prevalência de dispositivos habilitados pela tecnologia sem fios, como o Bluetooth, identificação por radiofrequência (*RFID*), aproveitando a omnipresença crescente do mesmo, Wi-Fi e serviços móveis, bem como nós incorporados de sensores e atuadores. Na última década, a definição tem sido mais inclusiva. Este crescimento advém pelo elevado impacto que estes dispositivos podem ter em vários aspetos no nosso quotidiano, resultando numa clara aceitação no mercado, tanto por parte do consumidor final como de empresas dos mais variados sectores económicos, como por exemplo, setores ligados à área da saúde ou à área de transportes. [14] Devido ao seu impacto global, inegavelmente o IoT cresceu, encontrando-se hoje à beira de transformar a atual Internet em uma Internet do futuro totalmente integrada. [15]

Segundo a empresa Gartner [1], prevê-se que até 2020 haverá mais de 20 mil milhões de dispositivos conectados, traduzindo-se numa ascensão por parte do mercado. [16]

[1] A Gartner é uma empresa de consultadoria e pesquisa que fornece informações, conselhos e ferramentas sobre as várias tecnologias que surgem no mercado.

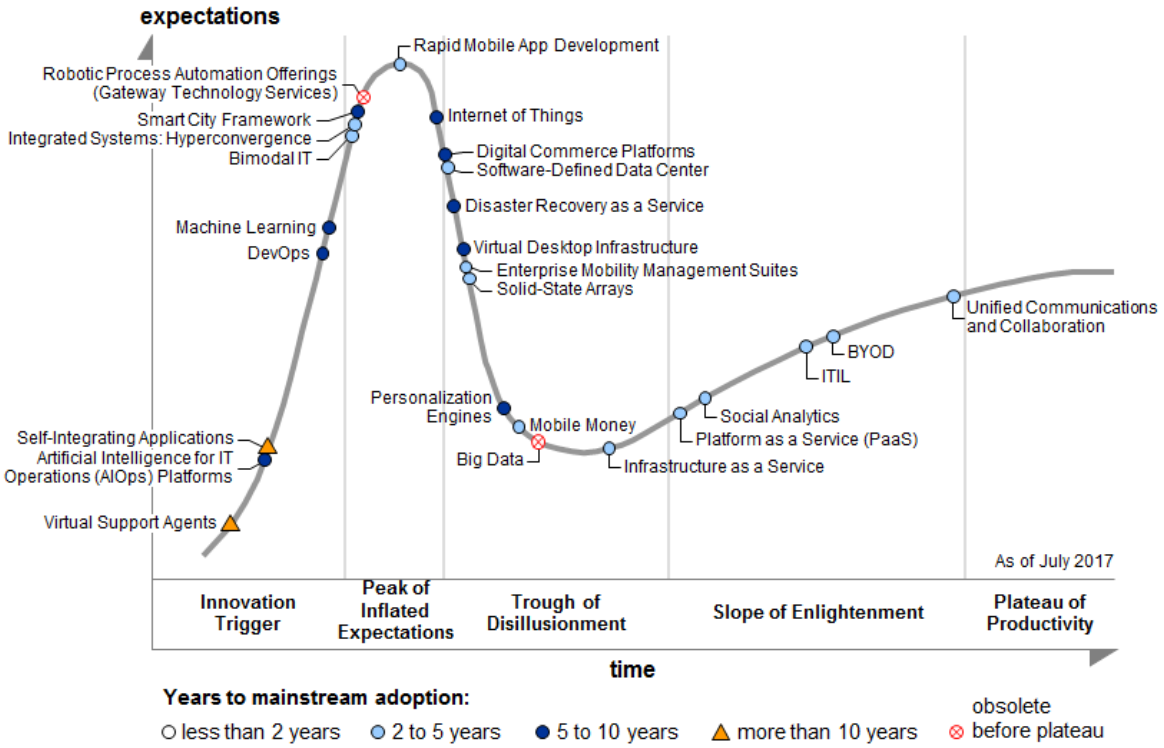


Figura 2.1: Hype Cycle das tecnologias emergentes - 2017

Conforme se pode constatar pela figura 2.1, retirado de [17], o IoT já atingiu o pico de expectativas. Em 2017, Mark Hung Vice-Presidente da empresa publicou um artigo [16], onde salienta o desenvolvimento e envolvimento do IoT nos vários setores até 2020, em particular no setor da indústria. Dos aspetos mais importantes destacam-se os seguintes:

- Mais de 65% das empresas (acima de 30% hoje) adotarão produtos IoT;
- Mais de 25% dos ataques identificados nas empresas envolverão dispositivos IoT;
- A Pesquisa de *Backbone* IoT de 2016 da Gartner mostrou que 32% dos líderes de Tecnologias de Informação citam a segurança como uma barreira principal para o sucesso no IoT;
- A falta de especialistas em ciência de dados inibirá 75% das organizações de alcançar todo o potencial do IoT;
- A Gartner estima que as "Coisas" conectadas à Internet superarão os humanos em 4 para 1.

Assim, o objetivo principal deste capítulo recai em dar ao leitor a oportunidade de compreender o conceito, as várias arquiteturas existentes, os protocolos utilizados e, por último a segurança e a privacidade.

2.1 Definição

Tendo em consideração o facto do IoT ser um conceito complexo, este gera alguma dificuldade na sua compreensão, desde o que se encontra por detrás do conceito até às suas implicações sociais, económicas e à implementação das técnicas propriamente ditas.

De acordo com o grupo de projetos de pesquisa europeus para o IoT [14], as "Coisas" são participantes ativos em negócios onde são capazes de comunicar entre si e com o ambiente que os rodeia, trocando dados e informações sobre o mesmo, reagindo de forma autónoma aos eventos, executando processos que acionam ações e criam serviços com ou sem intervenção humana.

Segundo o artigo de Atzori et al. [18], o IoT pode ser considerado como a conjugação de três paradigmas - Orientados à Internet (*middleware*), Orientados a Coisas (*Sensors*) e Orientados à Semântica (*Knowledge*). Embora esse tipo de delimitação seja necessário devido à grande heterogeneidade existente no IoT, este pode ser desencadeado apenas num domínio de aplicação em que os três paradigmas se cruzam. De uma forma simplificada, o IoT pode ser considerado como uma infraestrutura de rede global composta por vários dispositivos conectados que dependem de tecnologias de processamento sensorial, de comunicação, de rede e de informações. As Tecnologias de identificação e pesquisa (*tracking*), redes de sensores e atuadores com e sem fios ou protocolos de comunicação são apenas alguns dos mais relevantes. A inclusão destas tecnologias serve como base ao IoT, elucidando-nos como uma variedade de dispositivos e objetos físicos que se encontram em nosso redor, podendo ser associados à Internet e permitindo que esses dispositivos e objetos cooperem e comuniquem entre eles, de forma a alcançar objetivos em comum. [19] Deste modo, o IoT nada mais é do que uma expansão de conectividade, uma vez que deixamos de utilizar apenas os dispositivos "tradicionais" e, começamos a retirar proveito dos benefícios que a Internet nos proporciona para qualquer cenário e o que envolve em torno deste tipo de objetos.

Todavia, embora a definição geral do IoT esteja quase madura, existe ainda uma falta consistente de consenso em torno de soluções técnicas e regulamentações.

2.2 Visão

A causa da indeterminação evidente, ainda nos dias de hoje, em torno deste conceito, prende-se no facto de Internet das Coisas ainda ser uma visão que se encontra em desenvolvimento e onde se tenta, de alguma forma, interpretar o conceito de IoT em relação às suas necessidades. Este conceito é composto por três termos, como anteriormente referenciado na secção 2.1.

O primeiro termo Orientados à Internet *(Middleware)* é direcionado para uma visão de IoT para a rede, no qual devem estar cientes os protocolos IPv4 e IPv6. São exemplos os *IP for Smart Objects (IPSO)*. O segundo termo Orientados a Coisas *(Sensors)* direciona o foco para os "objetos" genéricos a serem integrados numa estrutura comum. Esta técnica é estendida aos sensores e é importante compreender o facto de que a visão futura dependerá dos sensores e das suas capacidades para cumprir a visão orientada para as "Coisas". [20] São exemplos os RFID, NFC, sensores ou atuadores. Por último, o terceiro termo Semântica *(Knowledge)*, onde Internet das Coisas semanticamente significa uma rede mundial de objetos que permitem a atribuição de endereços IP, baseados em protocolos de comunicação padrão [21]. Exemplos de Semântica é *dataming*.

Através do diagrama de Ven da figura 2.2, podemos observar a conjugação das três visões que formam o IoT.

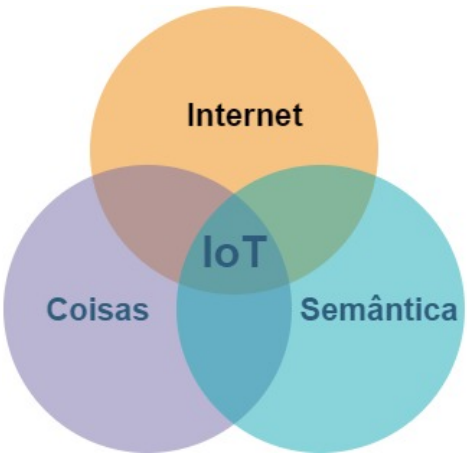


Figura 2.2: Diagrama de Venn do IoT

2.3 Elementos

A compreensão da taxonomia no IoT ajuda a obter uma melhor interpretação do seu real significado e da sua funcionalidade. Através da figura 2.3 podemos observar os constituintes desta taxonomia a partir de uma perspetiva de alto nível.

Figura 2.3: Taxonomia IoT

A **identificação** é um processo crucial, pois é neste ponto que são identificados os dispositivos. Para a sua identificação existem métodos como códigos *Electronic Product Code (EPC)* ou códigos omnipresentes (uCode). A atribuição de endereços IP é fundamental pois permite diferenciar entre o ID (nome) de um objeto e o seu endereço IP. Em relação aos **sensores**, a deteção passa por recolher dados dos objetos dentro da rede e enviá-los de volta para um repositório local ou *cloud*. Esses dados são posteriormente analisados para realizar ações específicas com base nos serviços requeridos. São exemplos, os sensores inteligentes ou atuadores. Na **comunicação**, as tecnologias presentes no IoT permitem conectar objetos heterogéneos de forma a fornecerem serviços específicos. Exemplos de meio de comunicação utilizados são o RFID, o Wi-Fi, o Bluetooth, o *Institute of Electrical and Electronics Engineers (IEEE)* 802.15.4, entre outros. Na **computação** as unidades de processamento e aplicações de *software* representam o "cérebro" e a capacidade computacional do IoT. Na vertente de *hardware*, temos componentes como Arduino ou Raspberry Pi e na vertente de *software* temos sistemas operativos em tempo real *Real Time Operating System (RTOS)*. Em relação aos **serviços**, estes podem ser categorizados em quatro classes: Serviços relacionados com a identidade (*shipping*), Serviços de agregação de informações (*smart grid*), Serviços de colaboração (*smart home*) e Serviços ubiquitous (*smart city*). Os serviços relacionados à identidade são os mais básicos e importantes, onde cada aplicação que necessite identificar objetos do mundo real para o mundo virtual precisa de identificar esses objetos. Os serviços de agregação de informações recolhem as medições sensoriais que necessitam ser processadas e relatadas para o dispositivo IoT. Os serviços de colaboração atuam sobre os serviços de agregação de informações e utilizam os dados obtidos para tomar decisões e reagir. Os serviços ubiquitous têm como objetivo fornecer serviços de suporte colaborativo para melhorar e aumentar o consumo de energia de casas e edifícios. Por último, a **semântica** refere-se à capacidade de extrair conhecimento de forma inteligente por diferentes máquinas para fornecer os serviços necessários. Além

disso, inclui o reconhecimento e a análise de dados para dar "sentido" à decisão certa de fornecer o serviço exato [22]. Esse requisito é suportado pelas tecnologias como o *Resource Description Framework (RDF)* e o *Ontology Web Language (OWL)*.

2.4 Tecnologia de Comunicação

A atualização do conceito do IoT no mundo real é possível através da integração de várias tecnologias. Um fator importante acerca deste tipo de dispositivos é que os mesmos são predominantemente qualificados por dispositivos limitados pelo baixo recurso de computação e capacidade de energia. De forma a superar estas questões é necessário prestar especial atenção à eficiência dos recursos, tendo em consideração os problemas de escalabilidade. Através da figura 2.4, adaptado de [23], pode-se verificar o alcance das várias tecnologias utilizadas em sistemas IoT.

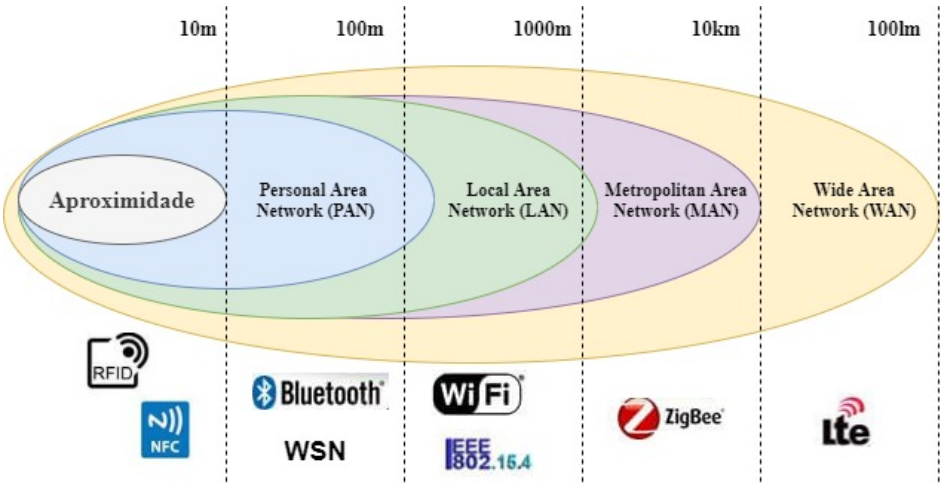


Figura 2.4: Alcance das tecnologias utilizadas no IoT

2.4.1 *RFID*

Os sistemas RFID são um dos principais componentes no IoT, uma vez que são classificados como sistemas de rádio e são utilizados para captura de dados. Este tipo de sistemas é composto por um ou mais leitores e várias etiquetas RFID. Um exemplo de um sistema RFID poderá ser um RFID fixado no para-brisas de carros alugados, podendo estes guardarem a identificação do veículo ou ajudar na localização do mesmo. A sua transmissão pode ocorrer em várias bandas de frequência, que vão desde baixas

frequências (LF) a 124–135 kHz até frequências ultra altas (UHF) a 860–960 MHz que possuem o maior alcance. [24]

2.4.2 *NFC*

Esta é uma tecnologia de proximidade, o que significa que só funciona quando dois dispositivos estão próximos ou em contacto. Quando isto não ocorre, o NFC está inativo e não consome energia nem partilha informações. Assim, o NFC garante a adoção bem-sucedida dos serviços de IoT numa casa inteligente. [25] Exemplos da utilização de NFC passam por obter leituras instantâneas, por exemplo, do *status* ou diagnósticos do dispositivo. A transmissão neste tipo de tecnologia fixa-se nos 13,56 MHz. [26]

2.4.3 Redes de Sensores Sem Fios (WSN)

Esta tecnologia pode cooperar com os sistemas RFID de forma a melhorar a monitorização do *status* dos objetos, ou seja, a sua localização, temperatura, movimentos, etc. A utilização desta tecnologia tem sido proposta em vários cenários de aplicação, como monitorização ambiental, saúde, sistemas de transporte inteligentes, entre outros. Atualmente, a maioria das soluções de redes de sensores sem fios comerciais baseia-se no padrão IEEE 802.15.4. [24]

2.4.4 Bluetooth

O Bluetooth é uma tecnologia de comunicação sem fios de curta distancia, muito diversificada, extensa e complexa. O Bluetooth incorpora um módulo designado de *Radio Frequency Communication (RFCOMM)* que emula a dita comunicação em série. Soluções como o *Bluetooth Low Energy (BLE)* estão a expandir as funcionalidades dos dispositivos IoT, criando uma estrutura mais confiável para uma maior conectividade. A tecnologia BLE visa melhorar e otimizar a operacionalidade global dos dispositivos domésticos inteligentes, criando velocidades de comunicação mais rápidas e ampliando o alcance do sinal. A transmissão neste tipo de tecnologia opera na banda de frequência de 2.4 GHz (2400 – 2483,5 MHz).[27]

2.4.5 Wi-Fi

O Wi-Fi é uma rede sem fios local que executa os padrões 802.11 estabelecidos pelo IEEE e opera nas bandas de rádio *Ultra High Frequency (UHF)* de 2.4GHz e *Super High Frequency (SHF) Industrial, Scientific and Medical (ISM)* de 5GHz. Com esta tecnologia, existe a facilidade de configuração, o suporte nativo a redes IP, as ferramentas de gestão de rede disponíveis e a familiaridade que as pessoas têm com elas. O Wi-Fi também oferece um alcance muito maior em comparação aos seus principais concorrentes de tecnologia, como se pode observar na figura 2.4. No entanto, o Wi-Fi é tradicionalmente uma tecnologia que exige mais energia do que as alternativas, portanto, para garantir a sua viabilidade para o IoT, é necessário tomar medidas para garantir que ela seja projetada da maneira mais económica possível. São exemplos práticos, lâmpadas com ligação Wi-Fi ou semáforos inteligentes.

2.5 Arquitetura

Os dispositivos de IoT são um enorme desafio para os investigadores desta tecnologia devido à elevada heterogeneidade existente. Assim, o IoT deve ter a capacidade de conectar milhões de objetos heterogéneos através da Internet e, para isso, necessita de uma arquitetura flexível em camadas.

O crescente número de arquiteturas propostas ainda não convergiu para um modelo de referência. [28] Atualmente, ainda não existe um consenso único sobre arquitetura para o IoT que seja universalmente aceite. Diferentes arquiteturas têm sido propostas por diferentes investigadores e que consistem em quatro modelos de referência. O modelo mais básico é composto por um arquitetura de três camadas sendo proposto por Khan et al. [29]. Na literatura mais recente, têm sidos propostos outros modelos, como os de Chaqfeh et al. [30], Tan et al. [31] ou Al-Fuqaha et al. [32] e têm como finalidade adicionar mais abstração à arquitetura IoT, passando para uma arquitetura de cinco camadas. Na figura 2.5, adaptado de [32], é demonstrado as quatro arquiteturas propostas pelas diferentes pesquisas realizadas até à data.

Não existindo um consenso claro sobre qual a arquitetura mais indicada e, atendendo ao facto que na literatura consultada a arquitetura de três camadas é a mais adotada, considerar-se-á neste trabalho esse modelo.

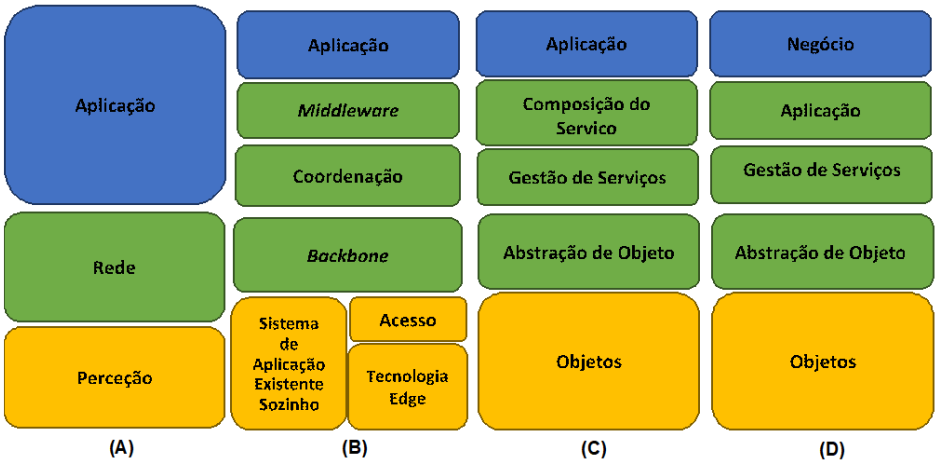


Figura 2.5: (A) Modelo de três camadas (B) *Middleware* (C) Modelo de cinco camadas (D) SOA

2.5.1 Camada de Perceção

A camada de perceção é também conhecida como a camada sensorial. Esta camada consiste em objetivos físicos e sensores com a finalidade de identificar e recolher informações específicas. Dependendo do tipo de sensores, as informações podem ser sobre localização, temperatura, movimento, entre outros. As informações recolhidas são posteriormente passadas para a camada de Rede.

2.5.2 Camada de Rede

A camada de Rede pode ser visto como a rede neural e o cérebro do IoT ou apenas como uma camada de transmissão. A sua principal função é transmitir, com segurança, as informações dos sensores para o sistema de processamento de informações. O meio de transmissão pode ser com ou sem fios e a tecnologia pode ser Wi-Fi, Bluetooth, infravermelho, Zigbee, aliados aos protocolos *Transmission Control Protocol/Internet Protocol (TCP/IP)*. Assim, a camada de Rede transfere as informações da camada de Perceção para a camada de Aplicação.

2.5.3 Camada de Aplicação

A camada de aplicação é também conhecida como a camada de negócio. Esta camada disponibiliza às aplicações a capacidade de processar, gerir e utilizar os dados obtidos

na camada de perceção acerca do ambiente físico. [32] É igualmente responsável por fornecer serviços solicitados pelos utilizadores, como por exemplo, fornecer medições de temperatura.

2.6 Modelos Standards

Diferentes grupos foram criados para fornecer protocolos de apoio ao IoT, incluindo esforços liderados pelo *World Wide Web Consortium* (W3C), a Internet Engineering Task Force (IETF), a EPCglobal, o IEEE e as Normas Europeias de Telecomunicações. A figura 2.6, adaptado de [32], fornece um resumo dos protocolos mais proeminentes definidos por esses grupos.

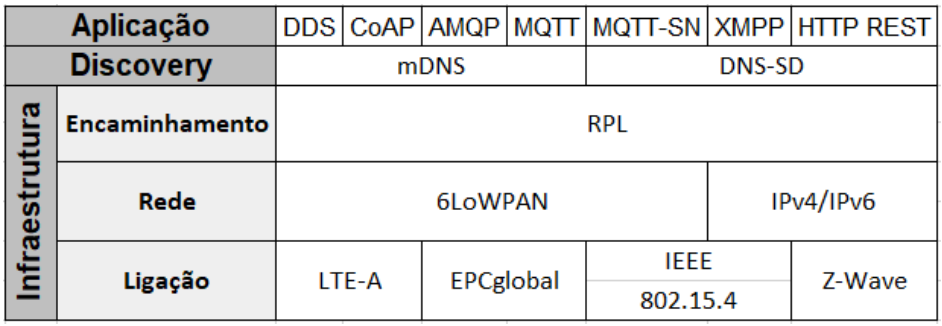


Figura 2.6: Standards e protocolos

Podemos observar, através da imagem 2.6, que os protocolos são agrupados em três categorias amplas: protocolos de aplicação, protocolos de descoberta de serviço e protocolos de infraestrutura.

De seguida, é abordada uma visão geral de alguns dos protocolos mais comuns por categoria, assim como, as suas principais funcionalidades.

2.6.1 Protocolos de infraestrutura

Os protocolos de infraestrutura são compostos por protocolos da camada de Ligação, Rede e Encaminhamento. Na camada de Ligação é onde os objetos são identificados, quer seja por endereços *Medium Access Control (MAC)* ou por um número de identificação exclusivo, utilizado na tecnologia RFID. Na camada de Rede é onde são atribuídos endereços IP aos objetos, encapsulamento de cabeçalhos e fragmentação para não ex-

ceder o máximo de *Maximum Transmission Unit (MTU)*. Por último, na camada de Encaminhamento é onde os pacotes são encaminhados para a camada superior. Este encaminhamento é realizado através da criação de vários caminhos de forma a precaver a perda de pacotes através de *links* que possam conter perdas.

2.6.1.1 Ligação

- *Long-Term Evolution Advanced (LTE-A)*

  O LTE-A é um protocolo de baixo custo e engloba um conjunto de protocolos de comunicação rádio para se adequar à comunicação *Machine-to-Machine (M2M)* e a dispositivos IoT. O LTE-A utiliza o *Orthogonal Frequency Division Multiple Access (OFDMA)* para dividir a largura de banda do canal em bandas menores, chamadas de *Physical Resource Block (PRB)*. A arquitetura da rede LTE-A conta igualmente com um *Core Network*, que controla dispositivos móveis e lida com fluxos de pacotes IP e, conta com uma *Radio Access Network (RAN)*, que é responsável por estabelecer o controlo dos dados, gerir a conectividade sem fios e gerir o controlo de acesso por rádio. [33]

- EPC

  O EPC foi desenvolvido pela EPCglobal com o intuito de identificar objetos através de um número de identificação exclusivo. Esse número de identificação é posteriormente armazenado numa etiqueta RFID. A arquitetura subjacente nas tecnologias RFID aliadas às etiquetas e leitores RFID servem como meio para partilhar informações sobre os objetos. [34] Esta arquitetura, é reconhecida como uma técnica promissora para o futuro nos dispositivos IoT devido à sua abertura, escalabilidade, interoperabilidade e confiabilidade. [35]

- IEEE 802.15.4

  O protocolo IEEE 802.15.4 foi criado para especificar uma sub-camada para o MAC e uma *Physical Layer (PHY)* para redes de área privada sem fios de baixa taxa (LR-WPAN). Devido às suas especificações, como baixo consumo de energia, baixa taxa de dados, baixo custo e alta taxa de transferência de mensagens, ele também é utilizado pelo IoT, M2M e WSN. Ele fornece um alto nível de segurança, criptografia e serviços de autenticação. No entanto, não fornece garantias de *Quality of Service (QoS)*. [36]

- Z-Wave

  Z-Wave é como um protocolo de comunicação sem fios de baixa potência projetado para redes *Personal Area Network (PAN)* e tem sido utilizado para comunicação em sistemas IoT, especialmente para domínios residenciais e domínios comerciais pequenos. Este protocolo foi inicialmente desenvolvido pela ZenSys e mais tarde foi aprimorado pela Z-Wave Alliance. O Z-Wave cobre cerca de 30 metros de comunicação ponto-a-ponto e é adequado para pequenas mensagens em dispositivos IoT, como o controlo de energia, o controlo de eletrodomésticos, a deteção de incêndios, entre outros. Utiliza também o método *Carrier-sense multiple access with collision avoidance (CSMA/CA)* para deteção de colisão e mensagens ACK para transmissão confiável.

2.6.1.2 Rede

- 6LoWPAN

  IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) é um dos primeiros padrões utilizados pelo IETF nesta categoria, sendo desenvolvido em 2007. Este protocolo é uma especificação dos serviços de mapeamento exigidos pelo IPv6 sobre *Wireless Personal Area Network (WPAN)* de baixa potência para manter uma rede IPv6 [37]. O protocolo encapsula eficientemente os cabeçalhos para reduzir a sobrecarga de transmissão, utiliza a fragmentação para atender ao máximo de *MTU*, que não pode exceder os 128 bytes, e o encaminhamento para a camada de ligação para suportar a entrega de múltiplos saltos. [38] As especificações do 6LoWPAN permitem muitos recursos, incluindo diferentes endereços de comprimento, diferentes tipologias de rede, baixa largura de banda, baixo consumo de energia, custo eficiente, mobilidade, confiabilidade e longos períodos de inatividade. [39]

- IPv4 e IPv6

  Os dispositivos IoT acompanham o progresso da tecnologia, permitindo ao utilizador optar por um endereçamento IPv4 e/ou IPv6. A utilização de endereçamento IP é essencial para a correta identificação dos dispositivos, pois permite diferenciar o ID (nome) de um objeto e o seu endereço IP. A utilização do protocolo IP é crucial para o bom funcionamento dos sistemas IoT, pois permite a criação e transporte de pacotes de dados que serão entregues a cada dispositivo.

2.6.1.3 Encaminhamento

- *Routing over Low Power and Lossy Networks (RPL)*

  É um protocolo de vetor de distância projetado pelo IETF para encaminhamento nos sistemas IoT. Este suporta uma variedade de protocolos, incluindo os abordados nos protocolos de aplicação. O RPL foi criado para oferecer suporte a requisitos mínimos de encaminhamento, através da criação de uma topologia robusta sobre *links* com perdas. Este protocolo de encaminhamento suporta modelos simples e complexos, como multiponto-ponto, ponto-multiponto e ponto-a-ponto. [40]

2.6.2 Protocolos de *Discovery*

A alta escalabilidade do IoT requer um mecanismo de gestão de recursos que seja capaz de registar, descobrir recursos e serviços de maneira auto-configurada, eficiente e dinâmica. Os protocolos mais dominantes nessa área são o *DNS multicast (mDNS)* e o *Domain Name System - Service Discovery (DNS-SD)*. Estes podem descobrir recursos e serviços oferecidos pelos dispositivos IoT. Embora estes dois protocolos tenham sido projetados originalmente para dispositivos de recursos, existem estudos que adaptam versões leves para ambientes IoT.

- Multicast DNS (mDNS)

  mDNS resolve nomes de *host* para endereços IP dentro de pequenas redes que não incluem um servidor de nomes local, ou seja, é um serviço que é executado num dispositivo para torná-lo detetável através de consultas DNS na rede local. [41]

- DNS Service Discovery (DNS-SD)

  A função de agregação dos serviços requisitados pelos clientes utilizando mDNS é uma chamada do serviço baseado em DNS-SD. Utilizando este protocolo, os clientes podem descobrir um conjunto de serviços desejados numa rede específica, utilizando mensagens DNS padrão. Essencialmente, o DNS-SD utiliza o mDNS para enviar pacotes DNS para endereços multicast específicos através do UDP. [42]

2.6.3 Protocolo de Aplicação

A tecnologia de comunicação padrão e em tempo real é uma inevitabilidade total para o desenvolvimento das redes para IoT. No entanto, a seleção de um protocolo de mensagens padrão e eficaz é uma tarefa desafiadora para qualquer organização, pois depende da natureza do sistema de IoT e de seus requisitos de mensagens. Neste ponto será realizado um breve resumo dos vários protocolos da camada de aplicação do IoT que são utilizados para a transmissão de mensagens e, que foram padronizados por diferentes organizações de padronização. A figura 2.7, adaptado de [32], resume algumas das caraterísticas dos protocolos mais utilizados no IoT.

Para a realização deste projeto será dado ênfase ao protocolo CoAP, visto ser o protocolo aplicacional a ser utilização na realização dos testes descritos no plano de testes, no capitulo 4.4.

Protocolo de Aplicação	Transporte	Request/ Response	Security
CoAP	UDP	✓	DTLS
MQTT	TCP	x	SSL
AMQP	TCP	x	SSL
XMPP	TCP	✓	SSL

Figura 2.7: Comparação dos protocolos mais utilizados no IoT

- CoAP

  É um protocolo da camada de aplicação projetado pelo grupo IETF Constrained RESTful Environments (CoRE), para fornecer uma interface Hypertext Transfer Protocol (HTTP) leve [43]. O CoAP permite que sensores de baixa potência utilizem serviços HTTP enquanto atendem às restrições de energia. Ele é construído sobre UDP, em vez de *TCP* comummente utilizado em HTTP e, tem um mecanismo para fornecer confiabilidade. [2]

- MQTT

  É um protocolo de mensagens que foi introduzido por Andy StanfordClark e Arlen Nipper em 1999 e foi padronizado em 2013 pela *Organization for the Advancement of Structured Information Standards (OASIS)* [44]. O MQTT é uma arquitetura de publicador/subscritor para fornecer flexibilidade de transição e simplicidade de implementação, em que o sistema consiste em três componentes principais:

editores, assinantes e um intermediário. De salientar que o MQTT é construído sobre o protocolo TCP.

- *Advanced Message Queuing Protocol (AMQP)*

  É um protocolo de camada de aplicação projetado para o setor financeiro. Ele foi construído sobre TCP e fornece uma arquitetura de publicação/subscrição semelhante à do MQTT. A diferença é que o intermediário é dividido em dois componentes principais, *exchange* e *queues*. O intermediário é responsável por receber mensagens do editor e distribuí-las em *queues* com base em funções e condições predefinidas. As *queues* representam os tópicos e são assinados pelos subscritores que receberão os dados sensoriais sempre que estiverem disponíveis na fila [44].

- *Extensible Messaging e Presence Protocol (XMPP)*

  É um protocolo padronizado pelo IETF e já foi utilizado em dispositivos IoT devido à utilização do protocolo *Extensible Markup Language (XML)*, tornando-o facilmente extensível. Este protocolo suporta a arquitetura de publicação/subscrição e solicitação/resposta, sendo construído sobre o protocolo TCP. Através das suas mensagens, permite que os utilizadores comuniquem entre si independentemente do sistema operacional que estiverem utilizando. Suporta eficientemente pequenas mensagens de baixa latência, contudo não fornece nenhuma garantia de qualidade de serviço, logo, não é prático para comunicações M2M. Além disso, as mensagens XML criam uma sobrecarga adicional devido a muitos cabeçalhos e formatos que aumentam o consumo de energia, o que é crítico, para o dispositivos IoT. [45]

2.6.3.1 CoAP - Constrained Application Protocol

> "O HTTP passou por mais de uma década de crescimento organizado, acumulando cada vez mais implementações, o que supera a capacidade de ser utilizado em dispositivos pequenos."(Bormann; Castellani; Shelby, 2012) [46]

Protocolos de rede direcionados para dispositivos pequenos são projetados para não existir sobrecarga. Este facto motivou a criação de protocolos como o CoAP. Assim, o CoAP é um protocolo da camada de aplicação, definido pela RFC 7252 [2] e, foi projetado pelo grupo IETF Constrained RESTful Environments (CoRE). Como o

HTTP, o CoAP é baseado no modelo REST[2], ou seja, servidores disponibilizam recursos num URL e os clientes acedem a esses recursos utilizando métodos como GET, PUT, POST e DELETE. Este formato representa uma maneira mais simples de trocar dados entre clientes e servidores. [43] O CoAP possibilita a utilização e identificação de diferentes tipos de carga útil (*payload*) e integra-se com XML, JSON, CBOR, ou qualquer formato de dados de sua escolha. Este protocolo é construído sobre UDP, em vez de TCP, comummente utilizado em HTTP e, contém um mecanismo para fornecer confiabilidade. De salientar que o CoAP fornece um modelo de interação de solicitação/resposta (*request/responde*) entre os dispositivos. Em relação à segurança, o CoAP utiliza parâmetros DTLS para obter segurança semelhante à presente na Web.

De acordo com a RFC 7252 [2] o protocolo CoAP tem como principais características:

- Protocolo Web atendendo aos requisitos M2M em ambientes restritos;
- Troca de mensagens assíncronas;
- Baixa sobrecarga e simples de analisar;
- Suporte à URI e tipo de conteúdo (*Content-type*);
- Recursos de proxy e *cache*;
- Interligação segura através do DTLS;
- Suporte aos métodos GET,POST, PUT, DELETE.

Formato da Mensagem

CoAP é baseado na troca de mensagens compactas que são transportadas sobre UDP. Este tipo de mensagens estão codificados num formato binário e têm um cabeçalho fixo de 4 *bytes*. A figura 2.8 mostra os campos do cabeçalho CoAP.

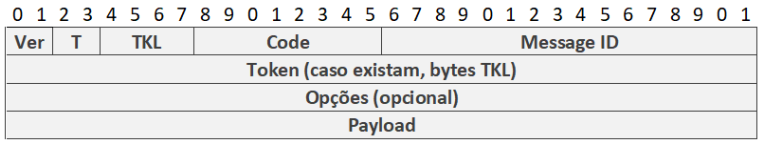


Figura 2.8: Cabeçalho CoAP

[2]REpresentational State Transfer (REST) é uma interface padrão entre o cliente HTTP e os servidores.

- **Versão (VER):** Campo de 2 *bits* que indica o número da versão do protocolo a ser utilizado por ambos os nós. Caso o valor seja desconhecido neste campo, a mensagem deve ser ignorada;

- **Tipo (T):** Campo de 2 *bits* que indica se a mensagem é do tipo *Confirmable* (0), *Non-Confirmable* (1), *Acknowledgement* (2) ou *Reset* (3);

- **Comprimento do Token (TKL):** Campo de 4 *bits* que indica o tamanho do campo *Token*, na segunda linha do cabeçalho;

- **Código:** Campo de 8 *bits* que caracteriza o tipo de mensagem, podendo indicar o método de requisição (table 2.1) ou o código de resposta (tabela 2.2);

Tabela 2.1: Códigos de métodos CoAP

Código	0.01	0.02	0.03	0.04
Nome	GET	POST	PUT	DELETE

Tabela 2.2: Códigos de resposta CoAP

Código	**Nome**
2.01	Criado
2.02	Apagado
2.03	Válido
2.04	Alterado
2.05	Conteúdo
4.00	Bad Request
4.01	Não Autorizado
4.02	Bad Option
4.04	Não Encontrado
4.05	Método não permitido
5.03	Serviço Indisponível
5.05	Proxying não Suportado

- **ID da Mensagem:** Campo de 16 *bits* utilizado para a deteção de duplicações de mensagens e também para comparação de mensagens do tipo ACK.

- **Token:** Campo de comprimento variável (0 a 8 *bytes*) com informação útil, tendo como finalidade relacionar a requisição à resposta, isto é, deve ser implementado de forma a que o *token* para o par cliente/servidor seja único;

- **Opções:** É um campo utilizado para definir informações como: URI do destinatário da mensagem, porto de destino, caminho do recurso (*path*, *query*(em caso de parametrização)), formato do *payload*, formato de conteúdo aceite como resposta, tempo limite para *cache*, entre outros.

- **Payload:** O *payload* de uma mensagem é a representação do recurso requisitado ao servidor, como por exemplo, GET, POST ou PUT. Caso o cliente requisite a URI "temperatura" e o servidor possuir a informação de que a temperatura é de 22°C, então essa informação será retornada no campo de carga útil.

Tipos de Mensagem e Resposta

A arquitetura do CoAP está dividida em duas sub-camadas principais, Mensagem e Solicitação/Resposta. A sub-camada de mensagens é responsável por lidar com o protocolo UDP, pela confiabilidade e duplicação de mensagens, enquanto a sub-camada de solicitação/resposta é responsável pela comunicação. Na figura 2.9 é demonstrado como é composto a arquitetura CoAP e onde se situa na pilha protocolar.

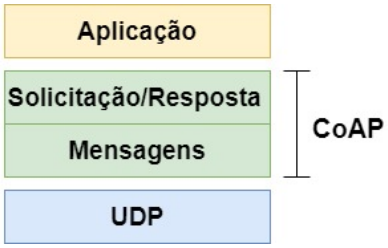


Figura 2.9: Pilha protocolar com suporte a CoAP

A sub-camada de Mensagem do CoAP possui quatro modos de mensagens: CON (*Confirmable*), NON (*Non-confirmable*), ACK (*Acknowledgement*) e RST (*Reset*).

- ***Confirmable*:** Este tipo de mensagens solicita a confirmação da receção da mensagem pelo destinatário. Essa confirmação passa por receber um ACK com o mesmo ID de mensagem (como 0x7d34 na figura 2.10). Caso o destinatário receba a mensagem mas, por alguma razão, falhe ao processar a mesma, ele responderá substituindo ACK pelo RST.

- ***Non-confirmable*:** Este tipo de mensagens não necessitam de confirmação por parte do destinatário. Aqui, também se o destinatário receber a mensagem mas, por alguma razão, falhe ao processar a mesma, ele responderá com um RST.

- ***Acknowledgment*:** São mensagens que confirmam o recebimento de uma mensagem *Confirmable*. É importante ressaltar que, por si só, uma mensagem ACK não indica sucesso ou falha de nenhuma requisição encapsulada na mensagem *Confirmable*.

- ***Reset*:** Indica que outra mensagem (CON ou NON) foi recebida, mas por falta de algum contexto ela não foi devidamente processada. Este tipo de mensagem pode ocorrer no caso de algum dispositivo ter reiniciado e a mensagem enviada não foi devidamente interrompida.

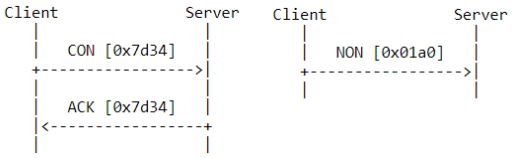


Figura 2.10: CON e NON [2]

Na sub-camada de Solicitação/Resposta o CoAP possui dois modos resposta: *Piggy-backed* e *Separate Responde*.

- ***Piggy-backed*:** É utilizado para comunicação direta entre cliente/servidor, onde o servidor envia sua resposta após receber a mensagem, ou seja, dentro da mensagem de confirmação(ACK). Na figura 2.11, para obter uma resposta bem-sucedida, o ACK contém uma mensagem de resposta (identificada por um *token*); para a resposta a falhas, o ACK contém o código de resposta a falhas.

- ***Separate Response*:** Se o servidor receber uma mensagem do tipo CON, mas não conseguir responder a essa solicitação imediatamente, ele enviará um ACK vazio. Quando o servidor estiver pronto para responder a essa solicitação, ele enviará um novo CON ao cliente e o cliente responderá uma mensagem ACK. O ACK é apenas para confirmar a mensagem CON, independentemente da solicitação ou resposta da mensagem CON.

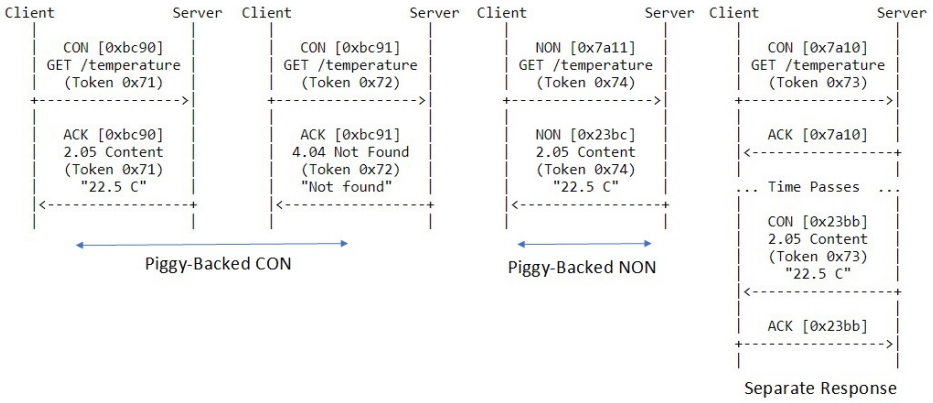


Figura 2.11: *Piggy-backed* e Separate Responde [2]

2.6.3.2 CoAPS - Constrained Application Protocol over DTLS

DTLS é um protocolo padronizado pelo IETF para fornecer comunicação segura para protocolos de transporte de datagramas não confiáveis como o protocolo UDP. Este foi baseado no protocolo *Transport Layer Security (TLS)* e fornece recursos de segurança equivalentes ao mesmo. Porém, o TLS é utilizado como o protocolo TCP e permite uma série de configurações como troca de chaves e criptografia simétrica e assimétrica [47]. Atendendo às limitações dos dispositivos IoT, há uma preferência pelo UDP, causando uma incompatibilidade com o TLS. Assim, o DTLS foi projetado para ser o mais similar possível ao TLS, a fim de aproveitar as infraestruturas e implementações de protocolo preexistente. O DTLS resolve várias incompatibilidades do protocolo TLS executando sobre protocolos não confiáveis [48], sendo que as principais alterações afetam o protocolo de *Handshake*. Através da figura 2.12, podemos observar onde se situa esta camada adicional na pilha protocolar.

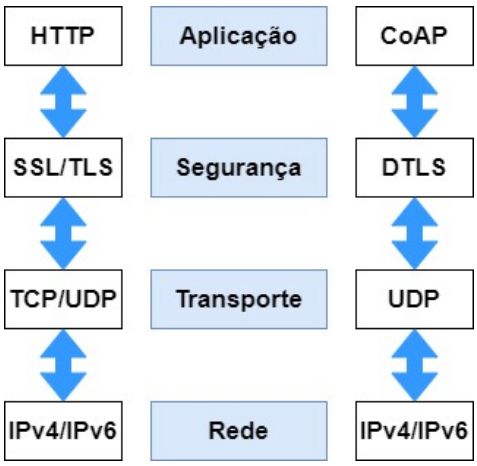


Figura 2.12: Pilha DTLS

HandShake

Devido à natureza do protocolo UDP não estabelecer conexão, o DTLS, ao contrário do TLS, é vulnerável a vários ataques de negação de serviço com endereços IP falsos [48]. Para atenuar essa ameaça, o *handshake* TLS foi estendido com uma técnica de troca de *cookies*. Assim, o *handshake* consiste numa sequência de troca de mensagens entre cliente e servidor para estabelecer uma sessão segura. As etapas do *handshake* são representadas na figura 2.13.

Como se pode observar na figura 2.13, cada etapa é separada em *Flights*. Um *flight* é definido por um conjunto de mensagens que, em caso de perdas de pacote, devem ser retransmitidas juntas, mesmo que sejam transmitidas em pacotes diferentes.

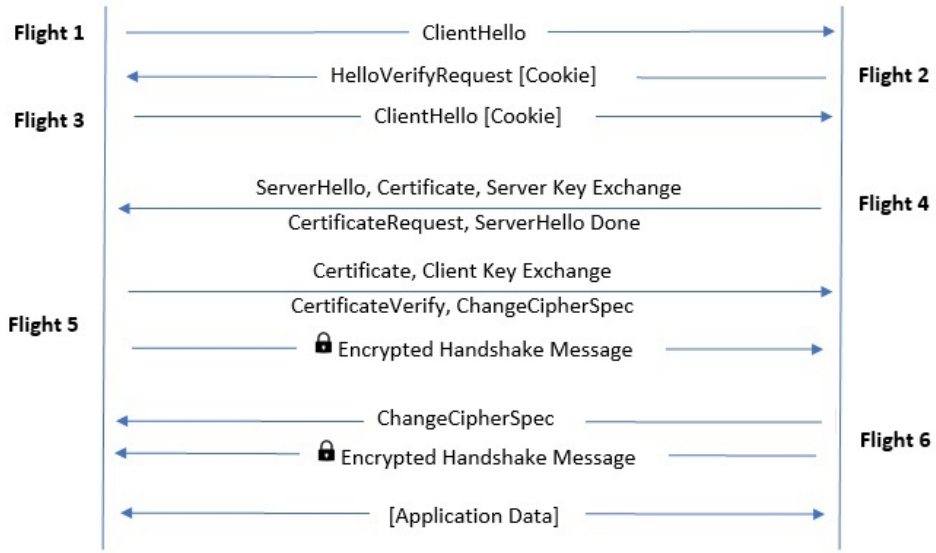


Figura 2.13: Autenticação DTLS

O processo inicia-se com o envio de uma mensagem *Client Hello* do cliente ao servidor e este responde com uma solicitação *HelloVerifyRequest*, no qual contem uma *cookie*. Esta resposta, por parte do servidor, não deve alocar recursos no mesmo de forma a evitar ataques DoS. De seguida, o cliente retransmite outro *ClientHello* juntamente com essa mesma *cookie*. A mensagem *ClientHello* deve conter a versão de protocolo suportada pelo cliente, bem como as cifras que o mesmo suporta. A partir do quarto *flight* é igual ao que ocorre no *Handshake* no TLS, ou seja, negociação de algoritmos para criptografia simétrica e assimétrica, autenticação e *hash* entre cliente e servidor.

2.7 Segurança em Sistemas IoT

A Segurança e a privacidade são o pilar de tudo o que acontece na indústria de *Consumer Electronics (CE)*. Devido à expansão do mercado, a proteção dos dados e do endereço IP, seja ele de uma empresa ou de uma casa doméstica, é fundamental e nos dias de hoje é forçosamente um desafio. Desafio esse que passa por proteger os dispositivos contra ataques de roubo de dados confidenciais de um utilizador, evitar o controlo de forma não autorizada ou ataques *Distributed Denial of Service (DDoS)*. Tradicionalmente, são implementados recursos de segurança, como por exemplo, criptografia, verificação de antivírus e deteção de intrusões nos dispositivos. Na era do IoT, no entanto, a abordagem convencional geralmente tem aplicabilidade limitada devido a muitos dispositivos incluírem recursos de computação limitados e requisitos de ener-

gia restritos [49], originando a que a implementação de recursos de segurança possa ser complexa. Atendendo ao número crescente de dispositivos conectados à Internet, muitos novos protocolos foram desenvolvidos como o CoAP, o RPL e 6LoWPAN.

2.7.1 Segurança na Pilha Protocolar

A pilha protocolar de uma rede IoT é algo complexa. Os vários meios utilizados pelos dispositivos para a comunicação entre si dificultam essa tarefa, acrescentando igualmente as várias restrições ainda inerentes na deteção e os fatores de escala do IoT que normalmente fazem com que a maioria das soluções de comunicações e segurança utilizadas, sejam inadequadas para este tipo de rede.

Tendo em apreciação estes aspetos, o IEEE juntamente com a IETF projetaram novos protocolos de comunicação e segurança que se relevaram fundamentais para a convergência de futuros dispositivos.

Estas soluções padronizadas visam garantir a interoperabilidade com os padrões existentes na Internet, de forma a garantir que os dispositivos possam comunicar com outros dispositivos. Através da tabela 2.3 é pretendido ilustrar um exemplo de uma pilha protocolar para um sistema IoT utilizando WSN. Com a utilização do WSN é adicionado uma camada adicional à pilha protocolar do IoT, nomeadamente a camada de Adaptação. De seguida, é abordado um breve resumo dos mecanismos de segurança que cada camada contem para prevenir ameaças e garantir a segurança em sistemas IoT.

Tabela 2.3: Exemplo de uma pilha protocolar para sistemas IoT utilizando WSN

Camada	Protocolo
Aplicação	CoAP
Transporte	UDP
Rede	IPv6, RPL
Adaptação	6LoWPAN
Física	MAC(IEE 802.15.4, IEE 802.15.4e) PHY (IEE 802.154)

2.7.1.1 Camada Física

As comunicações de baixa energia na camada física PHY e MAC são suportadas pelo IEEE 802.15.4, sendo este responsável por definir as regras para as comunicações nas camadas inferiores da pilha e estabelece o "terreno" para protocolos de comunicação IoT em camadas superiores. Esta camada utiliza o CSMA/CA de forma a evitar colisões , mecanismos para gerir a energia dos dispositivos IoT, utiliza também uma lista *Access Control List (ACL)* onde recebe as *frames* e, para garantir a integridade dessas mesmas *frames* utiliza o *Message Integrity Code (MIC)*. [50]

2.7.1.2 Camada de Adaptação

Ambientes de comunicação de baixa energia utilizam a norma IEEE 802.15.4 para a transmissão de dados em camadas mais altas da pilha. A camada de adaptação 6LoWPAN auxilia a norma IEEE 802.15.4, permitindo a transmissão de pacotes IPv6. O 6LoWPAN também implementa mecanismos de fragmentação [51], reconstrução de pacotes [52] ou técnicas de criptografia, utilizando algoritmos como o Advanced Encryption Standard (AES), o Elliptic Curve Cryptography (ECC), entre outros. [53]

2.7.1.3 Camada de Rede

O encaminhamento em ambientes 6LoWPAN é suportado pelo protocolo RPL [40]. Em vez de ser um protocolo de encaminhamento, o RPL fornece uma estrutura que é adaptável aos requisitos de determinados domínios de aplicação do IoT. Como mecanismos de segurança são utilizados os modos *Preinstalled secure mode (PSM)* em que utiliza as chaves de criptografia simétrica pré-instalada para proteger as mensagens de controlo e, o modo *Authenticated security mode (ASM)* em que utiliza igualmente chaves criptografia simétrica pré-instalada mas agora para os nós se registarem na rede. [54]

2.7.1.4 Camada de Transporte

A camada de transporte garante a confiabilidade dos pacotes. O sensor transmite um pacote para o *gateway* e depois volta a "dormir". Como a transmissão de rede, especialmente para redes de sensores sem fios, é um dos maiores consumidores de energia,

esse padrão resulta em uma economia de energia maior do que se o sensor utiliza-se o protocolo TCP, permanecendo acordado para processar a transmissão. No entanto, o uso de UDP sem retransmissão na camada de transporte reduz significativamente a confiabilidade. Como mecanismos de segurança são utilizados protocolos como o TLS, Secure Socket Layer (SSL) ou DTLS.

2.7.1.5 Camada de Aplicação

A nível de proteção nesta camada, depende um pouco do protocolo aplicacional que se está a utilizar no sistema IoT. O protocolo de aplicação mais utilizado é o CoAP [2], no qual está sendo projetado para fornecer interoperabilidade em conformidade com a arquitetura de transferência de estado representativo da web. Como mecanismos de segurança são utilizados mensagens cifradas através de protocolos como o DTLS

2.7.2 Ataques e Vulnerabilidades

Segundo o glossário de segurança na Internet [55], define a privacidade como "o direito de uma entidade (normalmente uma pessoa), agindo em seu próprio nome, determinar o grau de interação com o ambiente, incluindo o grau em que a entidade está disposta para partilhar informações sobre si mesmo com os outros".

A privacidade dos utilizadores e sua proteção de dados foram identificados como um dos desafios mais importantes que precisam ser abordados nos sistemas IoT. Assim, a segurança é um grande desafio devido à sua complexidade, heterogeneidade e um grande número de recursos conectados entre si. O grau de complexidade é percetível quando, por um lado, existem várias soluções para os demais ataques, mas ao implementar todas essas soluções já existentes num só dispositivo, atendendo aos requisitos atuais (baixo consumo de energia, algoritmos otimizados, etc.), criaria muita sobrecarga e reduziria seu desempenho. Por outro lado, implementar uma solução para mitigar um só ataque, não significa que o mesmo não seja vulnerável na mesma. Um atacante, após conseguir penetrar através de uma vulnerabilidade, pode executar um ataque no dispositivo IoT infetado ou adulterando algum nó, ou seja, o ataque pode surgir por uma vulnerabilidade física ou de dentro da sua rede aproveitando falhas no protocolo de encaminhamento, ou utilizando aplicações mal-intencionadas, de forma a quebrar a criptografia. Por exemplo, um atacante pode reprogramar um sensor de temperatura de tal forma que envia dados, não apenas para um servidor legítimo, mas também para

o servidor do atacante. O consumir do ataque pode implicar danos ou ações ilegais.

2.7.2.1 Ataques - Camada de Perceção

Como os principais objetivos desta camada são a recolha de dados e a atuação, muitos ataques visam esta camada pois todas as funcionalidades da camada superior dependem dela. Os intrusos podem realizar ataques "não-técnicos", como destruir sensores, ou realizar ataques "técnicos", como adulteração de dispositivos. Em geral, os ataques mais comuns são:

- ***Jamming***

  É um tipo de ataque ativo em que é responsável pela modificação do fluxo de dados ou pela criação de fluxo de dados falsos, interrompendo a disponibilidade no meio de comunicação. Esta comunicação é realizada através de interferências utilizando frequências de rádio. A perda de alguma mensagem crucial, pode destruir todo o sistema. [56]

- **Adulteração de dispositivos**

  Este ataque tem como finalidade obter acesso físico ao nó para conseguir aceder ou extrair informações confidenciais, como chaves de criptografia ou outros dados no nó. Como o atacante pode aceder as vezes que pretender ao dispositivo, este pode ser substituído ou modificado, portanto o controlo total do nó vai para o atacante.

- **Duplicação de Fragmentos**

  Este ataque beneficia do facto de um recetor não conseguir verificar nesta camada se um fragmento é legítimo ou duplicado. Assim, o atacante pode explorar esse facto para bloquear seletivamente a reconstituição de pacotes fragmentados específicos em um nó de destino, como por exemplo, impedir a comunicação segura bloqueando os pacotes do protocolo DTLS.

- **Reserva de *Buffer***

  Este ataque visa atingir a escassa memória dos nós com recursos limitados e, alavanca o fato de que o recetor de um pacote fragmentado não pode determinar à *priori* se todos os fragmentos serão recebidos corretamente. Portanto, um nó recetor deve reservar espaço de forma a reconstruir do pacote completo, conforme indicado no cabeçalho 6LoWPAN. Outros pacotes fragmentados são descartados pelo destinatário se o *buffer* de reconstrução já estiver ocupado.

- **Injeção de código malicioso**

  Atacante injeta código malicioso em alguns pacotes para roubar ou modificar dados confidenciais, de forma a conseguir acessos indevidos ou controlo total sobre o sistema IoT.

2.7.2.2 Ataques - Camada de Rede e Transporte

Nesta camada é onde surgem os principais ataques aos dispositivos de IoT. Muitos destes ataques visam obter acesso indevido aos dispositivos, negação de serviços, obtenção de dados, entre outros. Dos vários ataques praticados nesta camada, destacam-se:

- ***Sinkhole***

  Um nó malicioso anuncia uma rota falsa para atrair nós com o intuito de redirecionar seus pacotes através dele. Apesar de não perturbar a rede, pode ser perigoso se for articulado com outro ataque; [57]

- ***Wormhole***

  Neste ataque, o nó adversário cria um túnel virtual entre duas extremidades. Este nó atua como um nó de encaminhamento entre dois nós reais. Os dois nós maliciosos geralmente afirmam que estão a um salto da estação base. O nó atacante também pode ser utilizado para convencer dois nós distintos de que eles são os vizinhos, retransmitindo pacotes entre deles. [57, 58]

- **Encaminhamento Seletivo**

  Neste ataque, o nó malicioso atua como um nó normal, mas descarta seletivamente alguns pacotes. O ataque *black hole* é a forma mais simples de ataque de encaminhamento seletivo, no qual todos os pacotes são descartados pelo nó malicioso.

- ***Sybil***

  Dispositivo malicioso que utiliza de forma ilegítima inúmeras identidades como se fossem sua na mesma rede. Este ataque é projetado para superar o objetivo principal das técnicas de redundância no armazenamento de dados dispersos. Além disso, pode ser utilizado para atacar algoritmos de encaminhamento;

- **Ataque de identidade**

  Combinação de ataques de *spoofing* e *Sybil*. Um atacante pode obter acesso ilegal a pacotes destinados a um nó específico clonando sua identidade;

- ***Hello Flood***

  Em uma rede de sensores, o protocolo de encaminhamento transmite uma mensagem de saudação para anunciar sua presença aos vizinhos. Um nó que recebe a *hello message* pode assumir que o nó de origem está dentro do seu alcance de comunicação e adicionar esse nó de origem à sua lista de vizinhos. [58];

- **Negação de Serviço (DoS)**

  Este ataque visa negar a disponibilidade da utilização normal da rede ou da administração da rede, com ou sem fios. Este tipo de ataque pode ser despoletado através de esquemas de ataques como o UDP *flood*, ICMP *flood*, SYN *flood* ou o *ping of death*.

2.7.2.3 Ataques - Camada de Aplicação

Atualmente, as aplicações raramente são desenvolvidas para operar num modo independente. Cada dispositivo está conectado a outros dispositivos que podem causar danos, tornando-os vulneráveis a muitos ataques. Alguns dos ataques nesta camada são:

- **Exploração de uma configuração incorreta**

  Em alguns casos, vários componentes, como sistemas operativos, base de dados e/ou servidores, podem ser utilizados para oferecer suporte a dispositivos IoT em execução. Assim, a configuração incorreta de tais componentes pode levar a problemas de segurança nesses dispositivos;

- **Reprogramação de ataques**

  Permite a reprogramação de dispositivos IoT remotamente. Uma vez que o processo de programação não está protegido, o atacante pode obter este procedimento para controlar uma grande parte da rede;

- ***Phishing***

  Este ataque é a forma mais simples de um ciberataque e, ao mesmo tempo, o ataque mais perigoso e eficaz. Através deste ataque, o atacante tenta obter dados confidenciais dos utilizadores, como credenciais de acesso ou basicamente tudo o que possa trazer benefícios ao atacante.

- **Virus e *Worms***

  Os vírus e os *worms* são outros grandes desafios para os serviços e aplicações IoT. Estes ataques podem-se propagar pelo equipamento sem serem detetados tentando garantir acesso a dados confidenciais que podem ser depois consultados, modificados ou eliminados pelo atacante.

2.7.3 Contra-medidas

As contra-medidas são um ponto fulcral para o bom funcionamento quer do dispositivo quer do sistema IoT. É essencial nos dias de hoje existir uma preocupação adicional com este tipo de sistemas. Muitos investigadores já tentaram solucionar este problema com contra-medidas, mas ainda não há soluções definitivas para todos os ataques à qual os sistemas IoT estão suscetíveis.

Os aspetos de segurança convencionais encontram-se divididas em três categorias principais, conhecidas como tríade da CIA: confidencialidade, integridade e disponibilidade. Apesar da popularidade da tríade da CIA [59], foi demonstrado que a mesma falha ao lidar com novas ameaças. Assim, de forma a tentar preencher esta lacuna, foram adotados mais requisitos de segurança no qual fornecem um conjunto mais abrangente, investigando um grande número de sistemas de informação em termos de segurança e garantia. Na Tabela 2.4 são descritos os objetivos de segurança propostos com os novos requisitos de segurança.

Tabela 2.4: Requisitos de Segurança

Requisitos de segurança	Definição
Confidencialidade	Apenas utilizadores legítimos podem obter acesso aos dados
Integridade	Garante que a informação manipulada mantém todas as características originais estabelecidas pelo proprietário
Não Repudio	Não permite a negação, por parte do utilizador, do envio de determinada informação
Disponibilidade	Garantia que os serviços estão sempre acessíveis para os utilizadores legítimos
Privacidade	Informação fornecida apenas ao proprietário da informação
Confiabilidade	Capacidade de provar uma identidade e certificar a confiança numa terceira entidade
Auditabilidade	Capacidade para realizar uma monitorização das ações num sistema de informação
Prestação de contas	Processo no qual responsabiliza todo indivíduo que trabalha com um sistema de informação quanto à garantia da informação

De seguida será abordado outras medidas convencionais que podem ser aplicadas.

2.7.3.1 *Firewall*

É um dispositivo de segurança que monitoriza o tráfego de entrada e saída da rede e determina se deve permitir ou bloquear o tráfego, com base em um conjunto de regras de segurança previamente definidos. As *firewall* são a primeira linha de defesa e constituem uma barreira entre redes internas seguras e controladas e a Internet. A *firewall* é vista como um mecanismo de defesa contra acessos não autorizados de e para uma rede privada. As *firewall* podem ser implementadas em *hardware* e *software* ou na combinação de ambos [60]. Graças a evolução de novos métodos de ataques e ameaças, as *firewall* evoluíram e hoje têm a capacidade para reconhecer assinaturas de aplicações ou serviços através de padrões de tráfego, analises de *payload* mediante inspeção de pacotes, entre outros.

2.7.3.2 Cifragem

A cifragem dos dados que circulam através de um canal de comunicação, significa torná-los ininteligíveis para quem não tem autorização de acesso aos mesmos. A cifragem tem como objetivo principal realizar uma comunicação segura, garantido os serviços de confidencialidade, segurança e privacidade dos dados. A utilização de protocolos seguros como o TLS ou o DTLS, entre outros, auxiliam a cifragem de canal ponto-a-ponto. Outra forma de auxílio passa pela utilização de chaves simétricas e/ou chaves assimétricas. Caso a cifragem seja bem sucedida, os dados tornam-se, assim, ininteligíveis em casos de ataques de *ransomware* ou de outros tipos de violações de segurança. Contudo é necessário ter em consideração os recursos limitados dos dispositivos de IoT para a utilização de algoritmos de criptografia. Os mecanismos utilizados podem ser, por exemplo, através de modos se segurança como a utilização de *PreSharedKey*, *RawPublicKey*, *RawPublicKey*, certificados, entre outros.

2.7.3.3 Mecanismos de autenticação e autorização

Com a ubiquidade dos sistemas computacionais, surgem diversos problemas de segurança, que começam com roubo de credenciais, interrupção de serviços e até roubo de identidade, no qual um atacante se faz passar por utilizador legítimo para escalar acessos. Assim, os mecanismos de autenticação e autorização visam garantir a identificação de um utilizador ou de processos. A autenticação de um utilizador ou processo desempenha um papel importante para o mecanismo de controlo de acesso, uma vez

que suas operações são realizadas com base em uma entidade autêntica. Os mecanismos utilizados podem ser, por exemplo, através de assinaturas digitais, *Public Key Infrastructure (PKI)* ou *Key Distribution Center (KDC)*. Estes mecanismos, adotando estratégias mais leves computacionalmente, permitem maior controlo na proteção de sistemas contra os ataques de falsificação, adulteração, interceção, elevação de privilégios, entre outros.

2.7.3.4 Auditoria

A auditoria visa verificar a conformidade do próprio ambiente informatizado, garantindo a integridade dos dados manipulados na rede, ou seja, ao auditar e registar as atividades permite detetar acessos indevidos ou transações anómalas. Esses registos de acessos podem ser obtidos, por exemplo, através de ficheiros *log*. A auditoria auxilia no combate a ataques de força bruta, repúdio e o escalar de privilégios.

2.7.3.5 IDS

O *Intrusion Detection System* (IDS) é um mecanismo de segurança que monitoriza o tráfego da rede em busca de atividades maliciosas (ataques internos e externos) e alerta o administrador da rede quando atividades maliciosas são detetadas. O IDS realiza uma monitorização passiva, pois apenas pode detetar as atividades maliciosas, mas não pode evitá-las. A implementação de um IDS pode ser por *software* e/ou *hardware*. As intrusões podem ser detetadas através de assinaturas conhecidas de ataques, classificação de tráfego normal ou anómalo, no qual são previamente especificados através de regras especificas. [61] O Snort é um exemplo de um IDS.

2.7.3.6 IPS

O Intrusion Prevention System (IPS) é um mecanismo complementar ao IDS e têm como função intercetar intrusões. Ao invés do IDS, que apenas monitoriza a rede e reporta caso detete alguma anomalia, o IPS atua executando ações/medidas, de forma a suspender o processo ou comunicação maliciosa. A *firewall* é um exemplo de um IPS.

2.8 Síntese

Neste capítulo foi efetuado um levantamento do estado do conhecimento científico sobre a área do IoT e questões relacionas à segurança do mesmo.

Pretendeu-se também, neste capítulo, dar a conhecer todo o conceito que envolve o paradigma IoT, nomeadamente a sua visão, elementos constituintes e as várias tecnologias de comunicação que podem ser utilizadas nos sistemas IoT. Desta forma, os dispositivos de IoT são um enorme desafio para os investigadores desta tecnologia devido à elevada heterogeneidade existente.

Devido a esta heterogeneidade e às limitações inerentes neste tipo de dispositivos, sendo estes classificados como dispositivos de baixos recursos computacionais e energéticos, necessitando de uma arquitetura flexível em camadas, no qual ainda não existe um modelo claro de referência que seja consensual sobre qual a arquitetura mais indicada para sistemas IoT.

Foram também desenvolvidos vários protocolos com o intuito de auxiliar não só na comunicação mas também na segurança. São preferencialmente utilizados protocolos leves, como o protocolo aplicacional CoAP, visto ser um protocolo direcionado para dispositivos com limitações onde não possa existir sobrecarga.

Devido à expansão do mercado e à crescente utilização dos dispositivos IoT, a segurança dos sistemas IoT e consequentemente, a segurança dos dados dos utilizadores, são um dos desafios que a comunidade científica tem tentado solucionar. De forma a garantir a segurança foram aplicados mecanismos pelas várias camadas tentando evitar ataques, deste a camada de perceção até à camada de aplicação. Foram também implementadas contra-medidas convencionais, como a criptografia ou autenticação e autorização, de forma a responder aos diversos ataques existentes. Contudo, estas contra-medidas não são suficientes para responder aos ataques que surgem diariamente, pelo que será necessária a utilização de mecanismos e soluções de segurança desenvolvidos e concebidos especificamente para sistemas IoT, de forma a respeitar a sua heterogeneidade, interoperabilidade e as suas gritantes limitações em termos de recursos energéticos e computacionais.

Uma das soluções que tem vindo a ser proposta pela comunidade científica tem sido a utilização de sistemas de deteção de intrusão. Assim, no capítulo seguinte será abordado as caraterísticas, seus constituintes e o modo como se pode classificar um

IDS. Para além disso, será apresentado de uma forma resumida as principais soluções de IDS desenvolvidas especificamente para os sistemas IoT, indicando quais as suas finalidades e vantagens.

Capítulo 3

IDS

Os sistemas de deteção de intrusão servem como uma forte linha de defesa para as redes de comunicações contra os atacantes. Sem um IDS, uma rede fica mais fragilizada na vertente de segurança. Apesar de existir soluções baseadas em cifras, os ataques ainda são possíveis, sendo o IDS uma ferramenta adicional de segurança essencial, pois monitoriza e analisa o tráfego, dados, comportamento ou recursos e tenta proteger a rede dos atacantes.

3.1 Definição

Atualmente, existem cada vez mais pessoas a utilizarem a Internet, quer seja por motivos pessoais ou comerciais, tornando a segurança uma prioridade. Devido a este impacto, atacantes têm realizado várias tentativas para derrubar redes de grandes companhias e serviços *web*. Muitos métodos têm sido desenvolvidos para manter a segurança da infraestrutura e comunicação de redes pela Internet, entre eles, o uso de *firewall*, encriptação de mensagens e criação de redes privadas virtuais.

O termo sistema de deteção de intrusão foi utilizado pela primeira vez por James Anderson no início dos anos 80 [62]. O autor introduziu o conceito de deteção de uso indevido e forneceu as diretivas básicas para um futuro projeto e desenvolvimento de um sistema de deteção de intrusões (*IDS*). A deteção de intrusão é um processo de identificação de uso não autorizado ou ataque, a um computador ou a uma rede.

Um IDS pode ser definido como uma ferramenta com a finalidade de ajudar a deter ou mitigar danos, procurando identificar atividades suspeitas, maliciosas e/ou violação de políticas. Os IDS permitem também detetar tentativas que possam comprometer a

confidencialidade, integração e disponibilidade. Para tal, recolhe dados de diferentes fontes, como por exemplo, *logs* da rede, do sistema, entre outros. De seguida, analisa-os para identificar possíveis violações de segurança e, por fim, caso algum destes fatores seja detetado é reportado ao administrador para que as ações necessárias possam ser tomadas.

Embora os IDS monitorizem o tráfego quanto a atividades potencialmente maliciosas, eles também estão sujeitos a criar falsos alarmes. As intrusões, ou tentativas, podem ter origem proveniente da Internet, bem como ter origem na rede interna, por meio de utilizadores legítimos dos sistemas que podem, intencionalmente ou não, causar algum dano aos sistemas da organização em questão.

De salientar que o principal foco de um IDS não é prevenir qualquer tipo de intrusão, mas sim em detetar essa intrusão ou indícios que evidenciem uma tentativa de intrusão. Outro dado importante a reter é que existem diferentes técnicas para a deteção e cada uma dessas técnicas traz diferentes tipos de benefícios que são aplicáveis numa grande variedade de ambientes sendo, portanto, necessário selecionar aquele tipo que mais se possa adequar às necessidades do ambiente a ser protegido. A forma mais correta de definir o tipo a ser utilizado consiste num conhecimento pleno de quais são requisitos específicos do sistema para, então, executar a implementação de um IDS.

Na tabela 3.1 são enumeradas algumas vantagens e desvantagens de um IDS. Como vantagens pode-se destacar a sua utilização como um complemento à segurança numa rede, auxiliando na monitorização e deteção de ataques. Por outro lado, uma inadequada implementação de um IDS pode causar análises incorretas, criando assim falsos positivos e negativos, no qual podem ser prejudiciais para a segurança da rede e, consequentemente incorreto funcionamento da mesma.

Tabela 3.1: Vantagens e Desvantagens de um IDS

IDS	
Vantagens	- Útil como complemento aos mecanismos preventivos - Fornece alertas de que o sistema está sob ataque - Auxilio na análise de *logs*
Desvantagens	- Falsos Positivos - Falsos Negativos - Vulnerável a novos ataques

Através da figura 3.1 é demonstrado como se pode classificar um IDS em relação à sua Arquitetura, Localização, Deteção e Fonte de Dados. Nas próximas sub-secções será abordado cada um destes temas em específico.

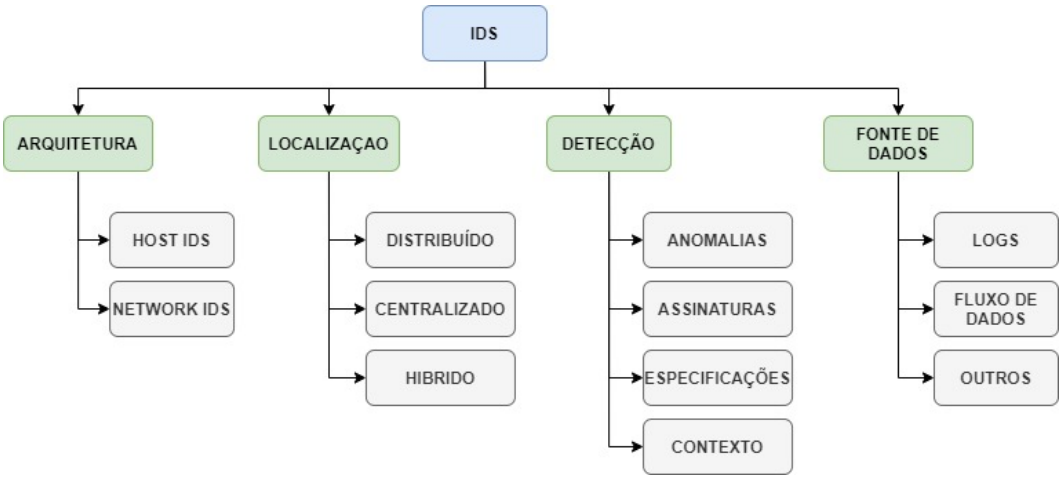


Figura 3.1: Identificação de um IDS

3.2 Arquitetura

Uma forma de determinar a arquitetura de IDS que se deve utilizar passa pela compreensão do que os mesmos monitorizam. Como se pode observar na figura 3.1, os IDS estão divididos em dois modelos, nos quais têm sido utilizados por administradores de sistemas e de redes em geral, de acordo com as necessidades definidas pelas políticas de segurança adotadas.

- **Baseado em um *Host* (*Host-based Intrusion Detection System (HIDS)*)**

  Os HIDS são executados em todos os computadores ou dispositivos da rede com acesso direto à Internet e à rede interna. Ele monitoriza apenas os pacotes de entrada e saída do dispositivo e alerta o administrador se forem detetadas atividades suspeitas ou mal intencionadas. Os recursos do HIDS incluem verificação de integridade, correlação de eventos, análise de *logs*, imposição de políticas, processador, memória, entre outros [63]. O HIDS tende a ser mais preciso e com menos falsos positivos do que o NIDS, porque analisa os ficheiros de *log* e, como resultado, pode determinar se um ataque foi bem-sucedido ou não [64]. Contudo, para analisar uma grande quantidade de dados de forma a distinguir entre processos normais de maliciosos, é necessário muito tempo de computação e muitos recursos.

- **Baseado em uma Rede (*Network-based Intrusion Detection System (NIDS)*)**

  O NIDS são implementados num ponto ou vários pontos estratégicos da rede para monitorizar e analisar o tráfego de entrada e saída dos dispositivos da rede para

deteção de atividades suspeitas. Os NIDS são utilizados na análise ao nível de pacotes ou fluxos, permitindo uma visão global sobre a rede. Entretanto com o aumento da velocidade, número de redes e tipos de ataques, os NIDS existentes enfrentam desafios ao nível da captura de pacotes ou fluxos, tornando a sua tarefa mais complicada. Além disso, estes sistemas não possuem as informações sobre os consumos de recursos individuais ou *logs* na rede, o que pode ser crucial para a deteção de ataques específicos.

Na tabela 3.2 são enumeradas algumas vantagens na escolha do tipo de arquitetura para um IDS. Como vantagens na escolha por uma arquitetura HIDS, destaca-se o facto da mesma auxiliar na deteção de ataques, como por exemplo, ataques de *Trojans* e, o facto de serem implementados em dispositivos, ao invés de uma arquitetura NIDS que é implementada em pontos estratégicos na rede, são mais eficazes na analise com tráfego cifrado. Por outro lado, uma arquitetura NIDS tem como vantagens a deteção e resposta em tempo real, a sua escalabilidade e o facto de não ter impacto no desempenho da rede.

Tabela 3.2: Vantagens das Arquiteturas de um IDS

Arquitetura	Vantagens
HIDS	- Eficaz em tráfego criptografado - Não requer *hardware* adicional - Ajuda na deteção de *Trojans* - Geram menos falso positivos - Independentes da topologia da rede
NIDS	- Escalabilidade - Custo-benefício - Fácil adição de um novo *host* - Deteção e resposta em tempo real - Não causam impacto no desempenho da rede - Identificação de erros na camada de rede

3.3 Estratégias de Localização

Uma boa estratégia de localização para um IDS é essencial, pois devido às caraterísticas de cada uma das estratégias, a escolha adotada pode ser vantajosa perante um bom planeamento. Como se pode observar na figura 3.1, um IDS em termos de estratégias de localização pode ser:

- **Distribuído**

  Um IDS distribuído (*Distributed IDS (dIDS)*) consiste em vários sistemas de IDS numa grande rede, onde todos comunicam entre si ou com um servidor central que facilita a monitorização avançada da rede, a análise de incidentes e dados de ataques instantâneos. Ao distribuir estes agentes cooperativos numa rede, a deteção de intrusões é executada localmente em todos os nós da rede. Para garantir um IDS em cada nó, a equipa de segurança necessita de adaptar a técnica ou o algoritmo de deteção de acordo com os recursos disponíveis, permitindo que uma empresa possa gerir com eficiência os recursos para análise de incidentes. Esta abordagem claramente não possui nenhuma sobrecarga de comunicação, no entanto, o IDS apenas possuí informações locais para analisar, a fim de detetar atacantes. [4]

- **Centralizado**

  Os IDS centralizados colocam a deteção de intrusão num ponto central e todas as informações de monitorização devem ser recolhidas aqui. Um dos principais motivos para selecionar um ponto central para a deteção de intrusões passa pelos recursos disponíveis. No entanto, os sistemas centralizados acompanham a sobrecarga de comunicação, pois os dados de monitorização precisam de ser transportados até o ponto central. Se nós mal intencionados impedirem que os dados de monitorização cheguem ao IDS centralizado, eles podem induzir o IDS em erro.

- **Híbrido**

  Os IDS híbridos harmonizam as duas arquiteturas de deteção anteriormente abordadas tentando beneficiar do melhor de cada uma delas. Este tipo de IDS é dividido em módulos nos quais são distribuídos pela rede. Esses módulos permitem aplicar a deteção de intrusões até certo ponto, compartilhar menos informações com o módulo centralizado e, assim, reduzir a sobrecarga de comunicação e aproveitar os recursos do módulo centralizado.

Na tabela 3.3 são enumeradas algumas vantagens na escolha do tipo de localização para um IDS. Como vantagens na escolha por uma localização distribuída, destaca-se a sua flexibilidade e maior facilidade de monitorização e análise. Para uma localização centralizada, a grande vantagem reside no facto de todas as atividades serem controladas por um único ponto central ao invés, da localização distribuída onde a deteção de intrusões é executada localmente. Por último, para um localização híbrida, destaca-se o facto desta combinar características das localizações distribuídas e centralizadas.

Tabela 3.3: Vantagens dos conceitos de Localização para um IDS

Localização	Vantagens
Distribuído	- Flexibilidade - Escalabilidade - Redução de custos computacionais - Monitorização e análise mais fácil e rápido - Tolerância a falhas
Centralizado	- Custos de manutenção mais baixos - Todas as atividades são controladas por um único ponto central - Instalação num dispositivo local ou remoto
Híbrido	- Combinação das vantagens dos IDS distribuídos e Centralizados

3.4 Metodologias de Deteção

Um IDS pode classificar o tráfego (pacote ou fluxo) como normal ou anómalo, onde o padrão de anomalia provavelmente será um ataque. No entanto, há ocasiões em que o uso legítimo dos recursos da rede pode levar a um resultado de classificação positivo para a deteção de intrusões baseada em assinatura ou anomalia. Como resultado dessa classificação incorreta, o IDS sinalizará um falso alarme. Este é um problema comum com os IDS e é chamado de falso positivo. Um dos parâmetros para medir a qualidade de um IDS é o número de falsos positivos. Quanto menor o número de falsos positivos, melhor o IDS. Como se pode observar na figura 3.1 os IDS em termos de estratégias de deteção podem ser:

- **Baseado em Anomalias**

  Os sistemas baseados em anomalias monitorizam o tráfego da rede de forma a aprendem o comportamento do sistema, ou seja, aprende o que será considerado como tráfego "normal", criando posteriormente um perfil. Desvios de perfil mostram possíveis anomalias. Este tipo de deteção permite detetar novos ataques, pois é expectável que os ataques originem desvios em relação ao comportamento dito "normal". No entanto, eles podem igualmente criar falsos alarmes e classificar incorretamente conexões legítimas como tentativas de invasão. Além disso, acredita-se que as técnicas baseadas em anomalias sejam mais complexas e usem mais recursos do que as outras técnicas de deteção.

- **Baseado em Assinaturas**

  Os sistemas baseados em assinaturas procuram por atividades que correspondam a um conjunto predefinido de eventos que descrevem exclusivamente um ataque conhecido. Estes IDS devem ser programados especificamente para detetar cada ataque conhecido. Assim pode detetar facilmente os ataques cujo padrão (assinatura) já existe no sistema, mas é bastante difícil detetar novos ataques de *malware*, pois seu padrão (assinatura) não é conhecido. Apesar disso, esta técnica é extremamente eficaz e é o principal método utilizado em produtos comerciais para detetar ataques.

- **Baseado em Especificações**

  Os sistemas baseados em especificações detetam invasões quando o comportamento da rede se desvia do comportamento expectável. Para detetarem invasões este tipo de sistema rege-se através de um conjunto de regras com base nas especificações previamente incorporadas pelo administrador da rede. Este tipo de sistema é idêntico ao sistema baseado em anomalias, pois tenta identificar desvios de comportamento, embora tenha que ser o administrador da rede a definir manualmente as regras de cada especificação. No entanto essas especificações podem não se adaptar a ambientes diferentes podendo em suma, consumir mais tempo e recursos, logo ficam mais suscetíveis a eventuais erros. Por outro lado, as especificações definidas manualmente geralmente fornecem taxas mais baixas de falsos positivos em comparação com a deteção baseada em anomalias.

- **Híbrido**

  Os sistemas híbridos passa pela utilização de duas ou mais técnicas de deteção, tentado tirar o maior partido das caraterísticas de todos. É claro que essa decisão pode ser cara em termos de recursos disponíveis e tem de ser bem ponderada.

Na tabela 3.4 são enumeradas algumas vantagens na escolha do tipo de deteção para um IDS. Como vantagens na escolha por uma deteção baseada em anomalias, destaca-se a sua eficiência para a deteção de novos ataques ou intrusões, embora detete que há algo de errado mas não consegue especificar. Para uma deteção baseada em assinaturas, destaca-se a sua eficiência na deteção de ameaças previamente conhecidas. Em relação à deteção baseada em especificações, destaca-se a baixa taxa de falsos positivos , não necessitando de uma fase de aprendizagem como os métodos anteriores. Por último, na deteção híbrida, destaca-se o facto desta combinar duas ou mais características das deteções anteriores.

Tabela 3.4: Vantagens dos conceitos de Deteção para um IDS

Localização	Vantagens
Anomalia	- Deteção de comportamento dos utilizadores - Eficiente na deteção de novos ataques ou intrusões - Gera informações que podem ser utilizadas para definir assinaturas
Assinatura	- Mais eficiente que a deteção por anomalia - Eficácia na deteção de ameaças previamente conhecidas
Especificação	- Taxas de falsos positivos baixos comparando com as abordagens de deteção anteriores - Não necessita de uma fase de aprendizagem
Híbrido	- Combina das vantagens das abordagens anteriores

3.5 Fonte de Dados

Para proteger a rede de atividades maliciosas externas podem ser utilizados IDS para monitorizar e validar a integridade da infraestrutura da rede. No entanto, para proteger as redes contra ameaças, como ataques DoS e surtos de *worms*, os IDS necessitam de ter uma fonte de dados onde possam executar as suas análises. Nesta secção serão abordados dois tipos de fontes de dados para IDS, *Packet-based* e *Flow-based*. Como se pode observar pela figura 3.1, podemos classificar a fonte de dados utilizando dois métodos: baseados em pacotes e baseados em fluxos.

- **Baseado em pacotes (*Packet-based*):** O sistema baseado em pacotes inspeciona todo o conteúdo dos pacotes, incluindo a *payload* dos mesmos. Tendo em conta este facto não podemos considerar que exista uma total confidencialidade para o utilizador. Este método produz baixos falsos alarmes mas consome mais processamento, memória e tempo comparando com o método baseado em fluxos.

- **Baseado em fluxos (*Flow-based*):**

  A deteção de anomalias baseada em fluxo é centrada no conceito de fluxo da rede, ou seja, em vez de olhar para todos os pacotes que atravessam a rede, como no *Packet-based*, este apenas analisa as informações agregadas dos pacotes, de modo a que, quantidade de dados a serem analisados seja reduzida. Portanto, os fluxos fornecem informações e padrões sobre a conexão de rede (podem ser representados em apenas alguns bytes) para serem analisados em vez do *payload* do pacote. Um fluxo pode conter vários dados, tais como, endereços IP, portas de rede, protocolos, quantidade de *bytes* enviados e recebidos, entre outros. Posteriormente os fluxos são agrupados e exportados para análise. De salientar que as

abordagens baseadas em fluxos dependem muito da capacidade dos dispositivos para gerar informações de fluxo.

Na tabela 3.5 são enumeradas algumas vantagens na escolha do tipo de fonte da dados para um IDS. Como vantagens na escolha por uma fonte de dados baseada em pacotes, destaca-se uma maior eficiência na deteção de ataques e uma menor taxa de falsos positivos, embora necessite de um maior puder computacional. No caso de uma escolha por uma fonte de dados baseada em fluxos, destaca-se o menor custo computacional comparado com o tipo de fonte de dados baseado em pacotes e uma maior privacidade dos dados para o utilizador.

Tabela 3.5: Vantagens dos conceitos de Fonte de Dados para um IDS

Localização	Vantagens
Baseado em Pacotes	- Menor taxa de falsos positivos - Maior eficácia na deteção de ataques
Baseado em Fluxos	- Menor custo computacional - Independência do *payload* - Apropriado para comunicações cifradas - Maior privacidade do utilizador

- Monitorização de Fluxos

O enorme crescimento da utilização dos dispositivos de IoT levou a um extraordinário aumento na taxa de transferência de dados e poder computacional, o que resultou em grandes desafios no qual exigem soluções alternativas.

A monitorização de fluxos visa apenas a captura de fluxos onde apenas obtém algumas informações sobre os pacotes, ao invés da captura de pacotes que passa por uma captura de um pacote na integra. Essas informações traduzem-se em endereços IP de origem e destino ou tamanho dos pacotes em *bytes*. Segundo o IETF, a definição de um fluxo consiste em - "Um fluxo é definido como um conjunto de pacotes ou quadros que passam por um ponto de observação na rede durante um determinado intervalo de tempo. Todos os pacotes pertencentes a um fluxo específico têm um conjunto de propriedades comuns."[65] A arquitetura típica da monitorização de fluxos consiste em várias fases, como se pode observar pela figura 3.2, adaptado de [66].

A primeira fase é o ponto de Observação de pacotes ou *Packet Observation* [66]. Nesta primeira fase, os pacotes são capturados e pré-processados a partir de um ponto de observação, que pode ser uma interface de rede ou em algum local onde o tráfego

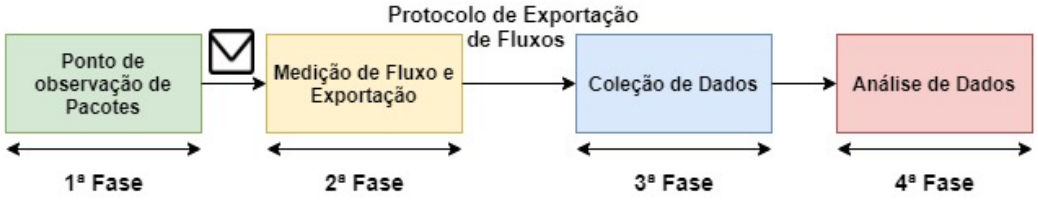


Figura 3.2: Arquitetura *flow monitoring*

passe. Esta mesma fase pode ser sub-dividida em outros cinco processos, captura de pacotes, *Timestamping*, agregação, amostragem de pacotes e, por último, filtragem de pacotes. O primeiro processo pode ser realizado através de uma máquina virtual ou um processo com o intuito de capturar o tráfego. Esta captura será o primeiro elo de ligação com o tráfego gerado. Aqui são também realizadas verificações de erros antes de serem armazenados nos *buffers*. No *Timestamping* os pacotes são registados com data e hora, sendo este um processo essencial, devido ao facto de quando existem pacotes de vários pontos de observação, estes devem ser agrupados num único conjunto de dados, sendo ordenados com base na data e hora. Os restantes processos são opcionais mas têm como finalidade a redução da quantidade de dados, ciclos, memória e largura de banda a serem enviados para a próxima fase. De salientar que a captura dos pacotes pode ser em linha (*in-line*) ou em espelho (*mirror*). A segunda fase é a medição e exportação de fluxos [66], onde no processo de medição, os pacotes são agregados em fluxos e são posteriormente exportados pelo processo de exportação. A agregação dos pacotes é realizada com base numa lista de Information Elements (IE) [67]. Cada entrada por fluxo é armazenada em *cache* até o fluxo ser finalizado. Após este processo se encontrar finalizado, ele é exportado através de um encapsulamento dos fluxos selecionados nas mensagens IPFIX para a recolha de dados, utilizado protocolos como o TCP, UDP ou *Stream Control Transmission Protocol (SCTP)*. A criação dos fluxos e exportação deve ter em consideração a arquitetura [68], tendo em conta tipos e tamanhos. Exemplos de exportadores *open-source* que suportam IPFIX são o YAF [69] ou o nProbe [70]. A terceira fase é a recolha de dados [66] e é onde o armazenamento e pré-processamento de dados de fluxo gerados pela fase anterior é recebido. As operações comuns de pré-processamento incluem agregação, filtragem e compactação de dados. Contudo, a funcionalidade e o desempenho fornecidos na recolha dos fluxo depende do formato de armazenamento de dados subjacente, pois isso define como e em que velocidade os dados podem ser lidos e gravados, podendo estes terem os seguistes formatos, Volátil e Persistente. O armazenamento volátil é realizado na memória RAM, o que o torna mais rápido, podendo ser útil para processamento de dados ou armazenamento em *cache*. Por outro lado, o armazenamento persistente é utilizado para armazenar dados por mais tempo possuindo uma capacidade maior de armazenamento. Em comparação ao armazenamento volátil, este é significativamente

mais lento. Outro fator importante na recolha de dados é a validação de critérios tais como a performance, formato de armazenamento, recursos do protocolo de exportação, atraso no processamento, duplicação de registos de fluxo e integração com outros sistemas. Exemplos de recetores de fluxos *open-source* que suportam IPFIX são o SiLK [71] ou o nProbe [70]. A fase final é a análise de dados [66], onde é realizado a análise dos fluxos. Esta análise pode ser classificada com base na análise e relatórios de fluxos, deteção de ameaças e monitorização de desempenho. [65] A análise e relatório de fluxos fornece a possibilidade de análise através da navegação e filtragem de dados de fluxo. Fornece igualmente uma visão geral com estatísticas, relatórios e alertas. Como exemplos de aplicações que fornecem essa funcionalidade são NfSen [72], nTopng [73] e SiLK [71]. Através de uma análise com base na deteção de ameaças [9, 10] é possível detetar através dos fluxos qual o dispositivo que comunicou e para quem comunicou, qual o número de pacotes enviados e recebidos, qual o tamanho desses mesmos pacotes, entre outros. Algumas propostas têm sido exploradas dos últimos, conforme pesquisado em [74]. Por último, na monitorização de desempenho é possível observar o *status* dos serviços em execução na rede, as métricas com tempo de resposta, atrasos, perda de pacotes, largura de banda, entre outros.

3.6 IDS para IoT

3.6.1 Caracterização

Com o surgimento da Internet das Coisas, um grande número de objetos físicos na vida quotidiana foi conectado de forma exponencial à Internet. Deste modo, garantir a segurança dos dispositivos não é tarefa fácil e o IoT não ficou de fora dos problemas de segurança e ameaças que colocam em risco os ambientes computacionais. Devido a estes desafios, os métodos que podem identificar pro-ativamente novos ataques são mais adequados para proteger as redes de IoT.

Desta forma, é necessário um IDS que possa detetar novos ataques em ambientes inteligentes tendo em consideração dispositivos com recursos limitados, tanto a nível de poder computacional e de memória, em detrito da redução do desempenho e performance. Outros fatores igualmente cruciais passam pela monitorizam da rede de forma a evitar o incumprimento de regras de segurança, perda de integridade e confidencialidade.

3.6.2 Estudo de soluções existentes de IDS para IOT

Visando expandir os conhecimentos sobre IoT e IDS, esta sessão irá apresentar alguns trabalhos desenvolvidos e apresentados para a comunidade científica, onde será abordado em mais detalhe cada um dos artigos mencionados. Em 2017 Zarpelão [4] e em 2018 Leonel Santos [75] realizaram um estudo sobre as técnicas de IDS existentes para sistemas IoT.

Cho et al. [9] em 2009 propuseram um IDS centralizado utilizando um esquema de deteção de redes de *botnets* com um método de deteção baseado em anomalias. Esta solução tem como finalidade analisar os pacotes que passam pelo encaminhador de periferia. Segundo os autores, esta solução calcula a média de três métricas de forma a criar um perfil de comportamento normal. O sistema ao monitorizar o tráfego de rede, gera um alerta caso as métricas de qualquer nó violem as médias calculadas para o perfil normal.

Misra et al. [12] em 2011 apresentaram um método de deteção baseado em especificações com a finalidade de impedir ataques DDoS sobre *middleware*. Esta solução especifica a capacidade máxima de *middleware* e, quando o número de solicitações é excedido, o sistema gera um alerta. Os autores não revelam qual seria a estratégia ideal para a solução.

Raza et al. [76] em 2013 implementaram o primeiro IDS para sistemas IoT, o SVELTE. Este IDS tinha uma estratégia de posicionamente híbrido e uma deteção baseada em anomalias visando detetar ataques de *sinkhole* e ataques de *spoofing*. O SVELTE, foi implementado para combinar a utilização de módulos de IDS leves em nós com restrições de recursos aliados a módulos de IDS com utilização intensitva em encaminhadores de periferia. Desta forma, segundo o autor, o SVELTE atinge uma taxa de 90% para ataques de *sinkhole* em pequenas redes com perdas e quase 100% em redes sem perdas. Contudo, falha na deteção de ataques de negação de serviço porque o SVELTE utiliza a rede para transmitir informações sobre os ataques e, caso esta esteja indisponível, não consegue detetar.

Murynets e Jover [77] em 2013 propuseram um IDS centralizado baseado em especificações com a finalidade de reconhecer atividades e ataques por SMS (*Short Messaging Service*) em uma combinação de níveis de *cluster* e dispositivos individuais IoT. Este IDS é composto por dois algoritmos que monitorizam o tráfego dos dispositivos na rede. Os algoritmos são baseados em uma análise volumétrica e de contacto. A análise

volumétrica é responsável pela deteção de anomalias enquanto o contacto é responsável por analisar as conexões de cada dispositivo. A combinação dos dois algoritmos auxilia a reconhecer ataques, como por exemplo, ataques de negação de serviço. Segundo os autores, a rede tem de ser robusta de forma a ser uma rede aceitável para a avaliação do seu desempenho. Eles concluem também que a solução possui uma alta precisão de deteção, contudo não é um IDS em tempo real e possui uma implementação complexa.

Kasinathan et al. [6] em 2013 propuseram uma estrutura de segurança para deteção de ataques DoS. A solução proposta emprega uma solução com localização centralizada onde seu principal objetivo é detetar ataques de DoS em redes baseadas em 6LoWPAN. O método monitoriza o tráfego da rede 6LoWPAN e, quando uma anomalia é reconhecida na rede, gera um alarme. As assinaturas são "pré-carregadas" num repositório para que o ataque possa ser detetado. Esta solução peca por ser restrita quando a topologia da rede é dinâmica. Além disso, não está claro como o repositório de assinaturas será atualizado.

Wallgren et al. [78] em 2013 propuseram uma abordagem com localização centralizada na qual o IDS é colocado no encaminhador de fronteira. O objetivo desta abordagem passa pela monitorização interna de forma a detetar ataques no domínio físico através de um algoritmo criado pelos autores. De acordo com o proposto, o encaminhador de fronteira envia solicitações eco ICMPv6 para todos os nós e espera que as respostas detetem ataques ou problemas de disponibilidade. Apesar da solução criar tráfego adicional na rede, os autores demonstraram que não é necessário alocar memória adicional para executar o algoritmo de pulsação e, a sobrecarga de energia era mínima.

Amaral et al. [11] em 2014 apresentaram um IDS com uma estratégia de posicionamento híbrida. Os autores propuseram que um conjunto de nós, previamente selecionados, tenham um IDS hospedado. Estes nós têm como missão a monitorização da rede para posteriormente identificar intrusões, espionando os pacotes trocados em sua área. Estes nós têm o "poder" para determinar se um nó é comprometido ou não, de acordo com um conjunto de regras. Cada um destes nós possui um conjunto específico de regras porque cada componente da rede pode ter um comportamento diferente, segundo os escritores. Como se trata de um IDS com deteção baseada em especificações, quando uma regra é violada, estes nós enviam um alerta para um sistema de gestão de eventos. A grande vantagem desta abordagem consiste em permitir a construção de um conjunto diferente de regras para cada área da rede.

Oh et al. [79] em 2014 propuseram um IDS com localização distribuída aliada a uma deteção baseada em assinaturas, utilizando um novo algoritmo de correspondência de múltiplos padrões para sistemas de segurança incorporados. Os autores propuseram, igualmente duas novas técnicas, deslocamento auxiliar e decisão antecipada. A ideia geral destas duas novas técnicas, passa por mitigar o número de operações desnecessárias existentes aquando comparado entre pacotes com *payload* e assinaturas de ataques, tentando assim diminuir o custo computacional. De forma a testar a veracidade do sistema, foi utilizado o Snort, um IDS de código aberto e o ClamAV, um antivírus de código aberto. Segundo os autores, os métodos propostos reduziram com êxito um grande número de operações desnecessárias tendo inclusive melhorado o seu desempenho, especialmente quando as quantidades de exemplos a testar são enormes. Contudo não é em tempo real.

Cervantes et al. [5] em 2015 propuseram um IDS chamado INTI (***I**nstrusion detection for Si**N**khole attacks over 6LoWPAN for Interne**T** of Th**I**ngs*), no qual combina conceitos de confiança e reputação em um método de deteção baseado em anomalia e especificação de forma a detetar e mitigar ataques. Utiliza uma estrutura hierárquica de nós, tornando-o num sistema com localização distribuída, onde os nós são classificados como líderes, associados ou membros, compondo uma estrutura hierárquica. A função de cada nó pode derivar com o tempo devido à reconfiguração da rede ou a um evento de algum ataque. Posteriormente, cada nó monitoriza um nó superior estimando o seu tráfego de entrada e saída. Quando um ataque é detetado por um nó, ele transmite uma mensagem para alertar os outros nós e isolar o atacante. Segundo os autores, este método tem algumas vantagens, como alta precisão de deteção, baixa taxa de falsos negativos e falsos positivos. No entanto, ele pode detetar apenas um tipo de ataque.

Pongle e Chavan [7] em 2015 propuseram um IDS utilizando uma estratégia de posicionamento híbrida. A técnica proposta utiliza os nós como responsáveis por detetarem alterações na vizinhança e enviar informações sobre os vizinhos para os módulos centralizados. Estes módulos são responsáveis por armazenar e analisar esses dados para detetar invasões e identificar possíveis atacantes. Segundo os resultados demonstrados pelos autores, esta solução será uma solução apropriada para sistemas IoT devido aos seus consumos de energia e memória baixos. Contudo têm como desvantagens uma alta taxa de falsos positivos, baixa precisão de deteção, conseguindo apenas detetar um único ataque.

Summerville et al. [10] em 2015 desenvolveram um IDS com deteção baseada em anomalias para identificar comportamentos normais e anormais. Segundo os autores, os

dispositivos de IoT utilizam poucos e simples protocolos, podendo originar um *payload* de rede semelhante. Com base nessa ideia, e utilizado uma técnica chamada correspondência de padrão de *bits*, permitiu realizar a seleção de recursos. O *payload* da rede são tratados como uma sequência de *bytes*, e a seleção do recurso opera em tuplas de *bytes* sobrepostas, chamadas n-gramas. Uma correspondência entre um padrão de *bits* e um n-grama ocorre quando os *bits* correspondentes correspondem a todas as posições. A afetividade e a capacidade dos detetores para identificar pacotes de anomalias a partir de uma enorme quantidade de ataques e tráfego específico nos dispositivos são altas. Através da avaliação realizada por parte dos autores, os resultados foram satisfatórios, tendo concluído que as taxas de falsos positivos para quatro ataques convencionais foram muito baixas.

Surendar e Umamakeswari [80] em 2016 propuseram um IDS com deteção baseada em especificações visando detetar ataques *sinkhole*. Os nós são agrupados num *cluster*, sendo o nó com probabilidade máxima, eleito o nó líder. Os restantes nós são chamados de nós observadores. Os nós líderes executam o processo de monitorização de nós e identificam se existe descarte de pacotes nos nós adjacentes. Posteriormente essa contagem é comparada com um valor limite para determinar se o nó é malicioso e, caso o seja, isola-o construindo a rede novamente sem esse nó. Os resultados obtidos pelos autores após simulação, revelou que a solução tem como vantagem a alta taxa de entrega de pacotes, baixo consumo de energia e baixa sobrecarga.

Midi et al. [8] em 2017 propuseram o Kalis. O Kalis é a primeira abordagem para a deteção de intrusões para IoT que não visa um protocolo e adapta a sua estratégia de deteção aos recursos específicos existentes na rede. Este IDS tem a capacidade de se adaptar a diferentes ambientes graças ao ser implementado com uma arquitetura NIDS, uma localização híbrida e uma deteção igualmente híbrida, pois é baseada em assinaturas e anomalias. Desta forma, o Kalis tem a capacidade para identificar técnicas de prevenção graças à utilização de funções criptográficas, o que o torna eficaz e eficiente em relação à gestão do consumo de recursos. Segundo o autor, o Kalis tem uma taxa de deteção de 91%, com uma precisão de 100% e uma baixa utilização de processamento, cerca de 0.19%.

A tabela 3.6 sintetiza os trabalhos analisados anteriormente sobre soluções existentes para IoT. Esta síntese engloba as diversas possibilidades de classificação para IDS, tendo em consideração a sua estratégia de posicionamento e método de deteção, as intrusões detetadas e por último, as vantagens e desvantagens de cada solução proposta.

Tabela 3.6: Obras cientificas sobre IDS para IoT

Trabalho	Estratégia de Posicionamento	Deteção	Ameaça de Segurança	Vantagens	Desvantagens
Cho et al. (2009) [9]	Centralizado	Anomalia	Botnet em 6LoWPAN	- Criação de alertas	-
Misra et al. (2011) [12]	-	Especificação	DDoS	- Gestão dos recursos utilizados - Flexível - Pode ser estendido para detetar mais ataques	- Ataques DoS podem afetar o SVELTE
Raza et al. (2013) [76]	Distribuído	Híbrido	Ataques de Encaminhamento	- Baixo consumo de energia -Baixa sobrecarga computacional	- Alta taxa de falsos positivos e negativos com especificação errada
Murynets e Jover (2013) [77]	-	Especificação	Ataques de Encaminhamento	- Alta precisão de deteção	- Implementação complexa - Não é em tempo real
Kasinathan et al. (2013) [6]	Centralizado	Assinatura	DoS IPv6	- IDS em tempo real - Alta Disponibilidade - Redução de falsos alarmes - Escalável	- Alto consumo de recursos - Baixa precisão de deteção - Ataques detetados dependem das regras declaradas
Wallgren et al. (2013) [78]	Centralizado	-	Ataques de Encaminhamento	- Baixa utilização de memória - Sobrecarga de energia mínima	- Criação de tráfego adicional
Amaral et al. (2014) [11]	Híbrido	Especificação	-	- Criação de regras diferentes para cada área da rede	- Maior consumo de energia

Trabalho	Estratégia de Posicionamento	Deteção	Ataques Detetados	Vantagens	Desvantagens
Oh et al. (2014) [79]	Distribuído	Assinatura	Ataques Convencionais	- Baixa complexidade computacional - Baixa utilização de memória - Alta precisão para assinaturas predefinidas	- Deteta um baixo número de intrusões - Não é em tempo real
Cervantes et al. (2015) [5]	Distribuído	Híbrido	Ataques de *sinkhole*	- Alta precisão de deteção - Baixa taxa de falsos positivos e negativos	- Deteta um baixo número de intrusões - Elevada sobrecarga computacional
Pongle e Chavan (2015) [7]	Hibrido	Anomalia	Ataques de *wormhole*	- Alta eficiência energética - Baixo consumo de recursos - Baixa sobrecarga de pacotes	- Baixa precisão de deteção - Alta taxa de falsos positivos - Apenas deteta um tipo de ataque
Summerville et al. (2015) [10]	-	Anomalia	Convencional	- Alta precisão de deteção - Baixa taxa de falsos positivos - Baixa latência	- Alta sobrecarga computacional
Surendar e Umamakeswari (2016) [80]	-	Especificação	Ataques de *sinkhole*	- Baixa sobrecarga - Baixo consumo de energia - Alta precisão de deteção - Baixa taxa de falsos positivos	- Não é em tempo real
Midi et al. (2017) [8]	Híbrido	Híbrido	Ataques de Encaminhamento e Convencionais	- Deteção em tempo real - Leve em termos de requisitos de CPU - Funciona em diferentes protocolos	- A perspetiva de alto nível pode não ser adequada para objetos de computação restritos

Com base na síntese da tabela 3.6 pode-se observar que existem diversas propostas, contudo todas elas têm as suas limitações.

Apesar da solução com uma estratégia de posicionamento centralizada ser a mais comum, ainda não existe uma solução ideal, pois tendo em consideração a proposta de Wallgren et al. [78], no qual têm como vantagem a utilização de uma baixa sobrecarga adicional apesar da criação de tráfego adicional, já a proposta de Kasinathan et al. [6] diz-nos que um dos principais problemas é o alto consumo de recursos.

Em relação a soluções com uma estratégia de posicionamento distribuída, são semelhantes aos resultados adquiridos para uma estratégia de posicionamento centralizada. Tendo em consideração a proposta de Raza et al. [76], este tem um baixo consumo de energia e sobrecarga computacional, contudo a proposta de Cervantes et al. [5] diz-nos que tem uma elevada carga computacional.

Para soluções com uma estratégia de posicionamento híbrida, os resultados adquiridos são semelhantes às estratégias de posicionamento anteriores. Tendo em consideração a proposta de Pongle e Chavan [7], este tem um baixo consumo de recursos e alta eficiência energética (ideal para sistemas com dispositivos restritos), contudo a proposta de Midi et al. [8] diz-nos este necessita de um grande *overhead* computacional.

Em relação a soluções com uma deteção baseada em anomalias, uma incorreta implementação pode-se traduzir num aumento significativo dos recursos, originando uma sobrecarga adicional, aliado a uma baixa precisão de deteção, podendo apenas detetar um único ataque.[7] Este método consegue detetar ataques de encaminhamento e convencionais.

Para soluções com uma deteção baseada em assinaturas, revelam uma menor carga em recursos computacionais, comparando com a deteção baseada em anomalias. Apresentam uma alta precisão para assinaturas predefinidas, reduzindo falsos alarmes, contudo detetam um baixo número de intrusões, pois os ataques detetados dependem das assinaturas. A atualização das assinaturas tem de ser de uma forma sistemática, pois diariamente surgem novas ameaças. Este método consegue detetar ataques convencionais e ataques de DoS.

Para soluções com uma deteção baseada em especificações, permitem a criação de especificações para cada ataque, podendo assim, detetar mais do que um ataque. Assim, a alta precisão de deteção é uma das suas fortes características. Contudo, dependem das especificações declaradas, pois com uma especificação errada aumenta

consideravelmente a taxa de falsos positivos e negativos . Desta forma, a sua implementação pode ser complexa. Este método consegue detetar ataques de encaminhamento e ataques de DDoS.

As soluções com uma deteção híbrida, têm revelado alta precisão na deteção de intrusões, devido ao serem utilizados mais do que um método de deteção para intrusões. Contudo, em alguma propostas a utilização de uma deteção baseada em anomalias aumenta a carga computacional alias ao baixo número de intrusões detetas, quando utilizado uma deteção baseada em especificações. O facto de utilizar mais do que um método de deteção pode induzir numa sobrecarga computacional. Este método consegue detetar ataques de encaminhamento e convencionais.

Em suma, para soluções com uma estratégia de posicionamento ainda não existe uma solução ideal para a deteção de intrusões. As soluções centralizadas necessitam da rede para transmitir informações, normalmente implementados em encaminhadores de periferia, adicionando um maior consumo de energia e recursos. Alguma soluções distribuídas exigem um maior consumo nos dispositivos IoT, que são dispositivos restritos para executar tais tarefas e, por fim, alguma soluções híbridas tentam combinar as vantagens de cada estratégia de posicionamento, tendo em consideração a gestão dos recursos computacionais e energéticos.

Relativamente as soluções baseadas numa deteção, também não existe um consenso sobre qual poderá ser o melhor método para deteção de intrusões. A utilização do método de deteção baseado em anomalias é o método que consome mais recursos computacionais e energéticos, enquanto dos métodos de deteção baseados em assinaturas e especificações são os que menos recursos consomem. Ambas as especificações de assinatura e especificações conseguem atingir uma elevada taxa de deteção, sendo que o método baseado em assinatura peca por detetar um número reduzido de intrusões. Em relação ao método de deteção híbrido, o combinar de dois ou mais métodos de deteção podem induzir num maior recurso computacional mas pode igualmente atingir melhores resultados na taxa de deteção. Em relação aos ataques detetados a maioria consegue detetar ataques de encaminhamento. Já os ataques de negação de serviço, os melhores são os métodos de deteção baseados em assinaturas e especificações.

A maioria das soluções apresentadas adota uma captura de pacotes *packet-based*, e respetivos *payload*, de forma a conseguirem desenvolverem os seus processos de deteção de intrusões.

Por fim, pode-se concluir que a comunidade científica tem realizado um esforço para

superar o desafio de implementar um IDS ideal para sistemas IoT. Contudo os IDS propostos têm ainda algumas limitações quando implementados em sistemas IoT. O facto deste tipo de sistemas necessitarem de uma boa gestão dos recursos por parte dos IDS, tem vindo a ser um problema na sua implementação. O facto de se utilizar uma captura de pacotes e realizar uma inspeção a cada um deles, aumenta a necessidade da utilização dos recursos. O baixo número de deteções de intrusões é igualmente um fator que urge na necessidade de melhorar e diversificar as abordagens. A elevada taxa de falsos positivos apresentadas em algumas soluções, podem ter impacto tanto na performance do IDS como no bom funcionamento do sistema IoT.

Tendo em apreciação estes factos, a utilização de uma solução de deteção de intrusões baseada na análise de fluxos, iria reduzir a poder computacional necessário (ideal para sistemas restritos), comparando com uma análise de pacotes, mais espaço de armazenamento disponível, maior rapidez na deteção de intrusões e aumento da privacidade do utilizador. Adicionalmente a análise de fluxos é mais apropriada para comunicações cifradas.

3.7 Síntese

Neste capítulo foi efetuado um levantamento do estado do conhecimento científico sobre sistemas de deteção de intrusões.

Pretendeu-se também, neste capítulo, abordar as caraterísticas de um IDS e seus constituintes. Foi também apresentado os quatros tipos de classificação de um IDS, nomeadamente por arquitetura, localização, deteção e fonte de dados.

Foi apresentado também uma análise das principais soluções de IDS concebidos especificamente pela comunidade científica para sistemas IoT.

Através desta análise, pode-se concluir que os IDS são um mecanismo válido e muito utilizado atualmente em redes convencionais. Com as diversas soluções propostas e tendo em consideração os resultados dos mesmos, também se pode concluir que os IDS baseados em uma análise por fluxos são mais "leves" computacionalmente comparado com a análise feita através da inspeção de pacotes. Pode-se concluir também que o método por especificação é o mais leve computacionalmente e o que teve melhores resultados a nível da deteção de intrusões comparando com os restantes métodos de deteção. Por sua vez, o método de deteção baseado em anomalias é considerado o mais

pesado computacionalmente. Outro fator predominante nesta análise é o facto de se poder compreender que ainda não existe uma solução de IDS ideal para sistemas IoT, pelo facto de em algumas soluções propostas apenas detetarem um único ataque, outras propostas são pesadas computacionalmente, tornando a sua utilização inviável para este tipo de sistemas e, outras propostas apresentam elevadas taxa de falsos positivos, provenientes de uma má deteção. Desta forma os IDS existentes para sistemas IoT ainda são muito limitados, pelo que é necessário continuar a melhorar os IDS concebidos para sistemas IoT.

No próximo capítulo será apresentado a *framework* proposta para avaliação. Será abordado também as especificações criadas para a deteção de intrusões, tanto para CoAP como para CoAPS. Por último, será realizado um plano de testes que visa validar a *performance* das especificações declaradas.

Capítulo 4

Solução Proposta

Atendendo aos objetivos e motivações apresentados no Capítulo 1, foram realizados no Capítulo 2 e 3 levantamentos sobre o estado do conhecimento científico nas áreas da Internet das Coisas e da deteção de intrusões em sistemas IoT. Este levantamento teve como principal ambição conhecer e validar quais as necessidades que ainda permanecem referentes à segurança e privacidade dos sistemas IoT, analisando igualmente soluções e propostas já existentes para responder a tais dificuldades. Após uma análise cuidada, os resultados indicam que ainda existe um longo percurso, existindo ainda aspetos consideráveis a melhorar em relação à segurança deste tipo de redes.

Com o emergir exponencial dos dispositivos IoT nas redes atuais, a exploração de um IDS é identificado como umas das contra-medias que podem integrar uma solução para sistemas IoT. Apesar desta ser uma solução viável, as propostas apresentadas ainda manifestam limitações que têm de ser ultrapassadas.

Considerando estes fatores, este capítulo visa corresponder à fase de conceção e desenvolvimento apresentado uma proposta de um IDS que pretende contribuir para a resolução dos problemas identificados, alcançando igualmente os objetivos propostos anteriormente.

4.1 Arquitetura Proposta

A diversidade de aplicações nos sistemas IoT, nos mais diversos domínios, impõem grandes desafios ao nível da segurança destes sistemas. Assim, é perentório um solução que possa corresponder a tais dificuldades, garantindo a segurança, a comunicação e os serviços para sistemas IoT. Com o intuito de superar este desafio, foi definido os

seguintes requisitos para a conceção de uma solução de IDS para sistemas IoT [81]:

- **Deteção em todas as camadas:** o IDS deve conseguir detetar ataques e intrusões nas camadas da arquitetura dos sistemas IoT, perceção, rede e aplicação;
- **Deteção de intrusões internas e externas:** capacidade de detetar intrusões oriundas de dispositivos internos do sistema ou externos ao sistema;
- **Deteção em tempo útil:** capacidade para deteção de intrusões que estejam a decorrer no imediato e dentro de um intervalo de tempo razoável para os diversos tipos de intrusões, tornando-o mais *near-time* possível;
- **Escalabilidade:** capacidade para responder a um acréscimo da quantidade de atividades a analisar, considerando o aumento do número de dispositivos proveniente da expansão do sistema;
- **Interoperabilidade:** devido à enorme interoperabilidade neste tipo de sistemas é essencial garantir o suporte das diferentes tecnologias de comunicação, protocolos e standards de forma a garantir o correto funcionamento do sistema;
- **Reconfigurável:** deve ser garantido a inclusão e modificação das políticas previamente impostas para a deteção de intrusões durante a vida útil do sistema;
- **Reduzida implicação no desempenho:** é crucial minimizar o uso dos recursos computacionais dos dispositivos e serviços de forma a mitigar a utilização excessiva dos recursos, nomeadamente armazenamento, processamento, memória volátil e ocupação de largura de banda;
- **Proteção das comunicações internas do sistema IDS:** deve ser garantido a proteção e segurança das comunicações internas entre os diversos componentes do sistema IDS, recorrendo à utilização de canais seguros;
- **Sem alterações de software IoT:** o sistema IDS deve minimizar a modificação ou alteração de *software* aplicacional utilizado nos dispositivos e nos serviços IoT disponibilizados;

Esta solução tem como base os requisitos mencionados anteriormente, garantindo assim o desenvolvimento de um IDS específico para sistemas IoT.

4.1.1 Modelo arquitetural

De forma a privilegiar o desempenho e disponibilidade, a conceção de um IDS para sistemas IoT, atendendo aos recursos disponíveis para este tipo de sistemas, deve utilizar o mínimo de recursos disponíveis em detrito de equipamentos com mais disponibilidade e recursos, como por exemplo, encaminhadores de periferia ou sistemas baseados na nuvem.

Com esse desafio em mente, e considerando também os requisitos apresentados anteriormente, propõe-se a utilização de um IDS do tipo *network-based* baseado em uma análise de fluxos de tráfego IP. A arquitetura baseada em três camadas, abordada no capitulo 2.5, será a arquitetura utilizada de forma a compatibilizar com a arquitetura típica dos serviços e sistemas IoT, conforme ilustrado na Figura 4.1, adaptado de [1].

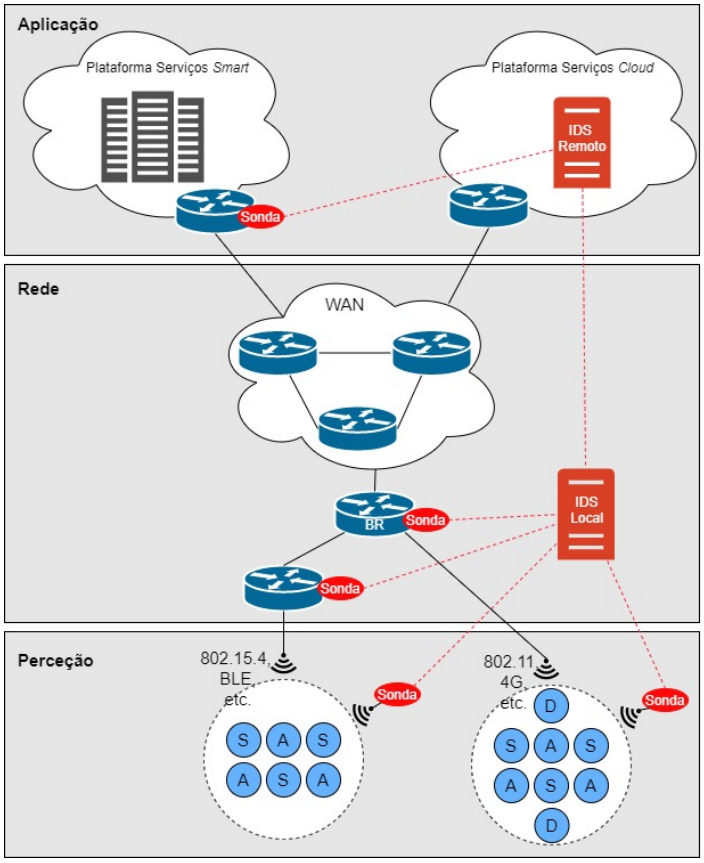


Figura 4.1: Arquitetura proposta para um IDS para IoT baseado na análise de fluxos de tráfego

4.1.2 Caracterização da *Framework*

A *framework* visa a utilização de sondas de captura em cada camada como parte constituinte do modelo arquitetado, pois visa garantir o acesso a todas as comunicações IP realizadas entre dispositivos IoT. Posteriormente, as informações recolhidas pelas sondas são encaminhadas para o IDS responsável pelo armazenamento e análise, designados de IDS local e IDS remoto. O IDS local encontra-se localizado na camada de Rede enquanto o IDS remoto entronca-se localizado na camada de Aplicação. Ambos os IDS estão implementados em dispositivos com maiores recursos computacionais (CPU, memória e energia), como por exemplo, encaminhador de periferia ou um *host* dedicado para esse efeito. Desta forma é possível obter uma visão global sobre as comunicações que ocorrem nos serviços IoT por camada, quer a nível interno quer com a Internet. Através desta visão proporcionada pela monitorização e captura das comunicações em tempo útil, viabiliza a deteção atempadamente de intrusões internas e externas como de ataques de negação de serviço, *Sybil*, entre outros.

Deste modo, o IDS proposto insere-se numa estratégia de localização híbrida, pois considera que a realização das escutas e capturas das comunicações por parte das sondas é efetuada de uma forma distribuída, pelas diferentes camadas, aliada a uma análise realizada de uma forma centralizada em um determinado dispositivo. Através desta estratégia é possível atenuar alguns dos problemas relacionados com a implementação de sistemas IDS em sistemas IoT, devido ao aumento do poder computacional que estes podem adicionar nos sistemas, cujas limitações são limitadas. Assim, é garantindo que existe pouca interferência do mesmo no desempenho dos serviços e sistemas como ainda, uma melhor gestão por parte dos recursos computacionais nos dispositivos IoT.

Quanto ao método de deteção de intrusões, o IDS proposto utiliza a metodologia de deteção de intrusões baseadas em especificações. A escolha por esta técnica recaí pelo facto desta não necessitar de uma fase de aprendizagem, podendo começar a operar de forma imediata após a configuração das especificações previamente declaradas. Outros fatores preponderantes que levaram à escolha desta técnica, passam por um menor poder computacional devido à necessidade das especificações quase de forma estática do comportamento normal dos dispositivos ou serviços. Em relação às outras técnicas anteriormente referenciadas e, também devido à obtenção de taxas de falsos positivos mais baixos, devido a um maior conhecimento dos sistemas e serviços a implementar. Relativamente à atualização das especificações, em caso de alterações, estas devem ser previamente acauteladas possibilitando a alteração e/ou inclusão de novas especificações.

Por último, em relação às intrusões ou ataques que possam ser detetados pela solução proposta, o objetivo passa por detetar qualquer comunicação cujo comportamento se desvie do que é considerado tráfego normal, no qual é individualizado através das especificações implementadas no componente de análise do IDS.

4.1.3 Funcionamento da *Framework*

Atendendo ao facto da *framework* proposta resultar em um IDS baseado na análise de fluxos de tráfego IP, o seu funcionamento deve comportar os componentes desse tipo de sistemas. Através da figura 4.2, adaptado de [1], pode-se observar a composição da *framework* de alto nível para o IDS proposto. Esta é composta por uma ou mais sondas e um módulo de IDS centralizado.

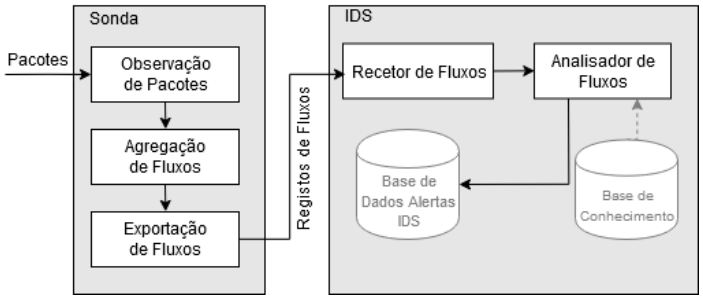


Figura 4.2: *Framework* proposta para um IDS para IoT baseado em análise de fluxos de tráfego

Uma das finalidades das sondas passa por escutar e capturar os pacotes de rede referentes a comunicações IP na rede. Estas podem ser integradas num ou vários dispositivos IoT, como por exemplo, atuadores ou sensores. As sondas localizadas na camada de perceção devem suportar as tecnologias e protocolos utilizados nos sistemas IoT, nomeadamente o 802.15.4, o BLE e o 802.11. Já as sondas localizadas na camada de rede devem capturar as comunicações IP de entrada e saída das interfaces dos dispositivos de encaminhamento. Uma outra finalidade das sondas é a criação e agregação de registos de fluxos, onde posteriormente são exportados para o módulo do IDS centralizado. Uma vantagem das sondas é que as mesmas comunicam quase em tempo real com os módulos IDS centrais, o que permite a deteção de intrusões em tempo útil. No que diz respeito ao módulo do IDS centralizado, este é responsável pela receção dos fluxos enviados pelas sondas, armazenando-os em memória persistente para análise posterior. Essa análise é realizada no componente de análise dos fluxos com base nas especificações armazenadas no repositório de especificações do IDS (Base de Conhecimento). Caso seja detetado alguma anomalia é gerado uma mensagem de alerta de intrusão para o repositório de alertas do IDS.

Com o intuito de assegurar a segurança e a privacidade das comunicações, trocadas entre os diversos componentes do IDS, são, sempre que possível, efetuadas através de redes com fios e transmitidas utilizando cifragem do canal de comunicação. As comunicações efetuadas entre as sondas, localizadas nas camadas de perceção e rede, e o módulo IDS, localizado na camada de rede, são igualmente asseguradas, porém estas utilizam uma VLAN (*Virtual Local Area Network*) dedicada a estas comunicações, introduzindo assim, uma camada complementar de proteção das comunicações internas para o IDS proposto.

Para uma melhor compreensão do funcionamento da *framework* proposta, de seguida será abordado a composição da *framework* de uma forma mais detalha para cada componente. Assim, através da figura 4.3, adaptado de [66], pode-se observar os diferentes componentes integrantes da *framework*, bem como os processos internos de cada um dos componentes de uma forma mais detalhada.

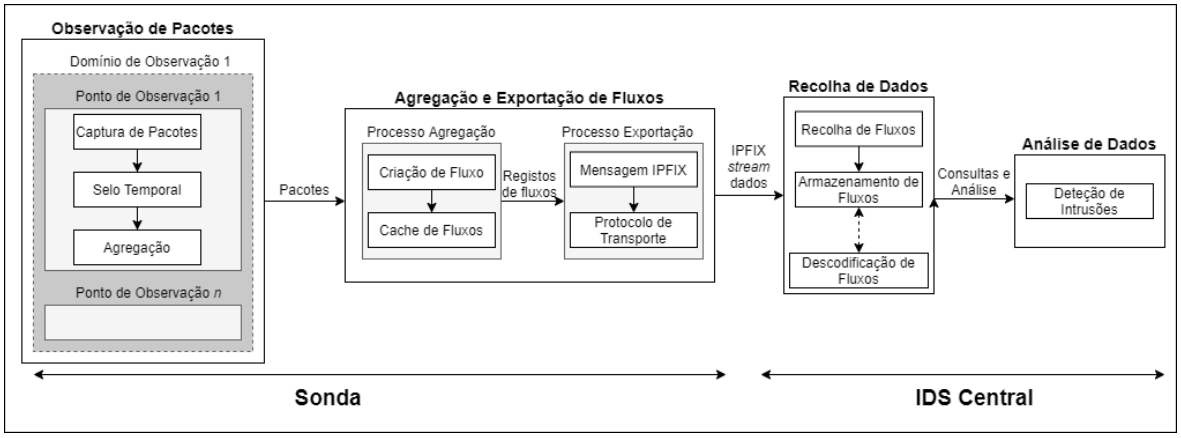


Figura 4.3: *Framework* proposta para um IDS para IoT baseado em análise de fluxos de tráfego

4.1.3.1 Observação de Pacotes

A etapa de observação de pacotes é parte integrante da sonda do IDS. Como abordado anteriormente, a sonda tem como finalidade escutar e capturar os pacotes de rede referentes à comunicação IP na rede. A instalação das sondas podem ser realizadas de duas formas distintas, em modo *port mirror* ou *in-line*. O modo *port mirror* é tipicamente o mais comum, pois é comummente atualizado nos encaminhadores de periferias pelo facto de possuírem mais recursos computacionais e, já integrarem funcionalidades de uma sonda. Em alternativa as sondas podem ser instaladas em modo *in-line* que corresponde ao posicionamento de uma sonda dedicada à interceção e escuta das comunicações com ou sem fios do sistema IoT. [1] De forma a que a captura dos pacotes

de uma ou várias sondas não tenham problemas de sincronismo, o ponto de observação deve sincronizar a data e hora do sistema com recurso ao protocolo Network Time Protocol (NTP). Desta forma a agregação dos pacotes é realizada de uma forma ordena.

4.1.3.2 Agregação e exportação de fluxos

A etapa de agregação e exportação de fluxos é responsável pela agregação dos pacotes capturados na etapa anterior, criando posteriormente, registos de fluxos onde são inseridos os pacotes capturados. Estes registos devem manter dados específicos de cada agregação realizada. Os dados específicos referem-se aos elementos de informação IE, definidos pelo protocolo IPFIX, sendo selecionados na configuração do exportador de fluxos. A utilização destes IE visa o aumento da quantidade de informação sobre cada fluxo e a possibilidade de realizar consultas mais detalhadas durante a análise de fluxos. A existência de IE específicos para registar informações relativas a dados de ambos os sentidos da comunicação, viabiliza a utilização de fluxos unidirecionais ou bidirecionais. A atualização dos IE é responsabilidade da autoridade IANA. Na tabela 4.1 [1] estão listados os IE selecionados para a *framework* proposta.

Tabela 4.1: Lista de IE selecionados para a *framework* proposta

Nome	IE ID	Descrição
flowStartMilliseconds	152	Tempo do início do fluxo em milissegundos desde 1970-01-0100:00:00 UTC. Sempre presente.
flowEndMilliseconds	153	Tempo do fim do fluxo em milissegundos desde 1970-01-0100:00:00 UTC. Sempre presente.
reverseFlowDeltaMilliseconds	21 (PEN 6871)	Diferença de tempo em milissegundos entre o primeiro pacote de saída e do primeiro pacote de entrada de resposta ao anterior. Presente se o fluxo for bidirecional.
protocolIdentifier	4	Protocolo de transporte IP do fluxo. Sempre presente.
sourceIPv4Address	8	Endereço IPv4 origem do fluxo. Sempre presente.
sourceTransportPort	7	Número de pacotes no sentido origem do fluxo. Sempre presente.
packetTotalCount	2	Porto TCP ou UDP de origem do fluxo. Sempre presente.
octetTotalCount	1	Número de octetos dos pacotes no sentido origem do fluxo. Sempre presente.

Nome	IE ID	Descrição
flowAttributes	40 (PEN 6871)	Atributos do fluxo no sentido de origem do fluxo.
sourceMacAddress	56	Endereço MAC origem do primeiro pacote no sentido origem do fluxo.
destinationIPv4Address	12	Endereço IPv4 destino do fluxo.
destinationTransportPort	11	Porto TCP ou UDP de destino do fluxo. Sempre presente.
reversePacketTotalCount	2 (PEN 29305)	Número de pacotes no sentido inverso do fluxo. Presente se o fluxo for bidirecional.
reverseOctetTotalCount	1 (PEN 29305)	Número de octetos dos pacotes no sentido inverso do fluxo. Presente se o fluxo for bidirecional.
reverseFlowAttributes	16424 (PEN 6871)	Atributos do fluxo no sentido inverso do fluxo.
destinationMacAddress	80	Endereço MAC origem do primeiro pacote no sentido inverso do fluxo. Presente se o fluxo for bidirecional.
initialTCPFlags	14 (PEN 6871)	Flags TCP do pacote inicial no sentido origem do fluxo. Presente se o protocolo do fluxo for 6 (TCP).
unionTCPFlags	15 (PEN 6871)	União das flags TCP de todos os outros pacotes, com exceção do inicial, no sentido origem do fluxo. Presente se o protocolo do fluxo for 6 (TCP).
reverseInitialTCPFlags	16398 (PEN 6871)	Flags TCP do pacote inicial no sentido inverso do fluxo. Presente se o protocolo do fluxo for 6 (TCP) e se o fluxo bidirecional.
reverseUnionTCPFlags	16399 (PEN 6871)	União das flags TCP de todos os outros pacotes, com exceção do inicial, no sentido inverso do fluxo. Presente se o protocolo do fluxo for 6 (TCP) e se o fluxo for bidirecional.
tcpSequenceNumber	184	Número de sequência inicial no sentido origem do fluxo. Presente se o protocolo do fluxo for 6 (TCP).
reverseTcpSequenceNumber	184 (PEN 29305)	Número de sequência inicial no sentido inverso do fluxo. Presente se o protocolo do fluxo for 6 (TCP) e se o fluxo for bidirecional.
ingressInterface	10	O índice da interface IP onde os pacotes de rede do fluxo estão a ser recebidos.
engressInterface	14	O índice da interface IP onde os pacotes de rede do fluxo estão a ser recebidos.
vlanId	58	Tag VLAN 802.1q do primeiro pacote no sentido origem do fluxo.
ipClassOfService	5	Para pacotes IPv4, é o valor do campo ToS do cabeçalho IPv4.

Nome	IE ID	Descrição
silkAppLabel	33 (PEN 6871)	Etiqueta da aplicação, definida como a primeira associação de uma aplicação e um porto.
flowEndReason	136	Código de fim do fluxo, conforme definido no IPFIX. Sempre presente.
tcpUrgTotalCount	223	Número de pacotes TCP que têm a flag URGENTE ativa.
smallPacketCount	500 (PEN 6871)	Número de pacotes que contêm menos de 60 bytes de *payload*.
nonEmptyPacketCount	501 (PEN 6871)	Número de pacotes que contêm pelo menos de 1 byte de *payload*.
dataByteCount	502 (PEN 6871)	Total de bytes transferidos como *payload*.
averageInterarrivalTime	503 (PEN 6871)	Média do número de milissegundos entre pacotes.
firstNonEmptyPacketSize	505 (PEN 6871)	Tamanho do *payload* do primeiro pacote não vazio.
largePacketCount	510 (PEN 6871)	Número de pacotes que contêm pelo menos de 220 byte de *payload*.
maxPacketSize	506 (PEN 6871)	O maior tamanho de *payload* transferido no fluxo.
firstEightNonEmptyPacketDirections	507 (PEN 6871)	Representa a direccionalidade dos primeiros 8 pacotes não vazios. 0 para sentido origem, 1 para sentido inverso.
reverseTcpUrgTotalCount	223 (PEN 29305)	Número de pacotes TCP que têm a flag URGENTE ativa no sentido inverso.
reverseSmallPacketCount	16884 (PEN 6871)	Número de pacotes que contêm menos de 60 bytes de *payload* no sentido inverso.
reverseNonEmptyPacketCount	16885 (PEN 6871)	Número de pacotes que contêm pelo menos de 1 byte de *payload* no sentido inverso.
reverseDataByteCount	16886 (PEN 6871)	Total de bytes transferidos como *payload* no sentido inverso.
reverseAverageInterarrivalTime	16887 (PEN 6871)	Média do número de milissegundos entre pacotes no sentido inverso.
reverseFirstNonEmptyPacketSize	16889 (PEN 6871)	Tamanho do *payload* do primeiro pacote não vazio no sentido inverso.
reverseLargePacketCount	16894 (PEN 6871)	Número de pacotes que contêm pelo menos de 220 bytes de *payload* no sentido inverso.

Esta seleção de IE mais alargada, justifica-se pela necessidade de obter mais informação acerca dos pacotes que são analisados, de forma a que a análise seja o mais preciso possível. Alguns dos IE mais comuns na literatura para IoT são os IE de origem e destino dos endereços IP, a quantidade de pacotes e *bytes* enviados e recebidos, data e hora de início e término do fluxo, protocolo utilizado, entre outros.

Esta seleção de IE define o layout da *cache* de fluxos de tráfego mantida no dispositivo a utilizar como exportador de fluxos, nomeadamente uma sonda. No que diz respeito aos tipos, a *cache* de fluxos é do tipo permanente porque permite que os fluxos nunca expirem até serem exportados. Os fluxos de tráfego armazenados e que são exportados são denominados como registos de fluxos. Esses registos de fluxos são integrados em mensagens IPFIX que são, posteriormente, exportadas.

A exportação dos registos de fluxos é efetuada através do exportador de fluxos para o componente de recolha de fluxos utilizando o protocolo de transporte selecionado. Na *framework* proposta é utilizado o protocolo de transporte TCP, devido à sua capacidade de gestão de congestionamentos, disponibilidade, confiabilidade e transporte seguro de informação. Além disso, o TCP é igualmente utilizado sobre TLS, por forma a garantir a confidencialidade e privacidade das mensagens IPFIX. [1]

4.1.3.3 Recolha e armazenamento de fluxos

A recolha e armazenamento de fluxos é responsável pela receção, descodificação e armazenamento dos registos de fluxos previamente enviados por um ou vários exportadores da etapa anterior. Este componente está integrado num dispositivo IoT com maior capacidade computacional, como por exemplo, encaminhadores de periferia ou *hosts* dedicados, devido ao facto de este processo ser penoso para simples dispositivos IoT, cujo o seu processamento é bastante restrito.

De forma a garantir a segurança dos fluxos exportados e a confirmação de entrega dos mesmos, a recolha das mensagens IPFIX enviadas pelo(s) exportador(es), é realizada através de um canal seguro utilizando TLS.

Para finalizar, os fluxos recolhidos são armazenados num sistema de armazenamento persistente. Esta escolha recaí pela utilização otimizada do espaço em disco, contribuindo para um melhor desempenho na escrita dos dados. Estes fluxos passaram posteriormente por uma fase de análise, que é realizada pela próxima etapa.

4.1.3.4 Análise de fluxos

A última etapa é análise de fluxos e tem como principal finalidade a utilização de aplicações com capacidade de consultar os registos de fluxos previamente armazenados pela etapa anterior. Através da figura 4.4, adaptado de [1], pode-se observar um esquema do processo que é realizado nesta etapa.

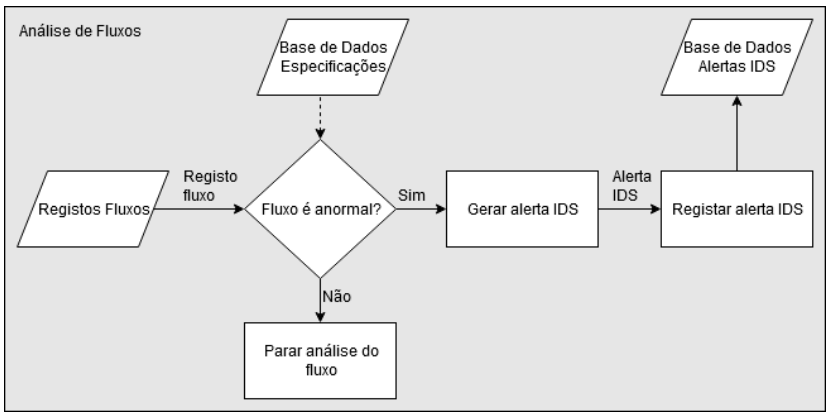


Figura 4.4: Esquema do processo de análise dos fluxos de tráfego

A verificação dos fluxos é baseada em especificações. De forma a validar se um fluxo poderá ser avaliado como um fluxo normal ou anómalo, a comparação é realizada com base em especificações dos comportamentos definidos como normais, no qual se encontram em uma base de dados. Ao realizar a análise, se um fluxo for considerado normal, a análise deve terminar de imediato, seguindo para o próximo fluxo. Caso a fluxo seja considerado anómalo, é gerado uma mensagem de alerta, regista-do-a em uma base de dados de mensagens de alerta de intrusões.

4.2 Protótipo Funcional

Neste subcapítulo será apresentado o protótipo desenvolvido de forma a demonstrar que a *framework* corresponde aos objetivos traçados inicialmente. Este protótipo será desenvolvido em ambiente real, no qual são utilizados atuadores, sensores e serviços IoT. Será também abordado a forma como foram concebidas as especificações a serem utilizadas/testadas no IDS, seguido de um plano de testes de forma a validar a *framework*, tendo como objetivo final a deteção de anomalias em sistemas IoT.

Na figura 4.5 é representado o cenário de testes que irá ser utilizado para a validação da *framework* proposta.

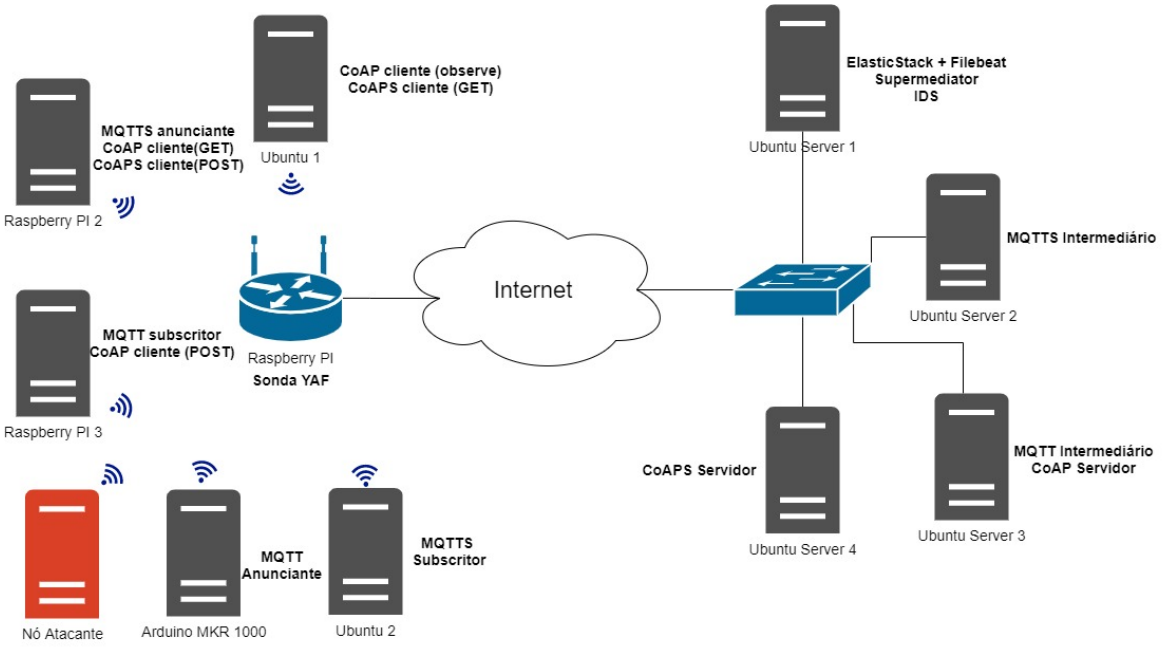


Figura 4.5: Esquema do processo de análise dos fluxos de tráfego

Através do cenário é pretendido recriar um ambiente real de um sistema IoT, sendo constituído dos dispositivos que recorrem a protocolos aplicacionais como o CoAP e o MQTT. Com a utilização destes protocolos pretende-se efetuar troca de dados entre sensores, atuadores e serviços IoT. O objetivo principal é simular um sistema que obtenha a leitura de dados como a temperatura do ar ou a luminosidade e, que atue como um sistema de controlo de Aquecimento, Ventilação e Ar Condicionado (AVAC), que opera mediante a temperatura do ar, ou um interruptor do sistema de iluminação, que adapta a luminosidade consoante a necessidade.

Como se pode observar na figura 4.5, o cenário de testes é constituído por clientes e servidores CoAP, CoAPS (CoAP sobre DTLS), MQTT e MQTTS (MQTT sobre TLS). É igualmente constituído por outros dois dispositivos, um Rasppberry PI que se localiza na sonda do IDS e um Ubuntu Server 1 que se localiza no módulo central do IDS. Para finalizar, existem também um outro dispositivo que tem como finalidade, gerar ataques e intrusões para a validação da *framework* proposta.

O cenário é composto por dois tipos de rede, uma rede *Wi-Fi* (IEEE 802.11n) de forma a criar uma rede interna constituída por sensores e atuadores, e uma rede de cablagem *Fast Ethernet (IEEE 802.3u)* que visa permitir a ligação à Internet e a serviços baseados na nuvem (*cloud*).

O encaminhador de periferia, designado como Raspberry Pi, efetua a interligação e o encaminhamento de tráfego entre a rede privada e a rede pública. Relativamente ao *hardware* utilizado no encaminhador de periferia, este é munido de um Raspberry Pi 3 Model B+ com duas interfaces de rede. Uma interface *Wi-Fi* que é utilizada na rede interna e tem como finalidade fazer de ponto de acesso a uma rede sem fios com endereçamento IPv4, permitindo que sensores e atuadores tenham acesso à Internet e a serviços IoT remotos, além da troca de mensagens entre si. Em relação à interface *Fast Ethernet*, esta tem a finalidade de servir como interligação *Wide Area Network (WAN)*. No que diz respeito a *software*, o encaminhador de periferia é constituído por um SO Raspbian Versão Kernel 4.14, tendo como função adicional, prestar serviços de *Dynamic Host Configuration Protocol (DHCP)* para a rede interna e de sonda para o sistema IDS. Como sonda, irá ficar a escutar e a capturar os pacotes de rede IP que sejam trocados entre a rede interna e a Internet.

A rede interna é constituída por seis dispositivos, um Raspberry PI 2 e Raspberry PI 3, igualmente com mesmo *hardware* e *software* utilizado no encaminhador de periferia, um Ubuntu 1 e Ubuntu 2, constituídos a nível de *hardware* com um Inter x86-64 e a nível de *software* com um SO Ubuntu Desktop 18.04 LTS, um Arduino MKR 1000, constituído a nível de *hardware* com um Arduino Yun e, por último, um dispositivo designado de Nó Atacante com o mesmo *hardware* e *software* dos dispositivos Ubuntu. Todos estes dispositivos apenas utilizam a interface de *Wi-Fi*, tendo como finalidade a interligação à rede interna do sistema IoT, adquirindo endereçamento através de DHCP que se encontra a operar no encaminhador de periferia.

Em relação à rede externa ou Internet, esta é constituída por quatro dispositivos, Ubuntu Server 1,2,3 e 4, constituídos a nível de *hardware* por um Inter x86-64 e a nível de *software* com um SO Ubuntu Desktop 18.04 LTS. Ambos os dispositivos também apenas utilizam a interface de *Wi-Fi* para adquirir endereçamento através de DHCP que se encontra a operar no encaminhador de periferia.

No que concerne às funcionalidades, os dispositivos da rede interna funcionam como sensores e atuadores do sistema IoT. Nestes dispositivos são utilizados protocolos aplicacionais CoAP e MQTT com a finalidade de transmitir dados adquiridos através do meio físico ou para receber dados que permitam atuar consoante a necessidade, ou seja, operam como anunciantes ou subscritores MQTT e clientes CoAP. Por outro lado, os dispositivos da rede externa ou Internet, funcionam como intermediários MQTT ou servidores CoAP para receber, armazenar e anunciar atualizações. Ao contrário dos restantes dispositivos na rede externa, o dispositivo designado como Ubuntu Server 1 tem como finalidade principal implementar o módulo central do IDS, onde é realizado

a receção, armazenamento e análise de todos os fluxos que são recolhidos e exportados pela sonda IDS.

Em relação aos dispositivos que funcionam como clientes e servidores CoAP e CoAPS, são utilizados os pacotes de *software libcoap* e *CoAPhton* para o desenvolvimento de programas que permitem aos dispositivos atuarem como clientes e servidores CoAP. Desta forma, os clientes conseguem obter ou enviar dados sobre a temperatura do ar para os servidores CoAP através da utilização de endereços do tipo URI. É possível também aos clientes CoAP solicitar a observação de atualização de dados de um determinando endereço URI. Este modo tem como finalidade, receber a atualização dos dados sempre que exista alterações de valor associado ao mesmo. Relativamente aos dispositivos que atuam como anunciantes, intermediários ou subscritores MQTT ou MQTTS, é utilizado o pacote de *software Eclipse Mosquitto* no qual permite que estes dispositivos atuem como anunciantes, intermediários ou subscritores MQTT. A ideia passa por permitir que os anunciantes MQTT enviem a temperatura do ar para um intermediário MQTT através da atualização de um tópico. Posteriormente esse intermediário anunciará a atualização ao subscritor MQTT.

Para finalizar, o dispositivo designado como Nó Atacante no qual serão efetuados os ataques e intrusões aos serviços e ao sistema IoT, utiliza as ferramentas de *net scan nmap* e *hping* para análises de rede, inundação e Denial of Service (DoS), assim como por meio de programas MQTT e CoAP. Com recurso a estas ferramentas pretende-se criar informação falsa e realizar um uso abusivo aquando das solicitações aos servidores CoAP e aos intermediários MQTT, de forma a gerar tráfego IP anómalo e malicioso.

Nas subsecções seguintes será abordado as funcionalidade implementadas pela sonda e pelo módulo central do IDS.

De realçar que este projeto teve como base a criação de um sistema IoT de forma a validar a *framework* proposta com o recurso a protocolos aplicacionais como o CoAP e o MQTT. Para a realização deste projeto apenas será contemplado a análise referente ao protocolo CoAP. Assim, apenas considerar-se-á a partir deste ponto a análise ao protocolo aplicacional CoAP.

4.2.1 Sonda

A sonda IDS que se encontra localizada no encaminhador de periferia é responsável por escutar e capturar os dados referentes da rede interna para o exterior. Esta responsabilidade passa por escutar, recolher e exportar informações relativas aos fluxos de tráfego IP que são transmitidos entre as *interfaces* de rede do encaminhador de periferia. Desta forma, são monitorizadas as comunicações IoT entre os clientes e servidores do sistema IoT. Como abordado no subcapítulo 4.1.3, a sonda utilizada pontos de observação de pacotes e para este cenário há apenas um domínio de observação. Neste ponto de observação, a sonda vai utilizar uma solução de código aberto desenvolvido pelo grupo CERT da *Carnegie Mellon University*, o *Yet Another Flowmeter (YAF)*. O YAF tem a capacidade de realizar as três etapas constituintes da sonda (observar figura 4.2). Por último, o YAF tem a vantagem de suportar todos os IE selecionados para a utilização da *framework*. De forma a obter uma melhor performance na captura dos pacotes de rede, foi utilizado a biblioteca *PF_RING*.

4.2.2 Módulo central

O módulo central do IDS encontra-se localizado num *host* designado como Ubuntu Server 1. Este é responsável por receber, armazenar e analisar todos os fluxos de tráfego IP que são exportados pela sonda do IDS. Como na sonda do IDS, o módulo central utilizada igualmente uma solução de código aberto desenvolvido pelo grupo CERT, o *super_mediator*. O *super_mediator* é um recetor e descodificador de mensagens IPFIX e pode utilizar-se em simultâneo com o YAF. O *super_mediator* permite também o armazenamento dos fluxos em ficheiros *JavaScript Object Notation (JSON)*, que posteriormente serão utilizados para a análise de criação das especificações referentes ao protocolo CoAP.

De forma a realizar uma análise dos fluxos armazenados, foi desenvolvido uma aplicação IDS em *Pthyton*. A figura 4.6, adaptado de [1], ilustra o *workflow* que a aplicação utiliza.

O funcionamento da aplicação desenvolvida, inicia-se com a leitura das especificações consideradas como sendo de comportamento normal e que se encontram numa base de dados. As especificações encontram-se armazenadas em formato JSON numa diretoria dedicada para o efeito e dizem respeito aos protocolos aplicacionais existentes no cenário de testes, nomeadamente o CoAP e o MQTT.

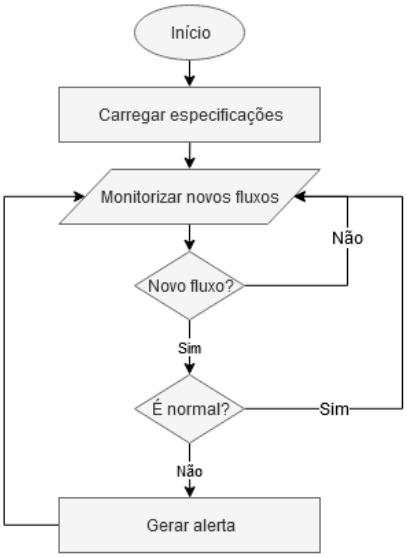


Figura 4.6: *Workflow* do processo de análise dos fluxos de tráfego no IDS

Após o carregamento das especificações, o IDS verifica se existem novos registos de fluxos. No caso de existir novos fluxos, estes serão comparados com as especificações de comportamento normal. Nos casos em que os fluxos sejam considerados como normais, então o processo de verificação termina e volta ao processo de averiguação de novos fluxos. Se o fluxo não for classificado como normal, então é gerado uma mensagem de alerta de intrusão através do protocolo *syslog*. Posteriormente esta mensagens será armazenada em um ficheiro de *logs* no IDS.

4.3 Especificações CoAP e CoAPS

Após a apresentação e descrição do funcionamento do protótipo funcional, pretende-se, neste capítulo, apresentar as especificações que foram utilizadas para a classificação dos fluxos de tráfego IP. As especificações foram alcançadas através da análise aos protocolos aplicacionais em questão, obtendo diversos tipos de mensagens com propósitos distintos, no qual geram inúmeras possibilidades de padrões de tráfego de rede. Por essa razão e para uma melhor compreensão sobre quais as características do tráfego gerado, foi realiza uma recolha ao tráfego em tempo real. De salientar que as mensagens geradas para CoAP foram mensagens do tipo *Confirmable* com respostas *Piggy-backed*.

Para a realização de fluxos de tráfego IP normais foram utilizados os clientes CoAP e CoAPS. Para a realização de fluxos de tráfego IP anómalos foram utilizados o nó atacante, visando criar tráfego de fluxos IP para os ataques, nomeadamente ataques de

Net Scan e, os clientes, de forma a reutilizar os recursos disponíveis, adaptado-os para realizarem ataques a URI inválidos e válidos e ataques DoS. Em relação a análise aos fluxos de tráfego IP foram utilizados os programas *open-source* RapidMider versão 9.3 e wireshark versão 3.0.3. O RapidMiner é uma plataforma de *software* que proporciona um ambiente integrado que auxilia na preparação dos dados para a realização de uma análise aos mesmos. Este programa foi utilizado no âmbito da unidade curricular de Gestão e Análise de Relatórios de Segurança. Em relação ao programa wireshark, este é um programa de análise de tráfego de rede.

Nas tabelas 4.2, 4.3 e 4.4 é apresentado o endereçamento IPv4 do cenário de testes da figura 4.5.

Tabela 4.2: Tabela de endereçamento IPv4 para CoAP

CoAP	**Rede Interna**		**Rede Externa**
Maquina	Raspberry PI 2 Cliente - GET	Raspberry PI 3 Cliente - POST	Ubuntu Server 3 Servidor
Endereço IP	10.42.0.46	10.42.0.118	192.168.111.22

Tabela 4.3: Tabela de endereçamento IPv4 para CoAPS

CoAPS	**Rede Interna**		**Rede Externa**
Maquina	Raspberry PI 2 Cliente - POST	Ubuntu 1 Cliente - GET	Ubuntu Server 4 Servidor
Endereço IP	10.42.0.46	10.42.0.137	192.168.111.23

Tabela 4.4: Tabela de endereçamento do nó atacante

Atacante	
Maquina	Ubuntu
Endereço IP	10.42.0.70

Nos próximos subcapítulos será abordado como foram realizadas as análises aos fluxos de tráfego IP para a criação das especificações.

4.3.1 Especificações para CoAP

Neste subcapítulo será analisado os fluxos referentes ao protocolo aplicacional CoAP, visando encontrar as especificações, tanto para os fluxos com classificação normal como para os fluxos com classificação anómala, sendo estes, posteriormente classificados por tipo de ataque.

Com base na RFC 7252, [2] os fluxos de tráfego IP para o protocolo aplicacional CoAP para este cenário de testes, terá à priori as seguintes características:

- Utilização do protocolo UDP e o porto 5683;
- Cabeçalho do protocolo CoAP é de 4 *bytes*;
- Utilização das seguintes mensagens, GET e POST;

4.3.1.1 Fluxos de tráfego IP classificados como normais

Com recurso ao Wireshark é realizado uma análise ao comportamento do protocolo aplicacional CoAP. Através da figura 4.7, é possível observar que o protocolo aplicacional CoAP funciona numa relação de 1:1, ou seja, o cliente faz um pedido ao servidor e este responde. Em relação aos métodos utilizados para análise apenas são utilizados o método GET e POST, pois são os mais comuns. De salientar que os pedidos realizados são ao URI */temperature*.

No.	Time	Source	Destination	Protocol	Length	Info
3954	559.43…	10.42.0.46	192.168.111.22	CoAP	58	CON, MID:54505, GET, /temperature
3958	559.43…	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:54505, 2.05 Content
4061	581.19…	10.42.0.118	192.168.111.22	CoAP	61	CON, MID:57697, POST, /temperature
4066	581.20…	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:57697, 2.04 Changed
4153	589.46…	10.42.0.46	192.168.111.22	CoAP	58	CON, MID:54506, GET, /temperature
4155	589.46…	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:54506, 2.05 Content
4317	611.23…	10.42.0.118	192.168.111.22	CoAP	61	CON, MID:57698, POST, /temperature
4322	611.24…	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:57698, 2.04 Changed

Figura 4.7: Análise ao comportamento do tráfego CoAP - GET e POST

Através da figura 4.8 é possível verificar que para o método GET, o tamanho para o pacote enviado é de 44 *bytes* e o tamanho do pacote recebido é de 35 *bytes*.

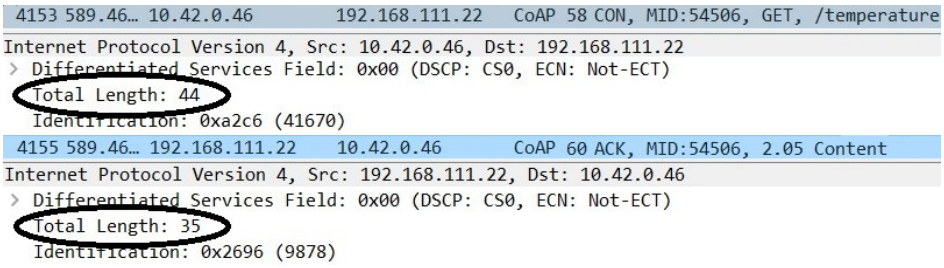


Figura 4.8: CoAP - Método GET - Pacote enviado e recebido

Através da figura 4.9 é possível verificar que para o método POST, o tamanho para o pacote enviado é de 47 *bytes* e o tamanho do pacote recebido é de 44 *bytes*.

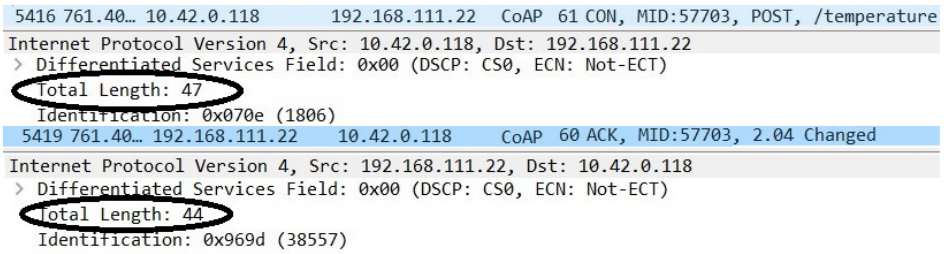


Figura 4.9: CoAP - Método POST - Pacote enviado e recebido

Através da figura 4.10 pode-se constatar que o tamanho de uma mensagem vazia, ou por outras palavras, o tamanho mínimo de uma mensagem CoAP é de 32 *bytes*.

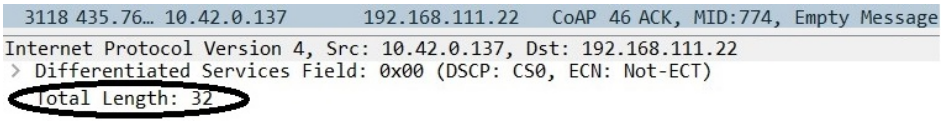


Figura 4.10: CoAP - Mensagem Vazia

De salientar que o tamanho dos pacotes adquiridos através do Wireshark são em função do *super_ mediator*, ou seja, todo o tráfego que venha da camada de Aplicação é considerado como carga útil (*payload*). Assim, pode-se concluir que, se uma mensagem vazia tem o tamanho mínimo de 32 *bytes*, este valor significa que 4 *bytes* são referentes ao cabeçalho CoAP (como referenciado na RFC 7252 [2]), 8 *bytes* são referentes ao cabeçalho UDP e por último, 20 *bytes* referentes ao cabeçalho IP.

Com recurso ao programa RapidMiner é pretendido ter uma outra perspetiva em relação aos fluxos de tráfego IP que não é possível obter através da análise com o Wireshark. Através dessa perspetiva é possível obter uma visualização e análise mais eficaz relativamente aos fluxos de tráfego IP, de forma a retirar algumas ilações referente à informação que cada fluxo.

Row No.	Protocol	src_addr	src_port	dst_addr	dst_port	packetTotalC...	reversePacke...	octetTotalCount	reverseOctetTot...	dataByteCount	ReverseDataByte...
19	UDP	10.42.0.118	39497	192.168.111.22	5683	6	6	282	264	114	96
20	UDP	10.42.0.46	55303	192.168.111.22	5683	6	6	264	210	96	42
21	UDP	10.42.0.118	39497	192.168.111.22	5683	6	6	282	264	114	96
22	UDP	10.42.0.46	55303	192.168.111.22	5683	6	6	264	210	96	42

Figura 4.11: Resultados obtidos dos fluxos com recurso ao RapidMiner

Analisando os resultados apresentados na figura 4.11, pode-se concluir que os fluxos com origem no endereço IP 10.42.0.46 e destino 192.168.111.22 são referentes a fluxos de tráfego IP com o método GET. Por sua vez, os fluxos com origem no endereço IP 10.42.0.118 e destino 192.168.111.22 são referentes a fluxos de tráfego IP com o método POST.

Pode-se igualmente verificar através do IE dst_port que o porto utilizado é o 5683, o que significa, tráfego CoAP. Para ambos os métodos, são enviados (IE *packetTotalCount*) e recebidos (IE *reversePacketTotalCount*) 6 pacotes, o que perfaz uma relação de 1:1.

Em relação ao número de *bytes* enviados e recebidos, pode-se verificar que através do IE *octetTotalCount*, que significa o número total em *bytes* de todos os pacotes enviados, é 264 para o método GET. Se dividirmos pelos 6 pacotes dá um total de 44 *bytes*. Aplicando o mesmo método para os pacotes recebidos, perfaz um total de 35 *bytes* (resultado observado na figura 4.8). Por sua vez, para o método POST, são enviados 282 *bytes*. Se dividirmos pelos 6 pacotes dá um total de 47 *bytes*. Aplicando o mesmo método para os pacotes recebidos, perfaz um total de 44 *bytes* (resultado observado na figura 4.9).

Através da tabela 4.5 é apresentada toda a informação que cada fluxo de tráfego IP contém. Esta tabela servirá de suporte para a criação das especificações para o tráfego normal.

Tabela 4.5: Especificação geral dos registos de fluxos CoAP (GET e POST)

IE's	**CoAP**	
	GET	**POST**
ProtocolIdentifier	17	17
destinationTransportPort	5683	5683
flowEndReason	active/idle	active/idle
octetTotalCount	264	281/283
reverseOctetTotalCount	209/211	264
packetTotalCount	6	6
reversePacketTotalCount	6	6
dataByteCount	96	113/115
reverseDataByteCount	41/43	96
maxPacketSize	16	19/20
reverseMaxPacketSize	8	16
smallPacketCount	6	6
reverseSmallPacketCount	6	6
nonEmptyPacketCount	6	6
reverseNonEmptyPacketCount	6	6
firstNonEmptyPacketSize	16	19
reverseFirstNonEmptyPacketSize	7	16
firstEightNonEmptyPacketDirections	aa	aa

Na tabela 4.5 estão resumidos os resultados adquiridos após análise com o auxílio do RapidMiner. Estes resultados dizem respeito às mensagens dos métodos GET e POST para fluxos normais CoAP. De salientar que tanto o método GET como o método

POST têm comportamento idêntico, pelo que se pode agrupar.

Desta forma, foi criada a tabela 4.6 que corresponde aos valores gerais considerados para os fluxos CoAP. Os IE considerados nestas tabelas são os IE a considerar na implementação das especificações. O motivo para a redução dos IE prende-se pelo facto da informação que estes contêm, ou seja, alguns IE têm informação redundante ou não têm informação de grande relevo.

A título de exemplo, o *octetTotalCount*, que representa o número total de *bytes* enviados e, o *firstNonEmptyPacketSize*, que representa o tamanho do *payload* do primeiro pacote enviado, apesar de terem um significado ligeiramente diferente, para o *octetTotalCount* ser considerado válido, obrigatoriamente o *firstNonEmptyPacketSize* também o terá de ser.

Tabela 4.6: Especificação geral dos registos de fluxos CoAP

IE	Valor
ProtocolIdentifier	17
destinationTransportPort	5683
flowEndReason	active ou idle
octetTotalCount	octetTotalCount / packetTotalCount > 32
reverseOctetTotalCount	reverseOctetTotalCount / reversePacketTotalCount > 32
packetTotalCount	>= 1 e >= reversePacketTotalCount
reversePacketTotalCount	>=1
dataByteCount	dataByteCount/ packetTotalCount > 4
reverseDataByteCount	reverseDataByteCount / reversePacketTotalCount > 4
reverseMaxPacketSize	>= 4
nonEmptyPacketCount	>= 1 e >= reverseNonEmptyPacketCount
reverseNonEmptyPacketCount	>= 1
firstEightNonEmptyPacketDirections	aa

No entanto, devido à possibilidade de o processo de exportação dos registos de fluxos causar a exportação dos mesmos devido ao *active timeout* (*ative timeout* 180 segundos e *idle timeout* 60 segundos), os registos de fluxos podem ter outras caraterísticas, mesmo que as comunicações capturadas sejam normais. Os registos dos fluxos são retirados da *cache* e exportados sempre que atingirem o tempo sem atividade de 1 minuto ou o tempo de atividade de 3 minutos. As tabelas 4.7 e 4.8 representam uma outra alternativa para os fluxos CoAP.

Tabela 4.7: Especificação alternativa dos registos de fluxos CoAP

IE	Valor
ProtocolIdentifier	17
sourceTransportPort	5683
flowEndReason	idle
octetTotalCount	> 32
reverseOctetTotalCount	-
packetTotalCount	1
reversePacketTotalCount	-
dataByteCount	> 4
reverseDataByteCount	-
reverseMaxPacketSize	-
nonEmptyPacketCount	1
reverseNonEmptyPacketCount	-
firstEightNonEmptyPacketDirections	a

Tabela 4.8: Especificação alternativa dos registos de fluxos CoAP

IE	Valor
ProtocolIdentifier	17
sourceTransportPort	5683
flowEndReason	active ou idle
octetTotalCount	octetTotalCount / packetTotalCount > 32
reverseOctetTotalCount	reverseOctetTotalCount / reversePacketTotalCount > 32
packetTotalCount	>= 1 e >= reversePacketTotalCount
reversePacketTotalCount	>=1
dataByteCount	dataByteCount/ packetTotalCount > 4
reverseDataByteCount	reverseDataByteCount / reversePacketTotalCount > 4
reverseMaxPacketSize	>= 4
nonEmptyPacketCount	>= 1 e >= reverseNonEmptyPacketCount
reverseNonEmptyPacketCount	>= 1
firstEightNonEmptyPacketDirections	aa

4.3.1.2 Fluxos de tráfego IP classificados como anómalos

De forma a classificar os fluxos anómalos por tipo de ataque, estes também terão as suas respetivas especificações. Serão analisados ataques do tipo *Net Scan*, de forma a evitar ataques de mapeamento na rede, ataques a URI inválidos, de forma a evitar tentativas de acesso a informação não autorizada e, ataques em rajada a URI validos e inválidos, de forma a evitar ataques de negação de serviço. De seguida, será abordado a forma como foram criadas as especificações para cada tipo de ataque.

Ataques de NET SCAN

Os ataques de *Net Scan* são ataques utilizados para mapear uma rede de forma a permitir ao atacante perceber que endereços IP existem na rede, qual o estado dos portos na rede, entre outras funcionalidades. Para a realização deste tipo de ataque, foram utilizadas as ferramentas NMAP e HPING. A ferramenta NMAP permite realizar mapeamentos numa rede, mapear portos tentando descobrir qual o seu estado, entre outros. No caso da ferramenta HPING pode ser utilizado como um analisador de pacotes, possui o modo *traceroute* e é utilizado na criação de pacotes para testes em sistemas de deteção de intrusões.

Novamente com recurso ao Wireshark é realizado uma análise ao comportamento dos ataques através das ferramentas NMAP e HPING. Através da figura 4.12 pode-se observar que o nó atacante gera pedidos ao servidor e através da figura 4.13 pode-se observar uma tentativa de mapeamento, tentando descobrir informações sobre o servidor CoAP. Ambas as figuras demonstram o comportamento dos ataques.

No.	Time	Source	Destination	Protocol	Length	Info
45	9.604598	10.42.0.70	192.168.111.22	UDP	42	1223 → 5683 Len=0
46	9.604892	10.42.0.70	192.168.111.22	UDP	42	1224 → 5683 Len=0
56	9.607165	192.168.111.22	10.42.0.70	CoAP	60	RST, MID:982, 4.00 Bad Request

Figura 4.12: Análise ao comportamento do ataque utilizando a ferramenta HPING

No.	Time	Source	Destination	Protocol	Length	Info
174	9.631253	10.42.0.70	192.168.111.22	ICMP	74	Destination unreachable (Port unreachable)
173	9.631108	10.42.0.70	192.168.111.22	ICMP	74	Destination unreachable (Port unreachable)
172	9.630989	192.168.111.22	10.42.0.70	CoAP	60	RST, MID:1017, 4.00 Bad Request
171	9.630356	192.168.111.22	10.42.0.70	CoAP	60	RST, MID:1016, 4.00 Bad Request

Figura 4.13: Análise ao comportamento do ataque utilizando a ferramenta NMAP

Através da figura 4.14 pode-se observar o tamanho dos pacotes referentes ao ataque *Net Scan* utilizando a ferramenta HPING. Para um o pacote enviado, o seu tamanho é de 28 *bytes* e o tamanho do pacote recebido é de 32 *bytes.*

No.	Time	Source	Destination	Protocol	Length	Info
55	9.607069	10.42.0.70	192.168.111.22	UDP	42	1233 → 5683 Len=0

Internet Protocol Version 4, Src: 10.42.0.70, Dst: 192.168.111.22

Total Length: 28 ⇦

No.	Time	Source	Destination	Protocol	Length	Info
56	9.607165	192.168.111.22	10.42.0.70	CoAP	60	RST, MID:982, 4.00 Bad Request

Internet Protocol Version 4, Src: 192.168.111.22, Dst: 10.42.0.70

Total Length: 32 ⇦

Figura 4.14: Ataque *Net Scan* utilizando a ferramenta HPING

Através da figura 4.15 pode-se observar o tamanho dos pacotes referentes ao ataque

Net Scan utilizando a ferramenta NMAP. Para um o pacote enviado, o seu tamanho é de 28 *bytes* e o tamanho do pacote recebido é de 32 *bytes*.

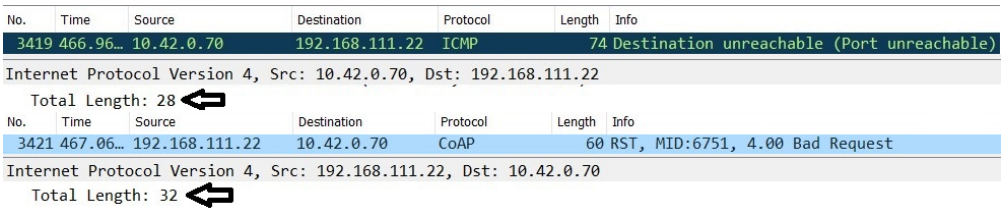


Figura 4.15: Ataque *Net Scan* utilizando a ferramenta NMAP

Novamente com recurso ao RapidMiner é pretendido demonstrar uma outra perspetiva dos ataques. A figura 4.16 demonstra uma visualização e analise mais eficaz relativamente aos fluxos de ataque.

Row...	Protocol	src_addr	src_port	dst_addr	dst_port	packetTotal...	reversePack...	octetTotalC...	reverseOcte...	dataByteCount
1	UDP	10.42.0.70	1552	192.168.111.22	5683	1	1	28	32	0
2	UDP	10.42.0.70	1553	192.168.111.22	5683	1	1	28	32	0

Row...	Protocol	src_addr	src_port	dst_addr	dst_port	packetTotal...	reversePack...	octetTotalC...	reverseOcte...	dataByteCount
1	UDP	10.42.0.70	47545	192.168.111.22	5683	1	1	28	32	0
2	UDP	10.42.0.70	47546	192.168.111.22	5683	1	1	28	32	0

Figura 4.16: Resultados RapidMiner para ataques *Net Scan* utilizando as ferramentas HPING e NMAP

Analisando os resultados apresentados na figura 4.16, pode-se concluir que os fluxos com origem no endereço IP 10.42.0.70 e destino 192.168.111.22 são referentes a fluxos de tráfego IP realizados pelo nó atacante. Relativamente à primeira parte da figura, esta representa os resultados utilizando a ferramenta HPING e a segunda parte representa os resultados utilizando a ferramenta NMAP.

Independentemente da ferramenta utilizada, pode-se verificar que os ataques são realizados ao porto 5683. Utilizando a ferramenta HPING, o tamanho de um pacote enviado é de 28 *bytes* e o tamanho de um pacote recebido é de 32 *bytes* (resultados obtido na figura 4.14). Utilizando a ferramenta NMAP, o tamanho de um pacote enviado é de 28 *bytes* e o tamanho de um pacote recebido é de 32 *bytes*. (resultado obtido na figura 4.15).

Por último, em ambos os ataques o IE *dataByteCount* tem o valor 0. Este valor representa o tamanho do *payload* em *bytes* dos pacotes enviados.

Na tabela 4.9 é apresentada toda a informação de cada fluxo referente aos ataques. Esta tabela servirá de suporte para a criação das especificações para os ataques de *Net Scan*.

Tabela 4.9: Especificação geral dos registos de fluxos para ataques *Net Scan*

IE's	*Net Scan*	
	HPING	**NMAP**
ProtocolIdentifier	17	17
destinationTransportPort	5683	5683
flowEndReason	idle	idle
octetTotalCount	28	28
reverseOctetTotalCount	32	32
packetTotalCount	1	1
reversePacketTotalCount	1	1
dataByteCount	0	0
reverseDataByteCount	4	4
maxPacketSize	0	0
reverseMaxPacketSize	4	4
smallPacketCount	0	0
reverseSmallPacketCount	1	1
nonEmptyPacketCount	0	0
reverseNonEmptyPacketCount	1	1
firstNonEmptyPacketSize	0	0
reverseFirstNonEmptyPacketSize	4	4
firstEightNonEmptyPacketDirections	1	1

Com base nestas caraterísticas foi criado uma especificação para ataques *Net Scan*. Na tabela 4.10 é demonstrado a especificação geral para este tipo de ataques.

Tabela 4.10: Especificação para ataques *Net Scan*

IE	Valor
ProtocolIdentifier	17
destinationTransportPort	5683
flowEndReason	idle
octetTotalCount	octetTotalCount / packetTotalCount <= 32
dataByteCount	dataByteCount / packetTotalCount = 0

Ataques de URI Inválidos

Os ataques a URI inválidos visam tentar obter informações sobre os dispositivos ou injetar informações erradas nos dispositivos. Desta forma é essencial evitar que o atacante ganhe acesso a informações que não tem autorização ou injete informações erradas. Um atacante através deste ataque pode tentar descobrir qual o URI correto para obter ou alterar informações no alvo. Este ataque pode ser utilizado em simultâneo com um ataque *Net Scan*, onde o atacante numa primeira instância, tenta descobrir os endereços IP disponíveis na rede para selecionar um alvo e, numa segunda instância,

através de um ataque a URI inválido, tentar descobrir qual o caminho correto para posteriores ataques. Caso este ataque seja concretizado em rajada, poderá inviabilizar a disponibilidade e capacidade do alvo em conseguir responder aos pedidos realizados, num curto período de tempo, criando assim um ataque de negação de serviço. Desta forma, um ataque em rajada a URI inválido é classificado como um ataque DoS podendo este assumir várias formas, isto é, através da sobrecarga da largura de banda no alvo de forma a torná-lo indisponível ou levá-lo ao esgotamento dos recursos, impedindo-o de responder ao tráfego e clientes legítimos.

Aqui serão analisados os ataques a URI inválidos, incluindo os ataques em rajada. Numa primeira fase será analisado os ataques a URI inválidos utilizando os métodos GET e POST. Numa segunda fase será analisado os ataques em rajada a URI inválidos, igualmente utilizando os métodos GET e POST.

Na figura 4.17 é demonstrado o comportamento dos ataques a URI inválidos utilizando os métodos GET e POST. Pode-se observar também que os pedidos são realizados ao URI */temp*. Contudo têm respostas diferentes, pois para o método GET a resposta é *4.04 Not Found*, pelo facto do atacante estar a realizar um pedido para obter informação relativa aquele URI que não existe. No caso do método POST, a resposta é *4.05 Method Not Alloed*, pois o atacante está a tentar alterar informação de um URI que não existe. Para ambos, a resposta recebida por parte do alvo é um *Reset*, o que equivale a 4 *bytes*.

No.	Time	Source	Destination	Protocol	Length	Info
78	10.220685	10.42.0.46	192.168.111.22	CoAP	51	CON, MID:48321, GET, /temp
79	10.226771	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:48321, 4.04 Not Found
190	25.038767	10.42.0.118	192.168.111.22	CoAP	54	CON, MID:27469, POST, /temp
191	25.044485	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:27469, 4.05 Method Not Allowed

Figura 4.17: Análise ao comportamento de ataques a URI inválidos - GET e POST

Através da figura 4.18 pode-se verificar que o tamanho para o pacote enviado com o método GET é de 37 *bytes* e o tamanho do pacote recebido é de 32 *bytes*.

No.	Time	Source	Destination	Protocol	Length	Info
78	10.220685	10.42.0.46	192.168.111.22	CoAP	51	CON, MID:48321, GET, /temp

Internet Protocol Version 4, Src: 10.42.0.46, Dst: 192.168.111.22
Total Length: 37 ⇦

No.	Time	Source	Destination	Protocol	Length	Info
79	10.226771	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:48321, 4.04 Not Found

Internet Protocol Version 4, Src: 192.168.111.22, Dst: 10.42.0.46
Total Length: 32 ⇦

Figura 4.18: Ataque a URI inválidos - GET

Através da figura 4.19 pode-se verificar que o tamanho para o pacote enviado com o método GET é de 40 *bytes* e o tamanho do pacote recebido é de 32 *bytes*.

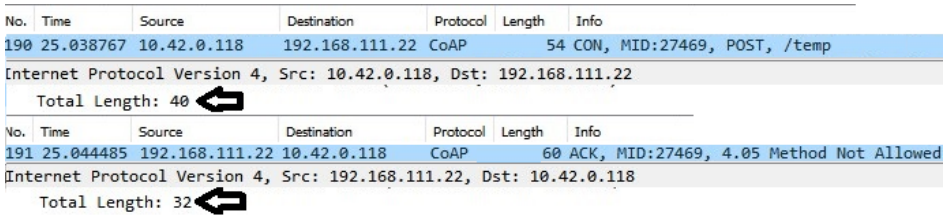


Figura 4.19: Ataque a URI inválidos - POST

A figura 4.20 demonstra a informação recolhida dos fluxos utilizando o RapidMiner.

Row No.	Protocol	src_addr	src_port	dst_addr	dst_port	flowEndR...	packetTotal...	octetTotal...	reversePac...	reverseOctetT...	ReverseMax...
1	17	10.42.0.46	59110	192.168.111.22	5683	active	6	222	6	192	4
2	17	10.42.0.46	59110	192.168.111.22	5683	active	6	222	6	192	4
3	17	10.42.0.118	43183	192.168.111.22	5683	active	6	240	6	192	4
4	17	10.42.0.118	43183	192.168.111.22	5683	active	6	239	6	192	4

Figura 4.20: Resultados RapidMiner para ataque a URI inválidos

Analisando os resultados apresentados na figura 4.20, pode-se concluir que os fluxos com origem no endereço IP 10.42.0.46 e destino 192.168.111.22 são referentes a fluxos de tráfego IP com o método GET. Por sua vez, os fluxos com origem no endereço IP 10.42.0.118 e destino 192.168.111.22 são referentes a fluxos de tráfego IP com o método POST.

Ambos os ataques são realizados ao porto 5683 e têm uma relação de 1:1. Para o ataque a URI inválido utilizando o método GET, o tamanho de um pacote enviado é de 37 *bytes* e o tamanho de um pacote recebido é de 32 *bytes* (resultado obtido na figura 4.18). Para o ataque a URI inválido utilizando o método POST, o tamanho de um pacote enviado é de 40 *bytes* e o tamanho de um pacote recebido é de 32 *bytes* (resultado obtido na figura 4.19).

Por último, ambos os métodos tem o valor de 4 no IE *reverseMaxPacketSize*. Este valor representa o tamanho do *payload* em *bytes* do maior pacote recebido. De salientar que o valor para uma mensagem *Reset*, anteriormente falado, coincide com o valor para o IE *reverseMaxPacketSize*.

Analisando agora os ataques em rajada a URI inválidos, utilizando os métodos GET e POST, pode-se verificar o seu comportamento através da figura 4.21. Pode-se constatar também que os pedidos enviados e recebidos são iguais ao ataque a URI

inválido, contudo sendo um ataque de rajada, são efetuados muitos mais pedidos num curto período de tempo.

No.	Time	Source	Destination	Protocol	Length	Info
19672	244.6392…	10.42.0.46	192.168.111.22	CoAP	51	CON, MID:1747, GET, /temp
19673	244.6395…	10.42.0.46	192.168.111.22	CoAP	51	CON, MID:1748, GET, /temp
19674	244.6418…	10.42.0.46	192.168.111.22	CoAP	51	CON, MID:1749, GET, /temp
19675	244.6434…	10.42.0.118	192.168.111.22	CoAP	53	CON, MID:25934, POST, /temp
19676	244.6443…	10.42.0.118	192.168.111.22	CoAP	54	CON, MID:25932, POST, /temp
19677	244.6464…	10.42.0.118	192.168.111.22	CoAP	53	CON, MID:25933, POST, /temp
19678	244.6489…	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:1747, 4.04 Not Found
19681	244.6523…	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:1748, 4.04 Not Found
19682	244.6549…	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:1749, 4.04 Not Found
19683	244.6572…	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:25934, 4.05 Method Not Allowed
19684	244.6595…	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:25932, 4.05 Method Not Allowed
19685	244.6605…	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:25933, 4.05 Method Not Allowed

Figura 4.21: Análise ao comportamento de ataques em rajada a URI inválido

Através da figura 4.22 pode-se verificar que o tamanho para o pacote enviado com o método GET é de 37 *bytes* e o tamanho do pacote recebido é de 32 *bytes*.

No.	Time	Source	Destination	Protocol	Length	Info
19672	244.6392…	10.42.0.46	192.168.111.22	CoAP	51	CON, MID:1747, GET, /temp

Internet Protocol Version 4, Src: 10.42.0.46, Dst: 192.168.111.22

Total Length: 37 ⇦

	Time	Source	Destination	Protocol	Length	Info
19678	244.6489…	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:1747, 4.04 Not Found

Internet Protocol Version 4, Src: 192.168.111.22, Dst: 10.42.0.46

Total Length: 32 ⇦

Figura 4.22: Ataque em Rajada a URI inválido - GET

Através da figura 4.23 pode-se verificar que o tamanho para o pacote enviado com o método POST é de 39 *bytes* e o tamanho do pacote recebido é de 32 *bytes*.

No.	Time	Source	Destination	Protocol	Length	Info
19675	244.6434…	10.42.0.118	192.168.111.22	CoAP	53	CON, MID:25934, POST, /temp

Internet Protocol Version 4, Src: 10.42.0.118, Dst: 192.168.111.22

Total Length: 39 ⇦

	Time	Source	Destination	Protocol	Length	Info
19683	244.6572…	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:25934, 4.05 Method Not Allowed

Internet Protocol Version 4, Src: 192.168.111.22, Dst: 10.42.0.118

Total Length: 32 ⇦

Figura 4.23: Ataque em Rajada a URI inválido - POST

A figura 4.24 demonstra a informação recolhida dos fluxos utilizando o RapidMiner.

Row No.	Protocol	src_addr	src_port	dst_addr	dst_port	flowEndR…	packetTotal…	octetTotalC…	reversePacket…	reverseOctet…	ReverseMax…
1	17	10.42.0.46	48626	192.168.111.22	5683	idle	615	22755	63	2016	4
2	17	10.42.0.46	49042	192.168.111.22	5683	idle	1003	37111	1003	32096	4
3	17	10.42.0.46	43082	192.168.111.22	5683	idle	1000	37000	1000	32000	4
23	17	10.42.0.118	35125	192.168.111.22	5683	idle	12	477	12	384	4
24	17	10.42.0.118	37882	192.168.111.22	5683	active	335	13371	335	10720	4
25	17	10.42.0.118	52318	192.168.111.22	5683	idle	364	14515	364	11648	4

Figura 4.24: Resultados RapidMiner para ataque de Rajada a URI inválido - GET

Analisando os resultados apresentados na figura 4.24, pode-se concluir que os fluxos com origem no endereço IP 10.42.0.46 e destino 192.168.111.22 são referentes a fluxos de tráfego IP com o método GET. Por sua vez, os fluxos com origem no endereço IP 10.42.0.118 e destino 192.168.111.22 são referentes a fluxos de tráfego IP com o método POST. Ambos os ataques são realizados ao porto 5683 e têm uma relação de 1:1. Para o ataque em rajada a URI inválido utilizando o método GET, o tamanho de um pacote enviado é de 37 *bytes* e o tamanho de um pacote recebido é de 32 *bytes* (resultado obtido na figura 4.22). Para o ataque em rajada a URI inválido utilizando o método POST, o tamanho de um pacote enviado é de 39 *bytes* e o tamanho de um pacote recebido é de 32 *bytes* (resultado obtido na figura 4.23).

Pode-se verificar também que os ataques diferem no resultado em relação ao IE *flowEndReason*. O ataque utilizando o método GET é sempre *idle*, enquanto o ataque utilizando o método POST tem ambas as opções, *active* e *idle*. Por último, ambos os métodos tem o valor de 4 no IE *reverseMaxPacketSize*. Este valor representa o tamanho do *payload* em *bytes* do maior pacote recebido.

Na tabela 4.11 é apresentado toda a informação de cada fluxo referentes ao ataque a URI inválido, incluindo o ataque em rajada.

Tabela 4.11: Especificação geral dos registos de fluxos para ataques a URI Inválidos

IE	URI Inválido		URI Inválido Rajada	
	GET	**POST**	**GET**	**POST**
ProtocolIdentifier	17	17	17	17
destinationTransportPort	5683	5683	5683	5683
flowEndReason	active	active	active	idle
octetTotalCount	222	240	> 240	> 240
reverseOctetTotalCount	192	192	> 192	> 192
packetTotalCount	6	6	> 1	> 1
reversePacketTotalCount	6	6	> 1	> 1
dataByteCount	54	72	> 72	> 72
reverseDataByteCount	24	24	> 24	> 24
maxPacketSize	9	12	9	12
reverseMaxPacketSize	4	4	4	4
smallPacketCount	6	6	> 1	> 1
reverseSmallPacketCount	6	6	> 1	> 1
nonEmptyPacketCount	6	6	> 1	> 1
reverseNonEmptyPacketCount	6	6	> 1	> 1
firstNonEmptyPacketSize	9	12	9	12
reverseFirstNonEmptyPacketSize	4	16	4	12
firstEightNonEmptyPacketDirections	aa	aa	aa/...	aa/...

Com base nestas características foi criado uma especificação para ataques a URI inválidos, incluindo ataques em rajada. Na tabela 4.12 é demonstrado a especificação geral para este tipo de ataque.

Tabela 4.12: Especificação para ataques a URI inválido

IE	Valor
ProtocolIdentifier	17
destinationTransportPort	5683
flowEndReason	active/idle
octetTotalCount	octetTotalCount / packetTotalCount > 32
reverseOctetTotalCount	reverseOctetTotalCount / reversePacketTotalCount > 32
packetTotalCount	>= 1 e >= reversePacketTotalCount
reversePacketTotalCount	>= 1
ReverseMaxPacketSize	= 4

Ataques em rajada a URI Válidos

Os ataques em rajada a URI válidos são classificados como ataques DoS, como o ataque anterior das rajadas a URI inválidos. Sendo um ataque DoS a ideia passa por inviabilizar a disponibilidade e capacidade do alvo em conseguir responder aos pedidos realizados, num curto período de tempo, impedindo-o de responder ao tráfego e clientes legítimos.

Aqui serão analisados os ataques em rajada a URI válidos utilizando os métodos GET e POST. Na figura 4.25 é demonstrado o comportamento deste tipo de ataque. Pode-se observar que os pedidos são realizados ao URI */temperature*. Contudo têm respostas diferentes, pois para o método GET a resposta é *2.05 Content*, o que significa que o atacante conseguiu obter informação sobre o URI. No caso do método POST, a resposta é *2.04 Changed*, o que significa que o atacante conseguiu alterar informação naquela URI.

No.	Time	Source	Destination	Protocol	Length	Info
41370	484.53…	10.42.0.46	192.168.111.22	CoAP	58	CON, MID:19648, GET, /temperature
41371	484.53.	10.42.0.46	192.168.111.22	CoAP	58	CON, MID:19649, GET, /temperature
41380	484.55.	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:19648, 2.05 Content
41381	484.55.	192.168.111.22	10.42.0.46	CoAP	60	ACK, MID:19649, 2.05 Content
41374	484.54.	10.42.0.46	192.168.111.22	CoAP	58	CON, MID:19652, GET, /temperature
1339	237.72	10.42.0.118	192.168.111.22	CoAP	61	CON, MID:58822, POST, /temperature
1340	237.72	10.42.0.118	192.168.111.22	CoAP	61	CON, MID:58823, POST, /temperature
1341	237.73	10.42.0.118	192.168.111.22	CoAP	61	CON, MID:58824, POST, /temperature
1349	237.76	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:58823, 2.04 Changed
1350	237.76	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:58822, 2.04 Changed
1351	237.76	192.168.111.22	10.42.0.118	CoAP	60	ACK, MID:58824, 2.04 Changed
1345	237.74	10.42.0.118	192.168.111.22	CoAP	61	CON, MID:58828, POST, /temperature

Figura 4.25: Análise ao comportamento do ataque Rajada a URI válido - GET

Através da figura 4.26 pode-se verificar que o tamanho para o pacote enviado com o método GET é de 44 *bytes* e o tamanho do pacote recebido é de 35 *bytes*.

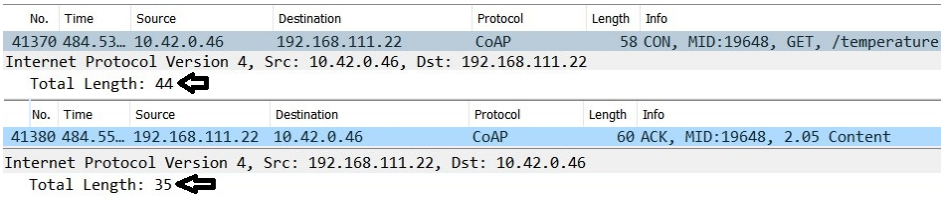


Figura 4.26: Ataque em Rajada a URI válido - GET

Através da figura 4.27 pode-se verificar que o tamanho para o pacote enviado com o método POST é de 47 *bytes* e o tamanho do pacote recebido é de 44 *bytes*.

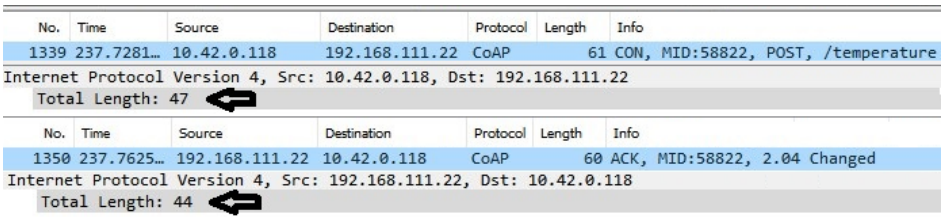


Figura 4.27: Ataque em Rajada a URI válido - POST

A figura 4.28 demonstra a informação recolhida dos fluxos utilizando o RapidMiner.

Row No.	Protocol	src_addr	src_port	dst_port	dst_addr	flowEndRea...	packetTotal...	octetTotalC...	reversePac...	reverseOcte...	firstEightNo...
1	UDP	10.42.0.46	44125	5683	192.168.111.22	idle	15	660	15	525	ef
2	UDP	10.42.0.46	48091	5683	192.168.111.22	idle	14	616	14	490	ff
13	UDP	10.42.0.118	47418	5683	192.168.111.22	active	95	4440	98	4300	aa
14	UDP	10.42.0.118	56054	5683	192.168.111.22	idle	9	421	9	396	aa
16	UDP	10.42.0.118	60266	5683	192.168.111.22	active	61	2862	61	2684	ff
17	UDP	10.42.0.118	60266	5683	192.168.111.22	idle	15	704	15	660	fe

Figura 4.28: Resultados RapidMiner para ataque de Rajada a URI válido - GET e POST

Analisando os resultados apresentados na figura 4.28, pode-se concluir que os fluxos com origem no endereço IP 10.42.0.46 e destino 192.168.111.22 são referentes a fluxos de tráfego IP com o método GET. Por sua vez, os fluxos com origem no endereço IP 10.42.0.118 e destino 192.168.111.22 são referentes a fluxos de tráfego IP com o método POST.

Ambos os ataques são realizados ao porto 5683 e têm uma relação de 1:1. Para o ataque em rajada a URI válidos utilizando o método GET, o tamanho de um pacote enviado é de 44 *bytes* e o tamanho de um pacote recebido é de 35 *bytes* (resultados obtido na figura 4.26). Para o ataque em rajada a URI válidos utilizando o método

POST, o tamanho de um pacote enviado é de 47 *bytes* e o tamanho de um pacote recebido é de 44 *bytes* (resultados obtidos na figura 4.27).

Pode-se concluir também que os ataques diferem no resultado em relação ao IE *flowEndReason*. O ataque utilizando o método GET é sempre *active*, enquanto o ataque utilizando o método POST tem ambas as opções, *active* e *idle*. Por último, ambos têm valores dispares no IE *firstEightNonEmptyPacketDirections*. Este valor corresponde à direção dos 8 primeiros pacotes não vazios, em hexadecimal. De salientar que os valores do IE *firstEightNonEmptyPacketDirections*, cujo valor seja diferente de "aa", será detetado como ataque. Caso o valor seja igual a "aa", passa como tráfego normal.

Na tabela 4.13 é apresentado toda a informação de cada fluxo referentes ao ataque a URI inválido, incluindo o ataque em rajada.

Tabela 4.13: Especificação geral dos registos de fluxos para ataques em rajada a URI válidos

IE	**URI válido Rajada**	
	GET	**POST**
ProtocolIdentifier	17	17
destinationTransportPort	5683	5683
flowEndReason	active	active/idle
octetTotalCount	> 32	> 32
reverseOctetTotalCount	> 32	> 32
packetTotalCount	> 1	> 1
reversePacketTotalCount	> 1	> 1
dataByteCount	> 16	> 19
reverseDataByteCount	> 7	> 16
maxPacketSize	16	entre 13 e 20
reverseMaxPacketSize	entre 6 e 7	entre 4 e 16
smallPacketCount	> 1	> 1
reverseSmallPacketCount	> 1	> 1
nonEmptyPacketCount	> 1	> 1
reverseNonEmptyPacketCount	> 1	> 1
firstNonEmptyPacketSize	16	entre 13 e 20
reverseFirstNonEmptyPacketSize	entre 6 e 7	entre 4 e 16
firstEightNonEmptyPacketDirections	aa/...	aa/...

Com base nestas características foi criado uma especificação para ataques em rajada a URI válidos. Na tabela 4.14 é demonstrado a especificação geral para este tipo de ataque.

Tabela 4.14: Especificação para ataque em rajada a URI válido

IE	Valor
ProtocolIdentifier	17
destinationTransportPort	5683
flowEndReason	active/idle
octetTotalCount	octetTotalCount / packetTotalCount > 32
reverseOctetTotalCount	reverseOctetTotalCount / reversePacketTotalCount > 32
packetTotalCount	>= 1 e >= reversePacketTotalCount
reversePacketTotalCount	>=1
dataByteCount	dataByteCount/ packetTotalCount > 4
reverseDataByteCount	reverseDataByteCount / reversePacketTotalCount > 4
nonEmptyPacketCount	>= 1 e >= reverseNonEmptyPacketCount
reverseNonEmptyPacketCount	>= 1
firstEightNonEmptyPacketDirections	!= aa

4.3.2 Especificações para CoAPS

Ao contrário das comunicações realizadas apenas com o protocolo aplicacional CoAP, as comunicações realizadas com CoAPS, obrigatoriamente têm de passar por um processo de troca de mensagens entre o cliente e o servidor para estabelecer uma sessão segura. Caso este processo por alguma razão não funcione, a comunicação é imediatamente cancelada e não é estabelecido uma sessão entre o cliente e o servidor. Desta forma, o DTLS auxilia na atenuação de ataques *Net Scan* e ataques DoS. Como abordado na secção 2.6.3.2 o DTLS insere-se na pilha protocolar entre a camada de Transporte e a camada de Aplicação pelo que o estabelecimento de uma sessão DTLS não cria uma sobrecarga adicional no dispositivo final. De salientar que com a utilização do DTLS, todas as mensagens CoAP são cifradas.

Assim, nesta subsecção será analisado os fluxos referentes ao protocolo aplicacional CoAPS, visando encontrar as especificações, tanto para os fluxos com classificação normal, como para os fluxos com classificação anómala, sendo estes, posteriormente classificados por tipo de ataque. Adicionalmente será abordado também apenas o estabelecimento de uma sessão válida em DTLS, de forma a mitigar ataques de *Net Scan* e ataques DoS.

Com base na RFC 7252 [2] os fluxos de tráfego IP para o protocolo aplicacional CoAPS para este cenário de testes, terá à priori as seguintes características:

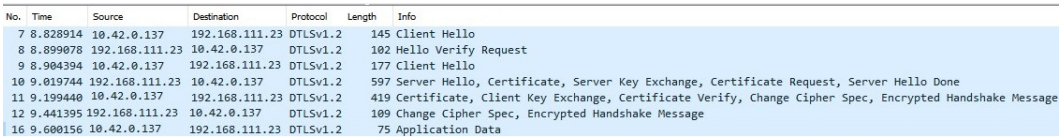
- CoAPS utiliza por definição o protocolo UDP e o porto 5684;
- Cabeçalho do protocolo CoAP é de 4 *bytes*;
- Utilização das seguintes mensagens, GET e POST;
- Utilização do protocolo DTLS como uma camada de segurança adicional.

4.3.2.1 *Handshake* DTLS

O estabelecimento de uma sessão DTLS é essencial para o correto funcionamento na troca de mensagens com segurança. Este método de segurança adicional auxilia na prevenção de ataques como o *Net Scan* e ataques DoS. Contudo, ao utilizar um IDS baseado em análise de fluxos, estes podem por alguma razão, ficarem incompletos, isto é, aquando a exportação por parte do YAF, o *handshake* poderá encontrar-se a meio, ficando a troca de mensagens do estabelecimento de sessão, separadas por fluxos. Assim, de forma a precaver este tipo de situações que podem surgir por congestionamento na rede ou por o *buffer* já se encontrar cheio para exportação, será analisado a troca de mensagens. O facto de atenuar ataques DoS, surge como mais um fator para a realização desta análise. De salientar, que estas especificações não englobam as mensagens CoAP, pois sem uma sessão estabelecida não existe comunicação entre o cliente e o servidor. Desta forma, o estabelecimento de uma sessão DTLS não provoca sobrecarga adicional nos dispositivos.

Aqui será analisado o estabelecimento de uma sessão válida em DTLS. Na figura 4.29 é demonstrado o comportamento para o estabelecimento de uma sessão válida. Pode-se constatar que o processo é iniciado através de um *Client Hello* por parte do cliente. Por sua vez, o servidor responde com um *Hello Verify Request* e o cliente volta a enviar novamente um *Client Hello*. Contudo, esta última mensagem já contem uma *cookie*, previamente recebida através do servidor. Com estes dois passos, o DTLS consegue assim resolver o problema que o TLS tinha com protocolos sem conexão, como é o caso do UDP. Os pacotes seguintes, são os pacotes que ocorrem também numa sessão TLS, onde o cliente e o servidor negoceiam os certificados ou chaves pré-partilhas que vão utilizar para o estabelecimento da sessão. É também negociado o tipo de cifra para a cifragem dos pacotes. No último passo (*Application Data*), a sessão já se encontra estabelecida pelo que o cliente e o servidor já podem trocar mensagens.

Através da figura 4.30, pode-se verificar que o tamanho total dos pacotes enviados pelo cliente são de 699 *bytes*. O tamanho total dos pacotes recebidos são de 766 *bytes*.



No.	Time	Source	Destination	Protocol	Length	Info
7	8.828914	10.42.0.137	192.168.111.23	DTLSv1.2	145	Client Hello
8	8.899078	192.168.111.23	10.42.0.137	DTLSv1.2	102	Hello Verify Request
9	8.904394	10.42.0.137	192.168.111.23	DTLSv1.2	177	Client Hello
10	9.019744	192.168.111.23	10.42.0.137	DTLSv1.2	597	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
11	9.199440	10.42.0.137	192.168.111.23	DTLSv1.2	419	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
12	9.441395	192.168.111.23	10.42.0.137	DTLSv1.2	109	Change Cipher Spec, Encrypted Handshake Message
16	9.600156	10.42.0.137	192.168.111.23	DTLSv1.2	75	Application Data

Figura 4.29: Análise ao comportamento de uma sessão válida - *Handshake* DTLS

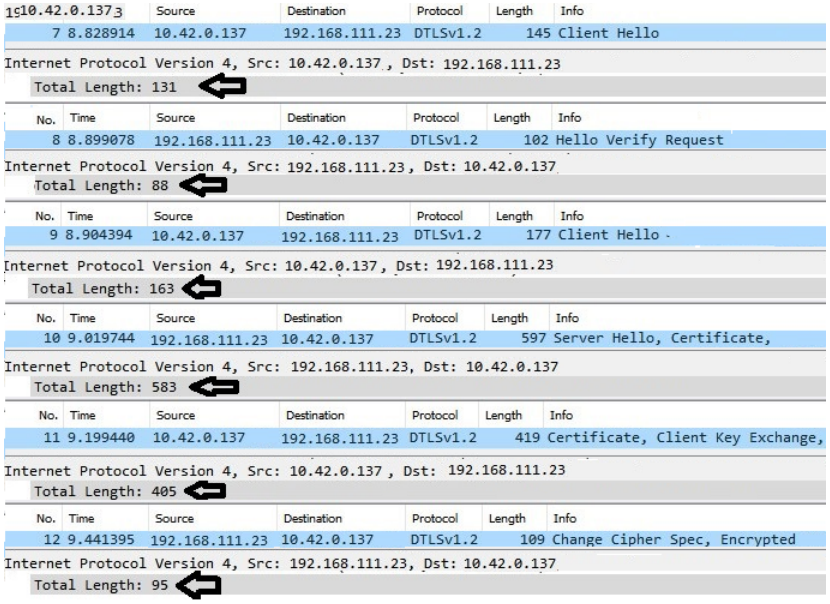


Figura 4.30: Estabelecimento de uma sessão DTLS

A figura 4.31 demonstra a informação recolhida dos fluxos utilizando o RapidMiner. De salientar que como o estabelecimento de uma sessão DTLS é independente ao protocolo aplicacional CoAP, neste caso para o RapidMiner os endereços de IP são ignorados, pois não são essenciais para a demonstração.

R..	Prot...	dst_port	packetTotal...	octetTotal...	reversePacket...	reverseOctet...	maxPacket...	ReverseMax...	firstNonEmpty...	ReverseFirstNon...	firstEightNon..
1	UDP	5684	3	699	3	766	377	555	103	60	aa

Figura 4.31: Resultados RapidMiner para estabelecimento de uma sessão DTLS

Analisando os resultados apresentados na figura 4.31, pode-se concluir que o tamanho dos pacotes enviados é de 699 *bytes* e o tamanho dos pacotes recebido é de 766 *bytes*. Em relação aos valores dos IE *maxPacketSize* e *reverseMaxPacketSize* são de 377 e 555, respetivamente. Estes valores representam o tamanho do *payload* em *bytes* do maior pacote enviado e o maior pacote recebido. Para os valores dos IE *firstNonEmptyPacketSize* e *reverseFirstNonEmptyPacketSize* são de 103 e 60, respetivamente. Estes valores representam o tamanho do *payload* em *bytes* do primeiro pacote enviado e do primeiro pacote recebido. Por último, o valor do IE *firstEightNonEmptyPacketDirections* é "aa". Este valor corresponde à direção dos 8 primeiros pacotes não vazios, em hexadecimal.

Na tabela 4.15 é apresentado toda a informação de cada fluxo referente ao *handshake* do DTLS.

Tabela 4.15: Especificação geral dos registos de fluxos para o estabelecimento de uma sessão DTLS

IE	*Handshake* DTLS
ProtocolIdentifier	17
destinationTransportPort	5684
flowEndReason	active
octetTotalCount	>= 699
reverseOctetTotalCount	>= 766
packetTotalCount	>= 3
reversePacketTotalCount	>= 3
dataByteCount	>= 615
reverseDataByteCount	>= 636
maxPacketSize	>= 377
reverseMaxPacketSize	>= 555
smallPacketCount	= 2
reverseSmallPacketCount	= 2
nonEmptyPacketCount	>= 3
reverseNonEmptyPacketCount	>= 3
firstNonEmptyPacketSize	103
reverseFirstNonEmptyPacketSize	60
firstEightNonEmptyPacketDirections	aa

Com base nestas características foi criado uma especificação para o estabelecimento de uma sessão válida. Na tabela 4.16 é demonstrado a especificação geral para o estabelecimento de uma sessão DTLS.

Tabela 4.16: Especificação *handshake* DTLS

IE	Valor
ProtocolIdentifier	17
destinationTransportPort	5684
flowEndReason	active/idle
octetTotalCount	octetTotalCount / packetTotalCount >= 233
reverseOctetTotalCount	reverseOctetTotalCount / reversePacketTotalCount >= 255
packetTotalCount	>= 3 e >= reversePacketTotalCount
reversePacketTotalCount	>= 3
maxPacketSize	>= 135
reverseMaxPacketSize	= 555
firstEightNonEmptyPacketDirections	aa

4.3.2.2 Fluxos de tráfego IP classificados como normais

Após o estabelecimento válido do *handshake* DTLS, o cliente e o servidor já podem trocar mensagens entre si utilizando o protocolo CoAP. De realçar de novo que estas mensagens são cifradas.

Através da figura 4.32, é demonstrado o comportamento para os fluxos normais utilizando DTLS. Ao contrário do comportamento no tráfego de fluxos IP classificados como normais, utilizando apenas CoAP, onde no método POST o cliente envia uma mensagem e o servidor responde, ao utilizar o DTLS as mensagens são cifradas e, isso implica um aumento do seu tamanho, fazendo com que para o método POST, devido ao tamanho da mensagem, esta seja divida em duas. Assim, para o método POST utilizando DTLS, o cliente envia um total de duas mensagens e o servidor outras duas. De salientar que os pedidos realizados são ao URI */temperature*, contudo como o tráfego de fluxos IP é cifrado, não é possível perceber através da *Info*, qual o método a ser utilizando nem qual o URI, pois o resultado apresentado é *Application Data*.

No.	Time	Source	Destination	Protocol	Length	Info
138169	31.917764	10.42.0.137	192.168.111.23	DTLSv1.2	95	Application Data
138188	31.921606	192.168.111.23	10.42.0.137	DTLSv1.2	87	Application Data
236398	60.038318	10.42.0.46	192.168.111.23	DTLSv1.2	99	Application Data
236438	60.043409	192.168.111.23	10.42.0.46	DTLSv1.2	75	Application Data
236441	60.044199	192.168.111.23	10.42.0.46	DTLSv1.2	87	Application Data
236487	60.087071	10.42.0.46	192.168.111.23	DTLSv1.2	75	Application Data

Figura 4.32: Análise ao comportamento de uma sessão válida - GET e POST

Através da figura 4.33 pode-se verificar que o tamanho para o pacote enviado como o método GET é de 81 *bytes* e o tamanho do pacote recebido é de 73 *bytes*.

No.	Time	Source	Destination	Protocol	Length	Info
138169	31.917764	10.42.0.137	192.168.111.23	DTLSv1.2	95	Application Data

Internet Protocol Version 4, Src: 10.42.0.137, Dst: 192.168.111.23

Total Length: 81 ⇦

No.	Time	Source	Destination	Protocol	Length	Info
138188	31.921606	192.168.111.23	10.42.0.137	DTLSv1.2	87	Application Data

Internet Protocol Version 4, Src: 192.168.111.23, Dst: 10.42.0.137

Total Length: 73 ⇦

Figura 4.33: CoAP utilizando DTLS - Método GET

Como referenciado anteriormente, o método POST contem o dobro dos pacotes enviados e recebidos. Através da figura 4.34, pode-se verificar que o tamanho para o primeiro pacote enviado do cliente é de 85 *bytes*, o primeiro pacote enviado pelo servidor é de 61 *bytes* e o segundo é de 73 *bytes*. Por último, o segundo pacote enviado pelo cliente é de 61 *bytes*. No total perfaz 146 *bytes* enviados e 134 *bytes* recebidos.

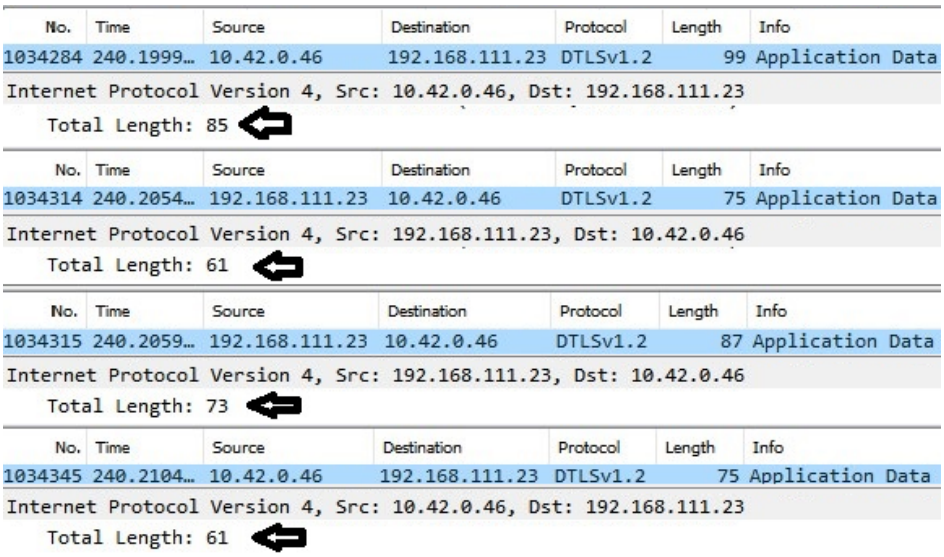


Figura 4.34: CoAP utilizando DTLS - Método POST

A figura 4.35 demonstra a informação recolhida dos fluxos utilizando o RapidMiner.

Row ...	Protocol	src_addr	src_port	dst_addr	dst_port	flowEndRe...	packetTotal...	octetTotalC...	reversePac...	reverseOcte...	firstEightNo...
1	UDP	10.42.0.137	42844	192.168.111.23	5684	active	6	486	6	438	aa
2	UDP	10.42.0.137	42844	192.168.111.23	5684	active	6	486	6	438	aa
3	UDP	10.42.0.137	42844	192.168.111.23	5684	active	6	486	6	436	aa
18	UDP	10.42.0.46	48651	192.168.111.23	5684	active	12	876	12	804	66
19	UDP	10.42.0.46	48651	192.168.111.23	5684	active	12	876	12	804	66
20	UDP	10.42.0.46	48651	192.168.111.23	5684	active	12	874	12	802	66

Figura 4.35: Resultados RapidMiner para mensagens GET utilizando DTLS

Analisando os resultados apresentados nas figuras 4.35, pode-se concluir que os fluxos com origem no endereço IP 10.42.0.137 e destino 192.168.111.2 são referentes a fluxos de tráfego IP com o método GET. Por sua vez, os fluxos com origem no endereço IP 10.42.0.46 e destino 192.168.111.23 são referentes a fluxos de tráfego IP com o método POST. Ambos as comunicações são realizadas ao porto 5684.

Em relação ao método GET, este apenas envia e recebe apenas um pacote, enquanto o método POST envia e recebe o dobro dos pacotes. Utilizando o método GET, o número total em *bytes* de todos os pacotes enviados é de 486 *bytes*. Se dividirmos pelos 6 pacotes dá um total de 81*bytes*. Aplicando o mesmo método para os pacotes recebidos, perfaz um total de 73 *bytes* (resultados obtidos na figura 4.33). Utilizando o método POST, o número total de *bytes* de todos os pacotes enviados é de 876 *bytes*. Se dividirmos pelos 12 pacotes dá um total de 73 *bytes*. Aplicando o mesmo método para os pacotes recebidos, perfaz um total de 67 *bytes* (resultados obtidos na figura 4.34).

Como para a realização das especificações para os fluxos de tráfego IP enviado é obtido através da seguinte forma, *octetTotalCount / packetTotalCount* e para os recebidos é obtido através da seguinte forma, *reverseOctetTotalCount / reversePacketTotalCount*, o RapidMiner faz a média, ou seja, o tamanho para o pacote enviado passa a ser de 73 *bytes* e o tamanho para o pacote recebido passa a ser de 67 *bytes*.

Ambos os métodos têm o valor de *active/idle* no IE *flowEndReason*. Este resultado indica a razão do término do fluxo.

Por último, para o método GET o valor no IE *firstEightNonEmptyPacketDirections* é de "aa", enquanto que para o método POST é de 66. Este valor corresponde à direção dos 8 primeiros pacotes não vazios, em hexadecimal.

Na tabela 4.17 é apresentado toda a informação de cada fluxo referente aos fluxos normais, utilizando os métodos GET e POST sobre DTLS.

Tabela 4.17: Especificação geral dos registos de fluxos normais CoAP sobre DTLS

IE	CoAPS	
	GET	**POST**
ProtocolIdentifier	17	17
destinationTransportPort	5684	5684
flowEndReason	active/idle	active/idle
octetTotalCount	486	876
reverseOctetTotalCount	434	804
packetTotalCount	6	12
reversePacketTotalCount	6	12
dataByteCount	318	540
reverseDataByteCount	270	468
maxPacketSize	53	57
reverseMaxPacketSize	45	45
smallPacketCount	6	6
reverseSmallPacketCount	12	12
nonEmptyPacketCount	6	6
reverseNonEmptyPacketCount	12	12
firstNonEmptyPacketSize	53	57
reverseFirstNonEmptyPacketSize	45	33
firstEightNonEmptyPacketDirections	aa	66

Com base nestas características foi criada uma especificação para o estabelecimento de uma sessão válida, utilizando os métodos GET e POST. Na tabela 4.16 é demonstrado a especificação geral para o estabelecimento de uma sessão DTLS utilizando os métodos GET e POST.

Tabela 4.18: Especificação geral dos registos de fluxos CoAP sobre DTLS

IE	Valor
ProtocolIdentifier	17
destinationTransportPort	5684
flowEndReason	active/idle
octetTotalCount	octetTotalCount / packetTotalCount > 69
reverseOctetTotalCount	reverseOctetTotalCount / reversePacketTotalCount >= 67
packetTotalCount	>= 1 e >= reversePacketTotalCount
reversePacketTotalCount	>= 1
dataByteCount	dataByteCount / packetTotalCount > 41
reverseDataByteCount	reverseDataByteCount / reversePacketTotalCount >= 39
reverseMaxPacketSize	>= 39
nonEmptyPacketCount	>= 1 e >= reverseNonEmptyPacketCount
reverseNonEmptyPacketCount	>= 1
firstEightNonEmptyPacketDirections	aa/66

4.3.2.3 Fluxos de tráfego IP classificados como anómalos

De forma a classificar os fluxos anómalos por tipo de ataque, estes também terão as suas respetivas especificações. Serão analisados ataques do tipo *Net Scan*, de forma a evitar ataques de mapeamento na rede, ataques a URI inválidos, de forma a evitar tentativas de acesso a informação não autorizada e, ataques em rajada a URI validos e inválidos, de forma a evitar ataques de negação de serviço. De seguida, será abordado a forma como foram criadas as especificações para cada tipo de ataque.

Ataques de NET SCAN

Como abordado anteriormente, os ataques de *Net Scan* são ataques utilizados para mapear uma rede de forma a permitir ao atacante perceber que endereços IP existem na rede. Para a realização deste tipo de ataques, foram novamente utilizadas as ferramentas NMAP e HPING.

Com recurso ao Wireshark é realizado uma análise ao comportamento dos ataques através das ferramentas NMAP e HPING. Através da figura 4.36 pode-se observar que o nó atacante gera pedidos para o servidor mas não recebe qualquer resposta. Na figura 4.37 pode-se observar uma tentativa de mapeamento, tentando descobrir informações sobre o servidor CoAP. Para este ataque, o nó atacante já recebe uma resposta, contudo não consegue dar seguimento ao *Handshake* DTLS. Isto acontece porque na resposta

do servidor é enviado uma *cookie* e o nó atacante ao efetuar um novo pedido *Client Hello*, não envia essa mesma *cookie*.

No.	Time	Source	Destination	Protocol	Length	Info
12509	342.427654	10.42.0.70	192.168.111.23	UDP	42	3750 → 5684 Len=0
12510	342.529834	10.42.0.70	192.168.111.23	UDP	42	3751 → 5684 Len=0
12512	342.628044	10.42.0.70	192.168.111.23	UDP	42	3752 → 5684 Len=0

Figura 4.36: Análise ao comportamento do ataque HPING

No.	Time	Source	Destination	Protocol	Length	Info
13223	424.539617	10.42.0.70	192.168.111.23	DTLSv1.2	109	Client Hello
13226	424.545583	192.168.111.23	10.42.0.70	DTLSv1.2	102	Hello Verify Request

Figura 4.37: Análise ao comportamento do ataque NMAP

Através da figura 4.38 pode-se observar o tamanho dos pacotes referentes ao ataque *Net Scan* utilizando a ferramenta HPING. Para um pacote enviado, o seu tamanho é de 28 *bytes* e não existe pacotes recebidos.

No.	Time	Source	Destination	Protocol	Length	Info
11388	281.200599	10.42.0.70	192.168.111.23	UDP	42	3140 → 5684 Len=0

Internet Protocol Version 4, Src: 10.42.0.70, Dst: 192.168.111.23

Total Length: 28 ⇦

Figura 4.38: Ataque *Net Scan* utilizando a ferramenta HPING

Através das figuras 4.39, pode-se observar o tamanho dos pacotes referentes ao ataque *Net Scan* utilizando a ferramenta NMAP. Para um pacote enviado, o seu tamanho é de 95 *bytes* e o tamanho do pacote recebido é 88 *bytes*.

No.	Time	Source	Destination	Protocol	Length	Info
13223	424.539617	10.42.0.70	192.168.111.23	DTLSv1.2	109	Client Hello

Internet Protocol Version 4, Src: 10.42.0.70, Dst: 192.168.111.23

Total Length: 95 ⇦

No.	Time	Source	Destination	Protocol	Length	Info
13226	424.545583	192.168.1…	10.42.0.70	DTLSv1.2	102	Hello Verify Request

Internet Protocol Version 4, Src: 192.168.111.23, Dst: 10.42.0.70

Total Length: 88 ⇦

Figura 4.39: Ataque *Net Scan* utilizando a ferramenta NMAP

Posto isto, pode-se constatar que um ataque *Net Scan*, utilizando a ferramenta HPING ou NMAP, não consegue estabelecer uma sessão DTLS válida, pelo que nunca chegam a comunicar com o alvo.

A figura 4.40 demonstra a informação recolhida dos fluxos utilizando o RapidMiner.

Ro...	Protocol	src_addr	dst_addr	dst_port	packetTotal...	octetTotalC...	reversePa...	reverseOcte...	flowEndRea...
1	UDP	10.42.0.70	192.168.111.23	5684	1000	28000	0	0	eof
2	UDP	10.42.0.70	192.168.111.23	5684	94	2632	0	0	eof
3	UDP	10.42.0.70	192.168.111.23	5684	94	2632	0	0	eof
Ro...	Protocol	src_addr	dst_addr	dst_port	packetTotal...	octetTotalC...	reversePa...	reverseOcte...	flowEndRea...
4	UDP	10.42.0.70	192.168.111.23	5684	1	95	1	88	idle
5	UDP	10.42.0.70	192.168.111.23	5684	1	95	1	88	idle
6	UDP	10.42.0.70	192.168.111.23	5684	1	95	1	88	idle

Figura 4.40: Resultados RapidMiner para ataque *Net Scan* utilizando as ferramentas HPING e NMAP

Analisando os resultados apresentados na figura 4.40, pode-se concluir que os fluxos com origem no endereço IP 10.42.0.70 e destino 192.168.111.23 são referentes a fluxos de tráfego IP realizados pelo nó atacante. Relativamente à primeira parte da figura, esta representa os resultados utilizando a ferramenta HPING e a segunda parte representa os resultados utilizando a ferramenta NMAP. Independentemente da ferramenta utilizada, pode-se verificar que os ataques são realizados ao porto 5684. Utilizando a ferramenta HPING, o tamanho de um pacote enviado é de 28 *bytes* e o tamanho de um pacote recebido é 0 (*reverseOctetTotalCount*) (resultados obtidos na figura 4.38). Utilizando a ferramenta NMAP, o tamanho de um pacote enviado é de 95 *bytes* e o tamanho de um pacote recebido é de 88 *bytes* (resultados obtidos na figura 4.39).

Em relação ao IE *flowEndReason*, o ataque utilizando a ferramenta HPING tem o valor de "eof", enquanto o ataque utilizando a ferramenta NMAP tem o valor de "idle". Este resultado indica a razão do término do fluxo. De salientar que nenhum dos ataques consegue estabelecer uma sessão válida DTLS.

Com base nestas características foi criado uma especificação para ataques *Net Scan*. Na tabela 4.19 é demonstrado a especificação geral para este tipo de ataques.

Tabela 4.19: Especificação para ataques *Net Scan*

IE	Valor
ProtocolIdentifier	17
destinationTransportPort	5684
flowEndReason	idle/eof
octetTotalCount	octetTotalCount / packetTotalCount <= 95
reverseOctetTotalCount	reverseOctetTotalCount / reversePacketTotalCount <= 88
reversePacketTotalCount	= 0

Ataques a URI Inválidos com sessão DTLS estabelecida

Os ataques a URI inválidos visam tentar obter dados sobre os dispositivos. Desta forma é essencial evitar que o atacante ganhe acesso a informações que não tem autorização.

Aqui serão analisados os ataques a URI inválidos com sessão DTLS estabelecida. Novamente, são utilizando os métodos GET e POST na realização dos ataques. A razão para esta análise do tráfego de fluxos IP, cuja sessão DTLS já se encontra estabelecida, prende-se pelo facto de evitar ataques, caso o atacante consiga obter acesso não autorizado a um dispositivo legitimo para o servidor e, este já se encontre com uma sessão DTLS estabelecida, podendo assim realizar ataques.

Na figura 4.41 é demonstrado o comportamento dos ataques a URI inválidos utilizando os métodos GET e POST. Pode-se observar que o comportamento é idêntico ao comportamento observado para os ataques a URI inválidos sem DTLS, pois para ambos os métodos utilizados, o cliente envia um pacote e recebe outro, o que perfaz uma relação de 1:1. De salientar que os pedidos realizados são ao URI */temp*, contudo como o tráfego de fluxos IP é cifrado, não é possível perceber através da *Info*, qual o método a ser utilizando nem qual o URI, pois o resultado apresentado é *Application Data*.

No.	Time	Source	Destination	Protocol	Length	Info
83	10.334507	10.42.0.46	192.168.111.23	DTLSv1.2	92	Application Data
84	10.338980	192.168.111.23	10.42.0.46	DTLSv1.2	83	Application Data
159	20.083586	10.42.0.137	192.168.111.23	DTLSv1.2	88	Application Data
160	20.089075	192.168.111.23	10.42.0.137	DTLSv1.2	83	Application Data

Figura 4.41: Análise ao comportamento de ataques a URI inválidos com DTLS - GET e POST

Através da figura 4.42 pode-se verificar que o tamanho para o pacote enviado com o método GET é de 74 *bytes* e o tamanho do pacote recebido é de 69 *bytes*.

No.	Time	Source	Destination	Protocol	Length	Info
159	20.083586	10.42.0.137	192.168.111.23	DTLSv1.2	88	Application Data

Internet Protocol Version 4, Src: 10.42.0.137, Dst: 192.168.111.23
Total Length: 74 ⇦

No.	Time	Source	Destination	Protocol	Length	Info
160	20.089075	192.168.111.23	10.42.0.137	DTLSv1.2	83	Application Data

Internet Protocol Version 4, Src: 192.168.111.23, Dst: 10.42.0.137
Total Length: 69 ⇦

Figura 4.42: Ataque a URI inválido - GET

Através da figura 4.43 pode-se verificar que o tamanho para o pacote enviado com o método GET é de 78 *bytes* e o tamanho do pacote recebido é de 69 *bytes*.

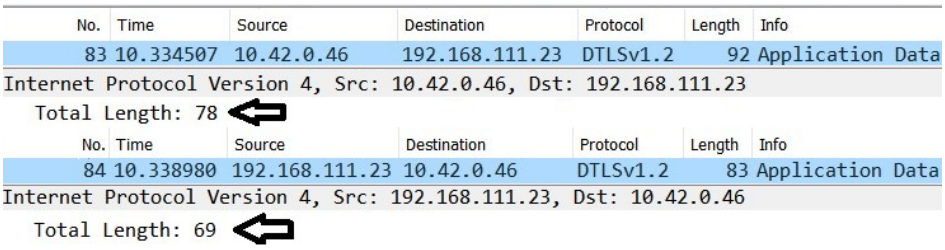


Figura 4.43: Ataque a URI inválido - POST

A figura 4.44 demonstra a informação recolhida dos fluxos utilizando o RapidMiner.

Row No.	Protocol	src_addr	src_port	dst_addr	dst_port	packetTotal...	octetTotal...	reversePacket...	reverseOctet...	ReverseMaxPacket...
1	UDP	10.42.0.46	53371	192.168.111.23	5684	6	467	6	414	41
2	UDP	10.42.0.137	56049	192.168.111.23	5684	6	444	6	414	41
3	UDP	10.42.0.46	53371	192.168.111.23	5684	6	468	6	414	41
4	UDP	10.42.0.137	56049	192.168.111.23	5684	6	444	6	414	41
5	UDP	10.42.0.46	53371	192.168.111.23	5684	6	468	6	414	41
6	UDP	10.42.0.137	56049	192.168.111.23	5684	6	444	6	414	41
7	UDP	10.42.0.46	53371	192.168.111.23	5684	6	467	6	414	41
8	UDP	10.42.0.137	56049	192.168.111.23	5684	6	444	6	414	41
9	UDP	10.42.0.46	53371	192.168.111.23	5684	6	467	6	414	41
10	UDP	10.42.0.137	56049	192.168.111.23	5684	6	444	6	414	41

Figura 4.44: Resultados RapidMiner para ataque a URI inválidos

Analisando os resultados apresentados na figura 4.44, pode-se concluir que os fluxos com origem no endereço IP 10.42.0.137 e destino 192.168.111.23 são referentes a fluxos de tráfego IP com o método GET. Por sua vez, os fluxos com origem no endereço IP 10.42.0.46 e destino 192.168.111.23 são referentes a fluxos de tráfego IP com o método POST.

Ambos os ataques são realizados ao porto 5684 e têm uma relação de 1:1. No ataque a URI inválido com sessão DTLS estabelecida utilizando o método GET, o tamanho de um pacote enviado é de 74 *bytes* e o tamanho de um pacote recebido é de 69 *bytes* (resultado obtido na figura 4.42). Para o ataque a URI inválido com sessão DTLS estabelecida utilizando o método GET, o tamanho de um pacote enviado é de 78 *bytes* e o tamanho de um pacote recebido é de 69 *bytes* (resultado obtido na figura 4.43).

Por último, ambos os ataques têm o valor de 41 no IE *reverseMaxPacktSize*. Este valor representa o tamanho do *payload* em *bytes* do maior pacote recebido.

Na tabela 4.20 é apresentado toda a informação de cada fluxo referentes ao ataque a URI inválido com sessão DTLS.

Tabela 4.20: Especificação geral dos registos de fluxos CoAP para ataques a URI inválidos sobre DTLS

IE	URI Inválido com DTLS	
	GET	POST
ProtocolIdentifier	17	17
destinationTransportPort	5684	5684
flowEndReason	active/idle	active/idle
octetTotalCount	444	467-468
reverseOctetTotalCount	414	414
packetTotalCount	6	6
reversePacketTotalCount	6	6
dataByteCount	276	279-300
reverseDataByteCount	246	246
maxPacketSize	46	50
reverseMaxPacketSize	41	41
smallPacketCount	6	6
reverseSmallPacketCount	6	6
nonEmptyPacketCount	6	6
reverseNonEmptyPacketCount	6	6
firstNonEmptyPacketSize	46	50
reverseFirstNonEmptyPacketSize	41	41
firstEightNonEmptyPacketDirections	aa	aa

Com base nestas caraterísticas foi criado uma especificação para ataques a URI inválidos com sessão DTLS estabelecida. Na tabela 4.23 é demonstrado a especificação geral para este tipo de ataque.

Tabela 4.21: Especificação para ataques a URI inválido com DTLS

IE	Valor
ProtocolIdentifier	17
destinationTransportPort	5684
flowEndReason	active/idle
octetTotalCount	octetTotalCount / packetTotalCount >= 69
reverseOctetTotalCount	= 69
reverseMaxPacketSize	= 41

Ataques em rajada a URI sem sessão DTLS estabelecida

Os ataques em rajada a URI inválidos e válidos são classificados como ataques DoS. Sendo um ataque DoS a ideia passa por inviabilizar a disponibilidade e capacidade do

alvo em conseguir responder aos pedidos realizados, num curto período de tempo, impedindo-o de responder ao tráfego e clientes legítimos.

Aqui serão analisados os ataques em rajada a URI inválidos e válidos sem sessão DTLS estabelecida. Novamente serão utilizados os métodos GET e POST. Na figura 4.45 é demonstrado o comportamento deste tipo de ataque. Pode-se observar que na realização de um ataque em rajada por um dispositivo legitimo para o servidor, este vai conseguir estabelecer uma sessão DTLS e o atacante pode realizar ataques. Podemos igualmente observar que existe um número elevado de pedidos *Client Hello* por parte do cliente e *Hello Verify Request* por parte do servidor de forma a tentar responder aos pedidos realizados pelo atacante. Como abordado anteriormente, as mensagens são cifradas, inviabilizando compreender se o ataque é realizado a um URI válido ou inválido, como também não é possível detetar se o método utilizado é GET ou POST.

No.	Time	Source	Destination	Protocol	Length	Info
2912	153.519492	10.42.0.46	192.168.111.23	DTLSv1.2	145	Client Hello
2913	153.524006	192.168.111.23	10.42.0.46	DTLSv1.2	102	Hello Verify Request
2914	153.525198	10.42.0.46	192.168.111.23	DTLSv1.2	145	Client Hello
2915	153.527551	192.168.111.23	10.42.0.46	DTLSv1.2	102	Hello Verify Request
2916	153.548213	10.42.0.46	192.168.111.23	DTLSv1.2	145	Client Hello
2917	153.550801	192.168.111.23	10.42.0.46	DTLSv1.2	102	Hello Verify Request
2918	153.616755	10.42.0.46	192.168.111.23	DTLSv1.2	145	Client Hello
2919	153.618826	10.42.0.46	192.168.111.23	DTLSv1.2	145	Client Hello
2920	153.619176	192.168.111.23	10.42.0.46	DTLSv1.2	102	Hello Verify Request
2921	153.620615	192.168.111.23	10.42.0.46	DTLSv1.2	102	Hello Verify Request
16880	197.089726	192.168.111.23	10.42.0.46	DTLSv1.2	597	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
16881	197.184538	10.42.0.46	192.168.111.23	DTLSv1.2	419	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
16882	197.184831	10.42.0.46	192.168.111.23	DTLSv1.2	145	Client Hello
16883	197.184968	10.42.0.46	192.168.111.23	DTLSv1.2	145	Client Hello
16892	197.222823	192.168.111.23	10.42.0.46	DTLSv1.2	109	Change Cipher Spec, Encrypted Handshake Message
16893	197.224646	192.168.111.23	10.42.0.46	DTLSv1.2	102	Hello Verify Request
32855	484.536065	10.42.0.46	192.168.111.23	DTLSv1.2	177	Client Hello
32856	484.539225	192.168.111.23	10.42.0.46	DTLSv1.2	102	Hello Verify Request
32857	484.556227	10.42.0.46	192.168.111.23	DTLSv1.2	177	Client Hello
32859	484.569566	192.168.111.23	10.42.0.46	DTLSv1.2	598	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
32866	485.571758	192.168.111.23	10.42.0.46	DTLSv1.2	598	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
32879	487.573322	192.168.111.23	10.42.0.46	DTLSv1.2	598	Server Hello, Certificate, Server Key Exchange, Certificate Request, Server Hello Done
32880	487.692620	10.42.0.46	192.168.111.23	DTLSv1.2	420	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec, Encrypted Handshake Message
32881	487.704383	192.168.111.23	10.42.0.46	DTLSv1.2	109	Change Cipher Spec, Encrypted Handshake Message
32882	487.724637	10.42.0.46	192.168.111.23	DTLSv1.2	92	Application Data
32883	487.727891	192.168.111.23	10.42.0.46	DTLSv1.2	83	Application Data

Figura 4.45: Análise ao comportamento de rajada a URI sem sessão DTLS estabelecida

A figura 4.46 demonstra a informação recolhida dos fluxos utilizando o RapidMiner.

Row ...	Protocol	dst_port	packetTotal...	octetTotalCo...	reversePac...	reverseOcte...	firstNonEmp...	maxPacket...	ReverseMaxP...	firstEightNo...
120	UDP	5684	7275	1026426	7240	663009	103	378	556	ff
121	UDP	5684	5020	658322	5021	442805	103	377	556	ff
122	UDP	5684	244	31964	244	21472	103	103	60	aa
123	UDP	5684	395	51745	395	34760	103	103	60	ff
124	UDP	5684	380	49780	380	33440	103	103	60	af
125	UDP	5684	301	39431	301	26488	103	103	60	aa
126	UDP	5684	68	8972	72	8811	103	135	555	cc
127	UDP	5684	24	3144	24	2112	103	103	60	aa
128	UDP	5684	468	61340	468	41184	103	135	60	ff

Figura 4.46: Análise ao comportamento de rajada a URI sem sessão DTLS estabelecida

Analisando os resultados apresentados na figura 4.46, pode-se concluir que ambos os ataques são realizados ao porto 5684. Ao contrario dos ataques em rajada em CoAP, os pacotes enviados e recebidos em algumas circunstancias já não mantêm uma relação de 1:1.

O tamanho do IE *firstNonEmptyPacketSize* é igual a 103 *bytes*. Este valor representa o tamanho do *payload* em *bytes* do primeiro pacote enviado e representa um pedido *Client Hello*. O tamanho do IE *maxPacketSize* tem diferentes valores. O valor 103 representa o primeiro *Client Hello*, o valor de 135 representa o segundo *Client Hello* e os valores compreendidos entre 377 e 378 representam a resposta do cliente validando o certificado e solicitando a cifra a utilizar. Este IE representa o tamanho do *payload* em *bytes* do maior pacote enviado. O tamanho do IE *reverseMaxPacketSize* também tem diferentes valores. O valor 60 representa um *Hello Verify Request* e os valores compreendidos entre 555 e 556 representam a resposta do servidor anunciando os certificados disponíveis e solicitando o certificado ao cliente. Este IE representa o tamanho do *payload* em *bytes* do maior pacote recebido.

Por último, o valor do IEs *firstEightNonEmptyPacketDirections* é diferente de "aa". Este valor corresponde à direção dos 8 primeiros pacotes não vazios, em hexadecimal.

Na tabela 4.22 é apresentado toda a informação de cada fluxo referente aos ataque em rajada a URI sem sessão DTLS estabelecida.

Tabela 4.22: Especificação geral dos registos de fluxos para ataques em rajada a URI sem sessão DTLS estabelecida

IE	URI sem sessão DTLS
ProtocolIdentifier	17
destinationTransportPort	5684
flowEndReason	active/idle
octetTotalCount	> 69
reverseOctetTotalCount	>= 60
packetTotalCount	>= 1
reversePacketTotalCount	>= 1
maxPacketSize	>= 103
reverseMaxPacketSize	= 60 ou >=555
firstNonEmptyPacketSize	>= 103
reverseFirstNonEmptyPacketSize	= 60 ou >= 555
firstEightNonEmptyPacketDirections	!= aa

Com base nestas caraterísticas foi criado uma especificação para ataques a URI sem sessão DTLS. Na tabela 4.23 é demonstrado a especificação geral para este tipo de ataque.

Tabela 4.23: Especificação para ataques a URI sem sessão DTLS

IE	Valor
ProtocolIdentifier	17
destinationTransportPort	5684
flowEndReason	active/idle
octetTotalCount	> 69
maxPacketSize	>= 103 ou >= 377
reverseMaxPacketSize	= 60 ou >= 555
firstNonEmptyPacketSize	= 103
firstEightNonEmptyPacketDirections	!= aa

Desta forma, para ataques em rajada a URI inválidos ou válidos sem sessão DTLS estabelecida, não é possível compreender que tipo de ataque está a ser realizado pelo atacante, contudo é possível compreender que um ataque está a acontecer, sendo posteriormente gerado um alerta. Isto acontece devido a uma limitação de IE existentes para obter uma maior informação, aliado à agregação dos pacotes em fluxos, isto é, os principais IE que nos podem auxiliar na deteção para este tipo de ataque, podem ter diversos valores, logo podem ter diversos significados, dificultando assim a deteção. A título de exemplo, se o IE *maxPacketSize* tiver o valor compreendido entre 377 e 378 e esse mesmo fluxo, conter mensagens cifradas CoAP (*Application Data*), não é possível perceber que estas mensagens estão inseridas no fluxo. Assim, para este tipo de ataque, podemos afirmar que a utilização de uma análise de fluxos omite informações que seriam relevantes para uma melhor deteção.

4.4 Plano de testes

Neste subcapítulo é apresentado o plano de testes implementado com a finalidade de avaliar a *framework* proposta, no que concerne ao funcionamento da aplicação IDS, à performance da deteção para o protocolo aplicacional CoAP e à performance da deteção para o protocolo aplicacional CoAP sobre DTLS. Numa primeira fase são apresentados os testes de funcionamento à aplicação IDS. Numa segunda fase são apresentados os testes de funcionamento para a deteção no protocolo aplicacional CoAP e, por último, numa terceira fase, são apresentados os testes de funcionamento para a deteção no protocolo aplicacional CoAP sobre DTLS.

Os teste foram realizados num cenário idêntico ao protótipo apresentado na figura 4.5, utilizando as especificações abordadas anteriormente no subcapítulo 4.3.

O processo inicial de funcionamento do cenário de testes decorreu em duas fases. Numa primeira fase foi estabelecido o sistema IoT para os clientes e servidores CoAP e CoAPS, de forma a trocarem mensagens relacionadas com a temperatura do ar e luminosidade do meio físico. Posteriormente, numa segunda fase, foi iniciado o sistema IDS proposto. Aqui, foi iniciado o YAF que é o responsável pela sonda no IDS, de forma a capturar os pacotes de rede trocados entre os dispositivos IoT, onde posteriormente serão agregados em registos de fluxos de tráfego e serão exportados para o módulo central do IDS. Este módulo encontra-se em funcionamento num *host* dedicado que contem um serviço de receção e armazenamento de registos de fluxos.

Existem alguns pressupostos de funcionamento do IDS no cenário de testes que devem ser considerados: [1]

- A sonda IDS é executada em modo *daemon*, por forma a garantir uma recolha contínua de pacotes de rede e respetiva exportação de fluxos de tráfego IP;
- A sonda vai agregar e exportar os fluxos de tráfego IP, presentes na memória *cache* do componente de agregação e exportação, através do protocolo IPFIX. A exportação tem a periodicidade de 1 minuto no caso de inatividade ou de 3 minutos em caso de atividade continua;
- A sonda vai exportar registos de fluxos com a lista de IE apresentados no subcapítulo 4.1.3. Desta forma é possível obter uma maior diversidade das caraterísticas das comunicações capturadas para uma melhor análise posterior;
- O módulo central do IDS recolhe os registos de fluxos de tráfego IP e armazena-os em ficheiros JSON;
- Sempre que exista novos registos de fluxos de tráfego IP armazenados, o componente de análise de fluxos, presente no módulo central do IDS, examina os registos de fluxos e classifica-os como fluxos normais ou anómalos. No caso dos registos de fluxos, forem classificados como anómalos, estes são igualmente classificados por tipos de ataque, nomeadamente ataques de *Net Scan*, inundação de pedidos válidos e inválidos CoAP e CoAPS e ataques DDoS;
- Sempre que é detetado um registo de fluxo anormal, é emitida e armazenada uma mensagem de alerta de intrusão através do protocolo *syslog*;
- Para além do tráfego normal CoAP utilizado pelos dispositivos IoT, é gerado tráfego anormal através de um dispositivo atacante que irá gerar ataques de *Net Scan* aos serviços CoAP. Por forma a reutilizar recursos, os dispositivos CoAP

clientes também irão geral tráfego anormal, procedendo à sua devida adaptação, para a realização de ataques de uso indevido, injeção de informação errada e ataques Dos nos serviços CoAP.

Estes pressupostos auxiliaram na realização dos testes, posteriormente abordados, com o intuito de validar a performance das especificações para a deteção de intrusões.

4.4.1 Teste de funcionamento à aplicação IDS

O primeiro teste realizado destinou-se a validar o funcionamento da aplicação IDS. Esta aplicação, desenvolvida em *python* tem como intuito realizar uma leitura e análise ao tráfego de fluxos IP armazenados no módulo central do IDS.

Para testar a aplicação IDS, apenas são analisados os fluxos de tráfego IP armazenados no módulo central do IDS. Estes fluxos, são provenientes do tráfego gerado entre clientes e servidores CoAP e CoAPS, sendo armazenados pelo módulo central após exportação realizada pela sonda através do IPFIX.

Posto isto, a aplicação IDS vai realizar uma leitura e análise em duas etapas. Numa primeira etapa é realizado uma leitura das especificações consideradas como sendo de comportamento normal, que se encontram numa base dados de uma diretoria dedicada para o efeito. Numa segunda etapa, caso exista novos fluxos, estes serão comparados com as especificações de comportamento normal. Nos casos em que os fluxos são classificados como anómalos, então é gerado um alerta de intrusão através do protocolo *syslog*.

Dessa análise, resulta um relatório onde consta a classificação atribuída a cada fluxo. De salientar que caso o fluxo seja classificado como anómalo, este é igualmente classificado por tipo de ataque.

Espera-se aqui, que a aplicação desenvolvida consiga classificar os registos de fluxos de tráfego IP normais e anómalos, classificando posteriormente os registos de fluxos de tráfego IP anómalos por tipo de ataque, nomeadamente ataques de *Net Scan*, URI inválidos e válidos e, ataques DoS para CoAP e CoAPS.

4.4.2 Teste de deteção para CoAP

O segundo teste realizado destina-se a validar o mecanismo de análise aos registos de fluxos de tráfego IP, de forma a analisar a *performance* das especificações apresentadas no subcapítulo 4.3, referentes ao protocolo aplicacional CoAP.

Aqui é pretendido validar duas situações: i) análise dos registos de fluxos de tráfego IP de comunicações CoAP normais e ii) análise dos registos de fluxos de tráfego IP de comunicaçõesCoAP anormais ou maliciosos, seguido de respetiva classificação por ataque.

O processo para testar i), é realizado em 4 etapas. Num primeira etapa são ativados todos os dispositivos IoT CoAP de forma a gerar troca de mensagens entre clientes e servidores. Através desta troca de dados, são gerados tráfego de fluxos normais. Numa segunda etapa é executado o YAF, visando capturar os pacotes de rede trocados nas comunicações realizadas entre, os clientes CoAP da rede interna e, os servidores CoAP que se encontram na Internet. Numa terceira etapa é executado, no módulo central do IDS, o *super_ mediator* com as especificações descritas no subcapítulo 4.3 de forma a descodificar e armazenar os registos de fluxos de tráfego IP exportados pela sonda através do IPFIX. Por fim, na quarta e última etapa, executou-se a aplicação IDS, no qual vai realizar uma leitura dos fluxos de tráfego IP que se encontraram armazenados em formato JSON no módulo central do IDS e, posteriormente realizar uma análise tendo em consideração as especificações descritas no subcapítulo 4.3.

Dessa análise, resulta um relatório onde consta a classificação atribuída a cada fluxo. De salientar que caso o fluxo seja classificado como anómalo, este é igualmente classificado por tipo de ataque.

O processo para testar ii), é igualmente realizado em 4 etapas. Num primeira etapa são ativados todos os dispositivos IoT CoAP de forma a gerar troca de mensagens entre clientes e servidores. Através desta troca de dados, são gerados tráfego de fluxos anómalos. Adicionalmente e utilizado um dispositivo atacante que irá gerar ataques de *Net Scan* aos serviços CoAP. Por forma a reutilizar recursos, os dispositivos CoAP clientes também irão geral tráfego anómalo, procedendo à sua devida adaptação, para a realização de ataques a URI inválidos e válidos e ataques Dos nos serviços CoAP. Numa segunda etapa é executado o YAF, visando capturar os pacotes de rede trocados nas comunicações realizadas entre, os clientes CoAP e CoAPS da rede interna e, os servidores CoAP e CoAPS que se encontram na Internet. Numa terceira etapa é exe-

cutado, no módulo central do IDS, o *super_mediator* com as especificações descritas no subcapítulo 4.3 de forma a descodificar e armazenar os registos de fluxos de tráfego IP exportados pela sonda através do IPFIX. Por fim, na quarta e última etapa, executou-se a aplicação IDS, no qual vai realizar uma leitura dos fluxos de tráfego IP que se encontraram armazenados em formato JSON no módulo central do IDS e, posteriormente realizar uma análise tendo em consideração as especificações descritas no subcapítulo 4.3.

Dessa análise, resulta um relatório onde consta a classificação atribuída a cada fluxo. De salientar que caso o fluxo seja classificado como anómalo, este é igualmente classificado por tipo de ataque.

Para estas duas situações espera-se que os fluxos de tráfego IP na sua maioria sejam bem classificados tendo em consideração as especificações descritas no subcapítulo 4.3. É expectável que para os registos de fluxos de tráfego IP CoAP normais, sejam bem classificados por parte da aplicação IDS. Em relação aos registos de fluxos de tráfego IP CoAP anómalos, é expectável que sejam bem classificados, exceto alguns registos de fluxos de tráfego IP provenientes dos ataques DoS a URI válidos.

De salientar que, para além das comunicações normais e anormais, existem ainda na rede interna um serviço de DNS e de NTP que também vão gerar tráfego entre a rede interna e a Internet.

4.4.3 Teste de deteção para CoAPS

O terceiro e último teste realizado, destina-se a validar o mecanismo de análise aos registos de fluxos de tráfego IP, de forma a analisar a *performance* das especificações apresentadas no subcapítulo 4.3, referentes ao protocolo aplicacional CoAPS, ou seja, CoAP sobre DTLS.

Aqui é pretendido validar duas situações: i) análise dos registos de fluxos de tráfego IP de comunicações CoAPS normais e ii) análise dos registos de fluxos de tráfego IP de comunicaçõesCoAPS anormais ou maliciosos, seguido de respetiva classificação por ataque.

O processo para testar i), é realizado em 4 etapas. Numa primeira etapa são ativados todos os dispositivos IoT CoAPS, incluindo clientes e servidores. Ao contrário dos

testes realizados apenas com o CoAP, ao utilizar-se o DTLS como uma camada adicional de segurança, antes de ocorrer a troca de mensagens entre clientes e servidores, ambos têm de passar por um processo chamado de *handshake* DTLS, onde ocorre o estabelecimento de uma sessão válida. Após o estabelecimento de uma sessão válida os clientes e servidores já podem trocar mensagens entre si. Através desta troca de dados, são gerados tráfego de fluxos normais. De salientar que estes fluxos são cifrados de forma a garantir a privacidade do utilizador. Numa segunda etapa é executado o YAF, visando capturar os pacotes de rede trocados nas comunicações realizadas entre, os clientes CoAPS da rede interna e, os servidores CoAPS que se encontram na Internet. Numa terceira etapa é executado, no módulo central do IDS, o *super_ mediator* com as especificações descritas no subcapítulo 4.3 de forma a descodificar e armazenar os registos de fluxos de tráfego IP exportados pela sonda através do IPFIX. Por fim, na quarta e última etapa, executou-se a aplicação IDS, no qual vai realizar uma leitura dos fluxos de tráfego IP que se encontraram armazenados em formato JSON no módulo central do IDS e, posteriormente realizar uma análise tendo em consideração as especificações descritas no subcapítulo 4.3.

Dessa análise, resulta um relatório onde consta a classificação atribuída a cada fluxo. De salientar que caso o fluxo seja classificado como anómalo, este é igualmente classificado por tipo de ataque.

O processo para testar ii), é realizado em 4 etapas. Numa primeira etapa são ativados todos os dispositivos IoT CoAPS, incluindo clientes e servidores. Ao contrário dos testes realizados apenas com o CoAP, ao utilizar-se o DTLS como uma camada adicional de segurança, antes de ocorrer a troca de mensagens entre clientes e servidores, ambos têm de passar por um processo chamado de *handshake* DTLS, onde ocorre o estabelecimento de uma sessão válida. Após o estabelecimento de uma sessão válida os clientes e servidores já podem trocar mensagens entre si. Através desta troca de dados, são gerados tráfego de fluxos anómalos. Adicionalmente e utilizado um dispositivo atacante que irá gerar ataques de *Net Scan* aos serviços CoAPS. Por forma a reutilizar recursos, os dispositivos CoAPS clientes também irão geral tráfego anómalo, procedendo à sua devida adaptação, para a realização de ataques a URI inválidos e válidos e ataques Dos nos serviços CoAPS. De salientar que estes fluxos são cifrados de forma a garantir a privacidade do utilizador. Numa segunda etapa é executado o YAF, visando capturar os pacotes de rede trocados nas comunicações realizadas entre, os clientes CoAPS da rede interna e, os servidores CoAPS que se encontram na Internet. Numa terceira etapa é executado, no módulo central do IDS, o *super_ mediator* com as especificações descritas no subcapítulo 4.3 de forma a descodificar e armazenar os registos de fluxos de tráfego IP exportados pela sonda através do IPFIX. Por fim, na

quarta e última etapa, executou-se a aplicação IDS, no qual vai realizar uma leitura dos fluxos de tráfego IP que se encontraram armazenados em formato JSON no módulo central do IDS e, posteriormente realizar uma análise tendo em consideração as especificações descritas no subcapítulo 4.3.

Dessa análise, resulta um relatório onde consta a classificação atribuída a cada fluxo. De salientar que caso o fluxo seja classificado como anómalo, este é igualmente classificado por tipo de ataque.

Para estas duas situações espera-se que os fluxos de tráfego IP na sua maioria sejam bem classificados tendo em consideração as especificações descritas no subcapítulo 4.3. É expectável que para os registos de fluxos de tráfego IP CoAPS normais, sejam bem classificados por parte da aplicação IDS, mesmo com as mensagens cifradas. Em relação aos registos de fluxos de tráfego IP CoAP anómalos, é expectável que sejam bem classificados, exceto alguns registos de fluxos de tráfego IP cujo não têm sessão DTLS estabelecida.

De salientar que, para além das comunicações normais e anormais, existem ainda na rede interna um serviço de DNS e de NTP que também vão gerar tráfego entre a rede interna e a Internet.

4.5 Síntese

Neste capítulo apresentou-se o protótipo desenvolvido a fim de validar a *framework* proposta.

Pretendeu-se também, avaliar a *framewoek* proposta no que concerne à deteção de intrusões tendo como base as especificações previamente declaras. Esta bateria de testes pretendia validar a aplicação IDS quanto ao seu funcionamento, de forma a compreender se esta classificava os registos de fluxos de tráfego de uma forma correta. Pretendia igualmente validar a *performance* das especificações declaradas, tanto para CoAP como para CoAPS.

No próximo capítulo prceder-se-á apresentação e discussão dos resultados obtidos nos testes realizados.

Capítulo 5

Testes e Resultados

Neste capitulo são apresentados e discutidos os resultados obtidos aos testes efetuados com o intuito de validar a eficácia da *framework* proposta, no que diz respeito ao funcionamento da aplicação IDS, à avaliação da capacidade de deteção de intrusões, através das especificações para os fluxos de tráfego IP utilizando CoAP e CoAPS.

Através da realização da troca de mensagens entre clientes, servidores e o dispositivo atacante, ambos para CoAP e CoAPS foi criado um *dataset*. Este *dataset* é composto por tráfego normal, anómalo e tráfego Domain Name System (DNS) e NTP. Em relação ao tráfego normal, este foi gerado pelo protótipo após sete dias em operação, sem qualquer introdução de tráfego anómalo, de forma a garantir a total veracidade dos registos de fluxos de tráfego normais. Em relação ao tráfego anómalo, este foi gerado com os diversos ataques, nomeadamente com ataques de *Net Scan* utilizando as ferramentas de HPING e NMAP, ataques a URI inválidos e válidos e ataques em rajada, igualmente a URI inválidos e válidos. Por fim, os protocolos DNS e NTP geram tráfego entre a rede interna e a Internet.

Inicialmente, são apresentados os resultados obtidos pela aplicação IDS. Posteriormente são apresentados os resultados dos testes utilizando CoAP e, por último, são apresentados os resultados dos testes utilizando CoAPS. De salientar que os testes realizados consideram as especificações, apresentadas no subcapítulo 4.3.

A seleção dos registos de fluxos de tráfego para a realização dos testes, tiveram como base dispositivos IoT que integram o protótipo e que se conhece à posteriori que realizam comunicação normais ou ataques, consoante a análise pretendida.

5.1 Teste de funcionamento à aplicação IDS

Através do resultado deste teste pretende-se validar o protótipo relativamente ao funcionamento da aplicação IDS desenvolvida.

Para desenvolver este teste, foram utilizados e selecionados registos de fluxo de tráfego IP armazenados no módulo central do IDS. De salientar que estes registos de fluxos de tráfego IP, antes de serem armazenados no módulo central, passaram por um processo de troca de mensagens entre clientes, dispositivo ataque e servidores, ambos para CoAP e CoAPS. Posteriormente foram exportados para o módulo central, onde foram armazenados.

Posto isto, executa-se a aplicação IDS no qual realiza uma leitura e análise dos registos de fluxos de tráfego IP para CoAP e CoAPS, como base nas especificações apresentadas no subcapítulo 4.3. Tendo em consideração a análise realizada aos registos de fluxos de tráfego IP, estes são classificados como normais ou anómalos. Caso sejam classificados como anómalos, é especificado por tipo de ataque.

No final do processo de análise, recolhem-se os resultados obtidos pela aplicação IDS através de um relatório criado pela mesma. Através da figura 5.1, pode-se observar o resultado obtido na saída da aplicação de uma forma resumida, onde o administrador da rede pode observar quantos fluxos foram classificados como normais e anómalos, separados por CoAP e CoAPS. No caso dos fluxos anómalos, pode-se observar também quantos foram e qual o seu tipo.

```
ids4iot@elkserver:~/flows$ more coap.txt | grep COUNTERS
COUNTERS - 2 Normal + 4 Attacks ---------- 3 NET SCAN, 1 INVALID URI, 0 VALID URI
ids4iot@elkserver:~/flows$ more coaps.txt | grep COUNTERS
COUNTERS - 1 Normal + 1 DTLS Válido + 5 Attacks ---------- 2 NET SCAN, 2 INVALID URI, 1 BRUST
ids4iot@elkserver:~/flows$
```

Figura 5.1: Comando de saída da aplicação IDS - Resumo

Através da figura 5.2, pode-se observar o resultado obtido na saída da aplicação de uma forma detalhada, onde o administrador da rede pode observar a classificação de cada fluxo, isto é, se o fluxo foi classificado como normal, anómalo e caso seja anómalo, por tipo de ataque. Pode-se observar também que para os fluxos classificados como anómalos têm uma numeração. Esta numeração indica a posição de entrada do fluxo no ficheiro JSON. Desta forma, pretende-se facilitar a pesquisa pelo fluxo anómalo e, consequentemente uma maior rapidez na execução de medidas.

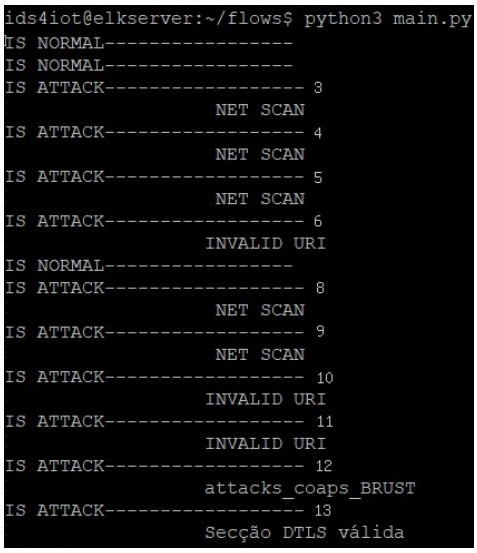


Figura 5.2: Comando de saída da aplicação IDS - Detalhado

Por fim, na figura 5.3 é apresentado o relatório, no qual é discriminado os resultados da análise realiza aos registos de fluxo de tráfego IP gerado pela aplicação IDS, onde é possível compreender quais são os fluxos classificados como normais ou anómalos.

NORMAL - {'flowStartMilliseconds': '2019-06-11 17:52:35.448', 'flowEndMilliseconds': '2019-06-11 17:55:05.546'
ATTACK - {'flowStartMilliseconds': '2019-06-12 17:51:32.103', 'flowEndMilliseconds': '2019-06-12 17:51:32.109',
ATTACK - {'flowStartMilliseconds': '2019-06-14 16:26:42.004', 'flowEndMilliseconds': '2019-06-14 16:29:12.195',
ATTACK - {'flowStartMilliseconds': '2019-09-04 16:54:58.840', 'flowEndMilliseconds': '2019-09-04 16:55:24.038',
NORMAL - {'flowStartMilliseconds': '2019-06-11 17:52:35.448', 'flowEndMilliseconds': '2019-06-11 17:55:05.546',

Figura 5.3: Conteúdo de um fluxo de tráfego IP

5.2 Testes e resultados obtidos para o protocolo aplicacional CoAP

Através do resultado deste teste pretende-se validar o mecanismo de análise aos registos de fluxos de tráfego IP, de forma a analisar a *performance* das especificações apresentadas no apresentadas no subcapítulo 4.3, referentes ao protocolo aplicacional CoAP.

Para desenvolver este teste, foram utilizados os registos de fluxo de tráfego IP CoAP armazenados no módulo central do IDS. De salientar que estes registos de fluxos de tráfego IP, antes de serem armazenados no módulo central, passaram por um processo de troca de mensagens entre clientes, dispositivo ataque e servidores CoAP. Posteriormente foram exportados para o módulo central, onde foram armazenados.

Como abordado no plano de testes, este conjunto de testes está dividido em duas situações: i) análise dos registos de fluxos de tráfego IP de comunicações IoT CoAP normais e ii) análise dos registos de fluxos de tráfego IP de comunicações IoT CoAP anómalos, seguido de respetiva classificação por ataque.

Os resultados são apresentados através da utilização de tabelas para cada uma das fases. Nelas estão discriminados o método utilizado na troca de mensagens, o número de registos de fluxos de tráfego que compõem o *dataset* utilizado, o número de registos de fluxos de tráfego referentes ao protocolo aplicacional, o número de registo de fluxos de tráfego que foram classificados como normais (RFN), número de registo de fluxos de tráfego que foram classificados como anómalos (RFA), taxa de falsos positivos (FP), taxa de falsos negativos (FN), taxa de verdadeiros positivos (TP) e por último a taxa de deteção (TD) para o teste efetuado.

5.2.1 Resultado aos fluxos de tráfego normais

De forma a avaliar e validar os registos de fluxos de tráfego IP para comunicações normais, foram efetuados testes em duas fases diferentes separados pelos métodos GET e POST. Numa primeira fase são selecionados alguns registos de fluxos normais de forma a conter um menor número de amostras para validação das especificações. Numa segunda fase, os resultados são utilizados todos os registos de fluxos de tráfego normais, independentemente do seu método. Estes registos de fluxos de tráfego encontram-se no *dataset*, anteriormente abordado. Estes ataques são realizados utilizando clientes CoAP.

Os resultados dos testes para os registos de tráfego CoAP normais são apresentados da seguinte forma. Na tabela 5.1 são apresentados os resultados dos testes referente à primeira fase e na tabela 5.2 são apresentados os resultados dos testes referentes à segunda fase que contem o *dataset* final apenas com tráfego normal.

Tabela 5.1: Resultado da deteção de fluxos normais CoAP por mensagem

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
GET	100	100	100	0	0	0	100	100
POST	100	100	100	0	0	0	100	100

Tabela 5.2: Resultado da deteção de fluxos normais CoAP

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
GET/POST	17260	2321	2321	0	0	0	100%	100%

5.2.2 Resultado aos fluxos de tráfego anómalos

De forma a avaliar e validar os registos de fluxos de tráfego IP para comunicações anómalas, foram efetuados testes em duas fases diferentes separados pelos métodos GET e POST para os ataques a URI e pelas ferramentas HPING e NMAP para os ataques de *Net Scan*. Numa primeira fase são selecionados alguns registos de fluxos normais de forma a conter um menor número de amostras para validação das especificações. Numa segunda fase, os resultados são utilizados todos os registos de fluxos de tráfego anómalos, independentemente do seu método ou ferramenta. Estes registos de fluxos de tráfego encontram-se no *dataset*, anteriormente abordado. Os ataques são categorizados em ataques de *Net Scan*, utilizando HPING e NMAP, ataques a URI inválidos e válidos, incluindo os ataques em rajada.

Resultado aos ataques de *Net Scan*

Considerando agora apenas os registos de fluxo de tráfego anómalos com ataques *Net Scan*, é pretendido validar a capacidade de deteção de intrusões do IDS. Estes ataques são realizados utilizando o dispositivo atacante.

Os resultados dos testes para os registos de tráfego CoAP anómalos para ataques *Net Scan* são apresentados da seguinte forma. Na tabela 5.3 são apresentados os resultados dos testes referentes à primeira fase e, na tabela 5.4 são apresentados os resultados dos testes referentes à segunda fase, que contem o *dataset* final apenas com tráfego anómalo.

Tabela 5.3: Resultado da deteção de fluxos com ataques *Net Scan* por ferramenta

Tipo de ferramenta	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
NMAP	100	100	0	100	0	0	100%	100%
HPING	100	100	0	100	0	0	100%	100%

Tabela 5.4: Resultado da deteção de fluxos com ataques *Net Scan*

Tipo de anomalia CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
Net Scan	28636	12564	0	12564	0	0	100%	100%

Resultado aos ataques a URI inválidos incluindo os ataques em rajada

Considerando agora apenas os registos de fluxo de tráfego anómalos com ataques a URI inválidos, incluindo os ataques em rajada, é pretendido validar a capacidade de deteção de intrusões do IDS. Estes ataques são realizados utilizando clientes CoAP.

Os resultados dos testes para os registos de tráfego CoAP anómalos para ataques a URI inválidos incluindo os ataques em rajada são apresentados da seguinte forma. Nas tabelas 5.5 e 5.6 são apresentados os resultados dos testes referentes à primeira fase e, na tabela 5.7 são apresentados os resultados dos testes referentes à segunda fase, que contem o *dataset* final apenas com tráfego anómalo.

Tabela 5.5: Resultado da deteção de fluxos com ataques a URI inválido por mensagem

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
GET	40	20	0	20	0	0	100%	100%
POST	40	20	0	20	0	0	100%	100%

Tabela 5.6: Resultado da deteção de fluxos com ataques em rajada a URI inválido por mensagem

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
GET	100	20	0	20	0	0	100%	100%
POST	100	20	0	20	0	0	100%	100%

Tabela 5.7: Resultado da deteção de fluxos com ataques a URI inválido incluindo o ataque em rajada

Tipo de anomalia CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
URI Inválido	34473	637	0	637	0	0	100%	100%

Resultado aos ataques em rajada a URI válido

Considerando agora apenas os registos de fluxo de tráfego anómalos com ataques em rajada a URI válido, é pretendido validar a capacidade de deteção de intrusões do IDS. Estes ataques são realizados utilizando clientes CoAP.

Os resultados dos testes para os registos de tráfego CoAP anómalos para ataques em rajada a URI válido são apresentados da seguinte forma. Na tabela 5.8 são apresentados os resultados dos testes referentes à primeira fase e, na tabela 5.9 são apresentados os resultados dos testes referentes à segunda fase, que contem o *dataset* final apenas com tráfego anómalo.

Tabela 5.8: Resultado da deteção de fluxos com ataques em rajada a URI válido por mensagem

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
GET	100	50	3	47	0	6%	94%	94%
POST	100	50	5	45	0	10%	90%	90%

Tabela 5.9: Resultado da deteção de fluxos com ataques em rajada a URI válido

Tipo de anomalia CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
Rajada URI Válido	17953	313	31	282	0	8%	92%	92%

Resultado do *dataset* para registos de fluxos de tráfego CoAP

Considerando agora o *dataset* com os registos de fluxo de tráfego CoAP capturados através do protótipo do cenário de testes, é pretendido validar a capacidade de deteção de intrusões do IDS.

Na tabela 5.10 são apresentados os resultados com do *dataset* para CoAP.

Tabela 5.10: Resultado do *dataset* para a deteção de registos de fluxos de tráfego CoAP

Dataset	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
Dataset	98322	15835	2352	13483	0	0.2%	99.8%	99.8%

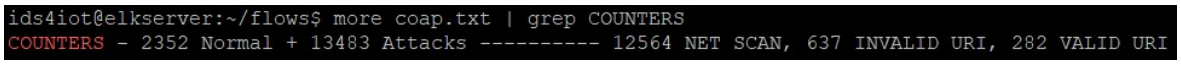


Figura 5.4: Resultado obtido à análise do *dataset* pela aplicação IDS

Após os resultados obtidos através dos testes com fluxos selecionados e o teste ao *dataset*, utilizando a aplicação IDS de forma a validar a performance do mesmo, podemos afirmar que foram positivas.

Numa primeira fase era pretendido validar e diferenciar se um registo de fluxo de tráfego CoAP seria um fluxo normal e anómalo. Numa segunda fase era pretendido validar e diferenciar um registo de tráfego CoAP anómalo por tipo de ataque, nomeadamente ataque de *Net Scan*, ataque a URI inválido e por último ataque a URI válido em rajada.

Em relação aos registos de fluxos de tráfego normais, numa primeira etapa, foram utilizados fluxos selecionado com poucos registos de fluxos de tráfego CoAP normal para a validação das especificações. Estes fluxos tiveram como base a utilização dos métodos GET e POST. Posteriormente, numa segunda etapa, foi utilizado um *dataset* que resume o tráfego gerado pelo protótipo após sete dias em operação, sem qualquer introdução de tráfego anómalo, de forma a garantir a total veracidade dos registos de fluxos de tráfego normais.

Através dos resultados obtidos através da aplicação IDS, podemos constatar que na tabela 5.2 o resultado para a classificação dos registos de fluxos de tráfego CoAP normais é de 100%.

Considerando os registos de fluxos de tráfego anómalos, numa primeira etapa, foram utilizados fluxos selecionado com poucos registos de fluxos de tráfego CoAP anómalo para a validação das especificações, consoante o tipo de teste. Os registos de fluxos selecionados tiveram como base a utilização das ferramentas HPING e NMAP para os ataques de *Net Scan* e os métodos GET e POST para os ataques a URI inválido ou válido. Posteriormente, numa segunda etapa, foi utilizado um *dataset* que resume o tráfego gerado pelo protótipo, apenas com registos de fluxos de ataques, de forma a garantir a total veracidade dos registos de fluxos de tráfego anómalos.

Através dos resultados obtidos através da aplicação IDS, pode-se constatar que na tabela 5.4 o resultado para a classificação dos registos de fluxos de tráfego CoAP anómalo para o ataque *Net Scan* é de 100%. Pode-se igualmente constatar que na tabela 5.7 o resultado para a classificação dos registos de fluxos de tráfego CoAP

anómalo para o ataque a URI Inválido incluindo o ataque em rajada é de 100%

Através dos resultados obtidos através da aplicação IDS, pode-se constatar que na tabela 5.9 o resultado para a classificação dos registos de fluxos de tráfego CoAP anómalo para o ataque em Rajada a URI Válido é de 92%. Para este tipo de ataque cerca de 8% dos registos de fluxos considerados como ataque são classificados como registos de fluxos normais. Este resultado advém por alguns registos de fluxos de tráfego anómalos serem mal classificados.

Por último, através dos resultados obtidos através da aplicação IDS ao *dataset*, pode-se constatar que na tabela 5.10 e figura 5.4 o resultado para a classificação dos registos de fluxos de tráfego CoAP normais e anómalos por tipo de ataque é de 99,8%. Com este resultado pode-se afirmar que as especificações cumprem de uma forma satisfatória a classificação dos registos de fluxos de tráfego CoAP.

5.3 Testes e resultados obtidos para o protocolo aplicacional CoAP sobre DTLS

Através do resultado deste teste pretende-se validar o mecanismo de análise aos registos de fluxos de tráfego IP, de forma a analisar a *performance* das especificações apresentadas no apresentadas no subcapítulo 4.3, referentes ao protocolo aplicacional CoAPS.

Para desenvolver este teste, foram utilizados os registos de fluxo de tráfego IP CoAPS armazenados no módulo central do IDS. De salientar que estes registos de fluxos de tráfego IP, antes de serem armazenados no módulo central, passaram por um processo de troca de mensagens entre clientes, dispositivo ataque e servidores CoAPS. Posteriormente foram exportados para o módulo central, onde foram armazenados.

Como abordado no plano de testes, este conjunto de testes está dividido em duas situações: i) análise dos registos de fluxos de tráfego IP de comunicações IoT CoAPS normais e ii) análise dos registos de fluxos de tráfego IP de comunicações IoT CoAPS anómalos, seguido de respetiva classificação por ataque.

Os resultados são apresentados através da utilização de tabelas para cada uma das fases. Nelas estão discriminados o método utilizado na troca de mensagens, o número

de registos de fluxos de tráfego que compõem o *dataset* utilizado, o número de registos de fluxos de tráfego referentes ao protocolo aplicacional, o número de registo de fluxos de tráfego que foram classificados como normais (RFN), número de registo de fluxos de tráfego que foram classificados como anómalos (RFA), taxa de falsos positivos (FP), taxa de falsos negativos (FN), taxa de verdadeiros positivos (TP) e por último a taxa de deteção (TD) para o teste efetuado.

5.3.1 Resultado aos fluxos de tráfego normais com sessão DTLS estabelecida

De forma a avaliar e validar os registos de fluxos de tráfego IP para comunicações normais, foram efetuados testes em duas fases diferentes separados pelos métodos GET e POST. Numa primeira fase são selecionados alguns registos de fluxos normais de forma a conter um menor número de amostras para validação das especificações. Numa segunda fase, os resultados são utilizados todos os registos de fluxos de tráfego normais, independentemente do seu método. Estes registos de fluxos de tráfego encontram-se no *dataset*, anteriormente abordado. Estes ataques são realizados utilizando clientes CoAPS.

Os resultados dos testes para os registos de tráfego CoAPS normais são apresentados da seguinte forma. Na tabela 5.1 são apresentados os resultados dos testes referente à primeira fase e na tabela 5.2 são apresentados os resultados dos testes referentes à segunda fase que contem o *dataset* apenas com tráfego normal.

Tabela 5.11: Resultado da deteção de fluxos normais CoAP com sessão DTLS estabelecida por mensagem

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
GET	100	100	100	0	0	0	100%	100%
POST	100	100	100	0	0	0	100%	100%

Tabela 5.12: Resultado da deteção de fluxos normais CoAP com sessão DTLS estabelecida

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
GET/POST	17260	8824	8824	0	0	0	100%	100%

5.3.2 Resultado aos testes de sessões DTLS válidas

Considerando agora apenas os registos de fluxo de tráfego onde é estabelecido uma sessão DTLS válida, é pretendido validar a capacidade de deteção de intrusões do IDS. Estas testes são realizados utilizando clientes CoAPS.

Neste ponto, a forma de validação terá de ser ligeiramente diferente pois não se trata de mensagens CoAP mas sim o estabelecimento de sessões DTLS válidas, sendo um percurso obrigatório para o canal garantir uma camada de segurança adicional na troca de mensagens CoAP. Assim, igualmente por meio de tabelas, serão demonstrados os resultados obtidos, sendo as tabelas discriminadas por tipo de tentativa de sessões, número de registos de fluxos de tráfego que existem no *dataset* utilizado, número de tentativas, número de tentativas com sucesso (S), número de tentativas sem sucesso (IN) e por último, a taxa de deteção (TD) para o teste efetuado.

Os resultados dos testes para os registos de tráfego DTLS são apresentados da seguinte forma. Na tabela 5.13 são apresentados os resultados dos testes com o *dataset* tendo em conta o estabelecimento de sessões DTLS válidas.

Tabela 5.13: Resultado da deteção de fluxos com sessões DTLS válidas

Sessões DTLS	Número de fluxos do dataset	Número de tentativas	S	IN	TD
Tentativas estabelecidas	17260	8	8	0	100%

5.3.3 Resultado aos fluxos de tráfego anómalos

De forma a avaliar e validar os registos de fluxos de tráfego IP para comunicações anómalas, foram efetuados testes em duas fases diferentes separados pelos métodos GET e POST para os ataques a URI e pelas ferramentas HPING e NMAP para os ataques de *Net Scan*. Numa primeira fase são selecionados alguns registos de fluxos normais de forma a conter um menor número de amostras para validação das especificações. Numa segunda fase, os resultados são utilizados todos os registos de fluxos de tráfego anómalos, independentemente do seu método ou ferramenta. Estes registos de fluxos de tráfego encontram-se no *dataset*, anteriormente abordado. Os ataques são categorizados em ataques de *Net Scan*, utilizando HPING e NMAP, ataques a URI inválidos e válidos, incluindo os ataques em rajada.

Resultado aos ataques de *Net Scan*

Considerando agora apenas os registos de fluxo de tráfego anómalos com ataques *Net Scan*, é pretendido validar a capacidade de deteção de intrusões do IDS. Estes ataques são realizados utilizando o dispositivo atacante.

Os resultados dos testes para os registos de tráfego CoAPS anómalos para ataques *Net Scan* são apresentados da seguinte forma. Na tabela 5.14 são apresentados os resultados dos testes referentes à primeira fase e, na tabela 5.15 são apresentados os resultados dos testes referentes à segunda fase, que contem o *dataset* final apenas com tráfego anómalo.

Tabela 5.14: Resultado da deteção de fluxos com ataques *Net Scan* por mensagem

Tipo de ferramenta	Número de fluxos do dataset	Número de fluxos CoAPS	RFN	RFA	FP	FN	VP	TD
NMAP	100	100	0	100	0	0	100%	100%
HPING	100	100	0	100	0	0	100%	100%

Tabela 5.15: Resultado da deteção de fluxos com ataques *Net Scan*

Tipo de anomalia CoAP	Número de fluxos do dataset	Número de fluxos CoAPS	RFN	RFA	FP	FN	VP	TD
Net Scan	28636	6210	0	6210	0	0	100%	100%

Resultado aos ataques a URI inválido com sessão DTLS estabelecida

Considerando agora apenas os registos de fluxo de tráfego anómalos com ataques a URI inválidos com sessão estabelecida, é pretendido validar a capacidade de deteção de intrusões do IDS. Estes ataques são realizados utilizando clientes CoAPS.

Os resultados dos testes para os registos de tráfego CoAPS anómalos para ataques a URI inválidos com sessão estabelecida são apresentados da seguinte forma. Na tabela 5.16 são apresentados os resultados dos testes referentes à primeira fase e, na tabela 5.17 são apresentados os resultados dos testes referentes à segunda fase, que contem o *dataset* apenas com tráfego anómalo.

Tabela 5.16: Resultado da deteção de fluxos com ataques a URI inválido por mensagem

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
GET	25	25	0	25	0	0	100%	100%
POST	25	25	0	25	0	0	100%	100%

Tabela 5.17: Resultado da deteção de fluxos com ataques a URI inválido

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
URI Inválido	1629	99	00	99	0	0	100%	100%

Resultado aos ataques em rajada a URI válido e inválidos sem sessão DTLS estabelecida

Neste ponto é onde reside a grande diferença na troca de mensagens com e sem sessão DTLS estabelecida. Após uma análise cuidada chegou-se à conclusão que não é ainda possível compreender se um ataque em rajada poderá ser a um URI válido ou inválido sem sessão DTLS estabelecida. Contudo, é possível compreender que um ataque ocorre independentemente do tipo de ataque e método utilizado. Desta forma, o administrador não saberá exatamente qual o ataque que está a ocorrer mas saberá que o mesmo estará a ocorrer, sabendo igualmente qual a sua origem do ataque.

Tendo este ponto em consideração, apenas serão considerados os registos de fluxo de tráfego anómalos, com ataques em rajada a URI inválidos e válidos sem sessão DTLS estabelecida, no qual é pretendido validar a capacidade de deteção de intrusões do IDS. Estes ataques são realizados utilizando clientes CoAPS. Estes ataques são realizados utilizando clientes CoAPS.

Os resultados dos testes para os registos de tráfego CoAPS anómalos para ataques a URI inválidos e válidos sem sessão estabelecida são apresentados da seguinte forma. Na tabela 5.18 são apresentados os resultados dos testes referentes à primeira fase e, na tabela 5.19 são apresentados os resultados dos testes referentes à segunda fase, que contem o *dataset* apenas com tráfego anómalo.

Tabela 5.18: Resultado da deteção de fluxos com ataques em rajada a URI

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
Rajada URI	10	10	0	10	0	0	100%	100%

Tabela 5.19: Resultado da deteção de fluxos com ataques em rajada a URI

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
Rajada URI	32844	226	0	226	0	0	100%	100%

Resultado do *dataset* para registos de fluxos de tráfego CoAP sobre DTLS

Considerando agora o *dataset* com os registos de fluxo de tráfego CoAP capturados através do protótipo do cenário de testes, é pretendido validar a capacidade de deteção de intrusões do IDS.

Na tabela 5.20 são apresentados os resultados com do *dataset* para CoAPS.

Tabela 5.20: Resultado do *dataset* para a deteção de registos de fluxos de tráfego CoAP

Tipo de mensagem CoAP	Número de fluxos do dataset	Número de fluxos CoAP	RFN	RFA	FP	FN	VP	TD
Dataset	98322	15359	8824	6535	0	0%	100%	100%

```
ids4iot@elkserver:~/flows$ more coaps.txt | grep COUNTERS
COUNTERS - 8824 Normal + 8 DTLS Válido + 6535 Attacks ---------- 6210 NET SCAN, 99 INVALID URI, 226 BRUST
```

Figura 5.5: Resultado obtido à análise do *dataset* pela aplicação IDS

Após os resultados obtidos através dos testes com fluxos selecionados e o teste ao *dataset*, utilizando a aplicação IDS de forma a validar a performance do mesmo, podemos afirmar que foram positivas.

Numa primeira fase era pretendido validar e diferenciar se um registo de fluxo de tráfego CoAP sobre DTLS seria um fluxo normal ou anómalo. Numa segunda fase era pretendido validar e diferenciar um registo de tráfego CoAP anómalo por tipo de ataque, nomeadamente ataque de *Net Scan*, ataque a URI e por último ataque em rajada a URI.

Em relação aos registos de fluxos de tráfego normais, numa primeira etapa, foram utilizados fluxos selecionado com poucos registos de fluxos de tráfego CoAPS normal para a validação das especificações. Estes fluxos tiveram como base a utilização dos métodos GET e POST. Posteriormente, numa segunda etapa, foi utilizado um *dataset* que resume o tráfego gerado pelo protótipo após sete dias em operação, sem qualquer

introdução de tráfego anómalo, de forma a garantir a total veracidade dos registos de fluxos de tráfego normais.

Através dos resultados obtidos através da aplicação IDS, podemos constatar que na tabela 5.12 o resultado para a classificação dos registos de fluxos de tráfego CoAPS normal é de 100%.

Adicionalmente, por se tratar de uma sessão DTLS, é necessário em primeiro lugar estabelecer uma sessão válida. Assim, através da tabela 5.13 pode-se constatar que o resultado para as sessões válidas é de 100%.

Considerando os registos de fluxos de tráfego anómalos, numa primeira etapa, foram utilizados fluxos selecionado com poucos registos de fluxos de tráfego CoAPS anómalo para a validação das especificações, consoante o tipo de teste. Os registos de fluxos selecionados tiveram como base a utilização das ferramentas HPING e NMAP para os ataques de *Net Scan* e os métodos GET e POST para os ataques a URI inválido ou válido com e sem sessão DTLS estabelecida. Posteriormente, numa segunda etapa, foi utilizado um *dataset* que resume o tráfego gerado pelo protótipo, apenas com registos de fluxos de ataques, de forma a garantir a total veracidade dos registos de fluxos de tráfego anómalos.

Através dos resultados obtidos através da aplicação IDS, pode-se constatar que na tabela 5.15 o resultado para a classificação dos registos de fluxos de tráfego CoAP anómalo para o ataque *Net Scan* é de 100%. Pode-se igualmente constatar que na tabela 5.17 o resultado para a classificação dos registos de fluxos de tráfego CoAP anómalo para o ataque a URI Inválido é de 100%.

Através dos resultados obtidos através da aplicação IDS, pode-se constatar que na tabela 5.19 o resultado para a classificação dos registos de fluxos de tráfego CoAP anómalo para o ataque em Rajada a URI é de 100%.

Por último, através dos resultados obtidos através da aplicação IDS ao *dataset*, pode-se constatar que na tabela 5.20 e figura 5.5 o resultado para a classificação dos registos de fluxos de tráfego CoAP sobre DTLS normais e anómalos por tipo de ataque é de 100%. Com este resultado pode-se afirmar que as especificações cumprem de uma forma satisfatória a classificação dos registos de fluxos de tráfego CoAP sobre DTLS.

5.4 Síntese

Neste capítulo apresentaram-se os resultados aos testes mencionados no plano de testes.

Pretendeu-se validar a aplicação IDS quanto à sua performance de classificar os registos de fluxos de tráfego, tendo consigo alcançar este objetivo com sucesso.

Pretendeu-se também validar a *performance* para a deteção de intrusões tendo em consideração as especificações declaras para CoAP. Desta forma, para os testes com registos de fluxos de tráfego normais, os resultados foram de 100% na taxa de deteção. Em relação aos registos de fluxos de tráfego anómalos, classificando-os de seguida, os resultados foram de 99,8%.

Por fim, pretendeu-se também validar a *performance* para a deteção de intrusões tendo em consideração as especificações declaradas para CoAPS. Desta forma, para os testes com registos de fluxos de tráfego normais, os resultados foram de 100% na taxa de deteção. Em relação aos registos de fluxos de tráfego anómalos e posterior classificação por ataque, os resultados foram de 100%. Contudo, é importante realçar que para os ataques a URI inválido e válido sem sessão DTLS não é possível compreender o tipo de ataque mas é possível perceber que está a decorrer um ataque, pelo que será sempre gerado um alerta de intrusão.

Capítulo 6

Conclusões

O trabalho efetuado permitiu conhecer características de novos equipamentos, novos protocolos, tecnologias específicas, soluções apresentadas por outros investigadores contribuindo na resolução de problemas que advém com estes tipos de sistemas, bem como um novo conceito, que à partida era tido como um conceito complexo.

A pesquisa realizada de forma minuciosa, direcionou este projeto para a criação de especificações para a *framework* proposta, visando especificar e conceber um sistema de intrusões em sistemas IoT a fim de detetar, em tempo útil, intrusões oriundas da rede interna ou da rede externa, centrando-se sobretudo nas intrusões que afetam as camadas de rede e aplicação da arquitetura IoT.

Como ponto de partida, é abordado no capítulo 2 uma visão geral sobre o conceito IoT e seus constituintes, abordando igualmente o protocolo aplicacional CoAP e onde este se insere neste vasto conceito que é o IoT. Ainda neste capítulo são abordados os respetivos desafios de segurança e alguns mecanismos que visam mitigar os problemas aderentes nestes tipos de sistemas através de contra medidas convencionais.

É também abordado no capítulo 3, uma visão geral sobre o conceito de IDS e as suas caraterísticas sobre classificação de um IDS. É também realizado uma síntese sobre as soluções propostas por outros investigadores visando a utilização de um sistema de deteção de intrusões para os sistemas de IoT.

Após a apreciação da pesquisa realizada sobre dos trabalhos apresentados pelos vários investigadores relativamente aos sistemas de deteção de intrusões para sistemas de IoT, é apresentado no capítulo 4, a *framework* proposta, o seu modo de funcionamento e seus constituintes. Ainda neste capítulo é abordado como foram realizadas as especificações para os registos de fluxos de tráfego normais e anómalos, visando classificar os

mesmos por tipos de ataque. Foi também realizado um plano de testes visando avaliar as especificações criadas.

Por último, no capítulo 5 são realizados e discutidos os testes efetuados para a validação da aplicação IDS, como os testes para validar as especificações previamente declaradas.

Os objetivos propostos na realização deste projeto foram todos alcançados. A análise das principais vulnerabilidades permitiu compreender o quão vulnerável se encontra um dispositivo IoT, devido às suas limitações, tanto a nível computacional como energético e à grande heterogeneidade existente neste tipo de sistemas, o que dificulta a implementação de mecanismos de segurança.

Através da análise às soluções propostas pela comunidade científica permitiu compreender que ainda não existe uma solução ideal para um sistema de deteção de intrusões para sistemas IoT.

As especificações criadas com o auxilio da análise aos registos de fluxos de tráfego gerados, relevaram-se satisfatórios, atingindo uma taxa de deteção para os registos de fluxos de tráfego CoAP de 99,8% e para os registos de fluxos de tráfego CoAP sobre DTLS de 100%. Com estas taxas de deteção pode-se afirmar que o resultado é satisfatório.

O facto para a taxa de deteção para os registos de fluxos de tráfego CoAP serem de 99,8% reside no facto dos IE disponíveis ainda não permitirem ter uma maior abrangência sobre as informações dos registos de fluxos de tráfego CoAP. Deste modo, para ataques em rajada aliados à agregação dos pacotes em fluxos, omitindo de alguma forma informações, não é possível detetar todos os fluxos respetivos a ataques, embora o resultado seja satisfatório.

Em relação à taxa de deteção para os registos de fluxos de tráfego CoAP sobre DTLS ser de 100%, algumas informações não são possíveis ser detetadas. Mais uma vez, com a limitação da informação que os IE podem fornecer aliada à agregação dos pacotes em fluxos, não é possível compreender num ataque de rajada a um URI, se este é um ataque para um URI válido ou inválido, tal como não é possível compreender o método utilizado. Apesar disso e, reforçando a ideia, não é possível detetar o tipo de ataque mas é possível detetar que um ataque está a ocorrer, lançando um alerta para o administrador. Em suma, o administrador não sabe o tipo de ataque mas sabe que está a ocorrer um ataque e sabe qual a sua origem e destino, podendo atuar em

conformidade.

Por fim, considerando os resultados obtidos através da aplicação do IDS, poder-se-á concluir que a *framework* proposta é viável para um sistema de deteção de intrusões para sistemas IoT baseado em especificações.

6.1 Principais Contribuições

Tomando em consideração os objetivos iniciais e os resultados obtidos, as principais contribuições deste projeto foram:

- Levantamento sobre as soluções propostas de IDS para IoT e suas limitações;

- Criação de especificações para registos de fluxos de tráfego CoAP com e sem DTLS;

- Contribuição para a comunidade através da publicação de um artigo, *CoAP flow signatures for the IoT* [82];

- Criação de um ou dois artigos sobre as especificações e resultados obtidos com a realização deste projeto.

6.2 Tópicos para Trabalho Futuro

Tendo em conta as conclusões obtidas das análises aos registos de fluxos de tráfego CoAP, perspetiva-se que no futuro haja muito trabalho a fazer nesta área, pois ainda existem mais ataques que podem ser testados, existe mais um tipo de mensagem e resposta em CoAP ou até a utilização de uma outra abordagem.

Para a realização deste projeto foram assumidas algumas opções que conduziram à escolha do protocolo aplicacional CoAP como protocolo a utilizar nesta solução. Como trabalho futuro a inserção dos restantes métodos que fazem parte do protocolo CoAP aliado a ataques em simultâneo do mesmo atacante, isto é, o atacante realizar dois ou mais ataques ao mesmo tempo. A razão prende-se pelo facto de tentar compreender se através dos fluxos se consegue obter bons resultados.

A realização de testes incluindo as mensagens *Non-comfirmable* e/ou o modo de resposta *Separate Response*. A razão prende-se pelo facto de tentar compreender o comportamento dos registos de fluxos de tráfego e respetivos ataques, ao utilizar mensagens *Non-comfirmable*. Em relação ao modo de resposta *Separate Response*, prende-se pelo facto de compreender como será o seu comportamento em fluxos.

Realização de testes com o método Observe, incluindo ataques ao mesmo. A razão prende-se com o facto de este método ao ser utilizado, o servidor apenas notifica o cliente se houver alterações no pedido efetuado pelo mesmo. Em caso de ataque o cliente pode receber informações erradas que podem ter consequências no sistema IoT.

A inclusão de outros ataques é igualmente um fator adicional, pelo facto de compreender se através destas especificações é possível detetar mais ataques.

A adição de contadores nos ataques DoS, de forma a auxiliar na deteção para os registos de fluxos de tráfego que foram classificados como normais.

Devido a algumas limitações ainda existentes nos IE, seria produtivo tentar uma abordagem com outros IE ou com outras formas de obter informação acerca dos registos de fluxos de tráego IP;

A escolha para uma deteção híbrida, tirando partido de uma deteção baseada em assinaturas e especificações, poderá aumentar a taxa de detenção e o número de ataques detetados. Consequentemente ter-se-á de ter em consideração as limitações existentes nos sistemas IoT.

Bibliografia

[1] Leonel Santos. Sistema de deteção de intrusões para a internet das coisas. pages 1–174, 2020.

[2] Zach Shelby, Klaus Hartke, Carsten Bormann, Brian Frank, et al. The constrained application protocol (coap). 2014.

[3] Mohammed El-Haii, Maroun Chamoun, Ahmad Fadlallah, and Ahmed Serhrouchni. Analysis of cryptographic algorithms on iot hardware platforms. In *2018 2nd Cyber Security in Networking Conference (CSNet)*, pages 1–5. IEEE, 2018.

[4] Bruno Bogaz Zarpelão, Rodrigo Sanches Miani, Cláudio Toshio Kawakani, and Sean Carlisto de Alvarenga. A survey of intrusion detection in internet of things. *Journal of Network and Computer Applications*, 84:25–37, 2017.

[5] Christian Cervantes, Diego Poplade, Michele Nogueira, and Aldri Santos. Detection of sinkhole attacks for supporting secure routing on 6lowpan for internet of things. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 606–611. IEEE, 2015.

[6] Prabhakaran Kasinathan, Claudio Pastrone, Maurizio A Spirito, and Mark Vinkovits. Denial-of-service detection in 6lowpan based internet of things. In *2013 IEEE 9th international conference on wireless and mobile computing, networking and communications (WiMob)*, pages 600–607. IEEE, 2013.

[7] Pavan Pongle and Gurunath Chavan. Real time intrusion and wormhole attack detection in internet of things. *International Journal of Computer Applications*, 121(9), 2015.

[8] Daniele Midi, Antonino Rullo, Anand Mudgerikar, and Elisa Bertino. Kalis—a system for knowledge-driven adaptable intrusion detection for the internet of things. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 656–666. IEEE, 2017.

[9] Eung Jun Cho, Jin Ho Kim, and Choong Seon Hong. Attack model and detection scheme for botnet on 6lowpan. In *Asia-Pacific Network Operations and Management Symposium*, pages 515–518. Springer, 2009.

[10] Douglas H Summerville, Kenneth M Zach, and Yu Chen. Ultra-lightweight deep packet anomaly detection for internet of things devices. In *2015 IEEE 34th international performance computing and communications conference (IPCCC)*, pages 1–8. IEEE, 2015.

[11] João P Amaral, Luís M Oliveira, Joel JPC Rodrigues, Guangjie Han, and Lei Shu. Policy and network-based intrusion detection system for ipv6-enabled wireless sensor networks. In *2014 IEEE International Conference on Communications (ICC)*, pages 1796–1801. IEEE, 2014.

[12] Sudip Misra, P Venkata Krishna, Harshit Agarwal, Antriksh Saxena, and Mohammad S Obaidat. A learning automata based solution for preventing distributed denial of service in internet of things. In *2011 international conference on internet of things and 4th international conference on cyber, physical and social computing*, pages 114–122. IEEE, 2011.

[13] Kevin Ashton et al. That 'internet of things' thing. *RFID journal*, 22(7):97–114, 2009.

[14] Harald Sundmaeker, Patrick Guillemin, Peter Friess, and Sylvie Woelfflé. Vision and challenges for realising the internet of things. *Cluster of European Research Projects on the Internet of Things, European Commision*, 3(3):34–36, 2010.

[15] J Buckley et al. The internet of things: from rfid to the next-generation pervasive networked systems, 2006.

[16] Gartner Research Vice President Mark Hung. "gartner insights on how to lead in a connected world". 2017.

[17] Gartner. "hype cycle special report for 2017", 2017. `https://www.gartner.com/en/documents/3768572`, último acesso em 2018-11-17.

[18] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[19] Rob van Kranenburg, Erin Anzelmo, Alessandro Bassi, Dan Caprio, Sean Dodson, and Matt Ratto. The internet of things. In *1st Berlin Symposium on Internet and Society: Exploring the Digital Future*, pages 25–27, 2011.

[20] Dhananjay Singh. "developing an architecture: Scalability, mobility, control, and isolation on future internet services". pages 22–25, 2013.

[21] in: Co-operation with the Working Group RFID of the ETP EPOSS Networked Enterprise & RFID INFSO & Nanosystems. "internet of things in 2020". page 27, 2008.

[22] Payam Barnaghi, Wei Wang, Cory Henson, and Kerry Taylor. Semantics for the internet of things: early progress and back to the future. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(1):1–21, 2012.

[23] Narayanan Raman. "how low-powered wi-fi sensors are the future of the iot", 2017. `https://www.imgtec.com/blog/how-low-powered-wi-fi-sensors-are-the-future-of-iot/`, último acesso em 2019-01-14.

[24] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[25] Erich Reisenhofer, M Bright, J Mangione, P Mosch, J Jeff Neafsey, and P Polak. Simplifying iot: Connecting, commissioning, and controlling with near field communication (nfc). *Retrieved January*, 13:2018, 2016.

[26] A uniform resource name (urn) namespace for the near field communication (nfc) forum - *https://tools.ietf.org/html/rfc4729*.

[27] J Nieminen, T Savolainen, M Isomaki, B Patil, Z Shelby, and C Gomez. Rfc 7668-ipv6 over bluetooth low energy. *IETF: Fremont, CA, USA*, 2015.

[28] Srdjan Krčo, Boris Pokrić, and Francois Carrez. Designing iot architecture (s): A european perspective. In *2014 IEEE World Forum on Internet of Things (WF-IoT)*, pages 79–84. IEEE, 2014.

[29] Rafiullah Khan, Sarmad Ullah Khan, Rifaqat Zaheer, and Shahid Khan. Future internet: the internet of things architecture, possible applications and key challenges. In *2012 10th international conference on frontiers of information technology*, pages 257–260. IEEE, 2012.

[30] Moumena A Chaqfeh and Nader Mohamed. Challenges in middleware solutions for the internet of things. In *2012 international conference on collaboration technologies and systems (CTS)*, pages 21–26. IEEE, 2012.

[31] Lu Tan and Neng Wang. Future internet: The internet of things. In *2010 3rd international conference on advanced computer theory and engineering (ICACTE)*, volume 5, pages V5–376. IEEE, 2010.

[32] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, 17(4):2347–2376, 2015.

[33] Monowar Hasan, Ekram Hossain, and Dusit Niyato. Random access for machine-to-machine communication in lte-advanced networks: Issues and approaches. *IEEE communications Magazine*, 51(6):86–93, 2013.

[34] Erick C Jones and Christopher A Chung. *RFID and Auto-ID in Planning and Logistics: A Practical Guide for Military UID Applications*. CRC Press, 2016.

[35] Daniel Minoli. *Building the internet of things with IPv6 and MIPv6: the evolving world of M2m communications*. John Wiley & Sons, 2013.

[36] T. Kivinen and P. Kinney. Ieee 802.15.4 information element for the ietf. *IETF: Fremont, CA, USA*, 2017.

[37] Maria Rita Palattella, Nicola Accettura, Xavier Vilajosana, Thomas Watteyne, Luigi Alfredo Grieco, Gennaro Boggia, and Mischa Dohler. Standardized protocol stack for the internet of (important) things. *IEEE communications surveys & tutorials*, 15(3):1389–1406, 2012.

[38] Jonathan W Hui and David E Culler. Extending ip to low-power, wireless personal area networks. *IEEE Internet Computing*, 12(4):37–45, 2008.

[39] P Thubert and R Cragie. Ipv6 over low-power wireless personal area network (6lowpan) paging dispatch. *IETF, RFC 8025*, 2016.

[40] Tim Winter, Pascal Thubert, Anders Brandt, Jonathan W Hui, Richard Kelsey, Philip Levis, Kris Pister, Rene Struik, Jean-Philippe Vasseur, and Roger K Alexander. Rpl: Ipv6 routing protocol for low-power and lossy networks. *rfc*, 6550:1–157, 2012.

[41] K Lynn, S Cheshire, M Blanchet, and D Migault. Requirements for scalable dns-based service discovery (dns-sd)/multicast dns (mdns) extensions. *Internet Requests for Comments, RFC Editor*, 2015.

[42] A. Sullivan. Selecting labels for use with conventional dns and other resolution systems in dns-based service discovery. *Internet Requests for Comments, RFC Editor*, 2017.

[43] Roy T Fielding and Richard N Taylor. *Architectural styles and the design of network-based software architectures*, volume 7. University of California, Irvine Irvine, 2000.

[44] OASIS Standard. Oasis advanced message queuing protocol (amqp) version 1.0. *International Journal of Aerospace Engineering Hindawi www. hindawi. com*, 2018, 2012.

[45] Peter Saint-Andre et al. Extensible messaging and presence protocol (xmpp): Core. 2004.

[46] Carsten Bormann, Angelo P Castellani, and Zach Shelby. Coap: An application protocol for billions of tiny internet nodes. *IEEE Internet Computing*, 16(2):62–67, 2012.

[47] Eric Rescorla and Tim Dierks. The transport layer security (tls) protocol version 1.3. 2018.

[48] Eric Rescorla and Nagendra Modadugu. Datagram transport layer security version 1.2. Technical report, 2012.

[49] Karen Rose, Scott Eldridge, and Lyman Chapin. The internet of things: An overview understanding the issues and challenges of a more connected world. *The Internet Society (ISOC)*, 22, 2015.

[50] Yang Xiao, Hsiao-Hwa Chen, Bo Sun, Ruhai Wang, and Sakshi Sethi. Mac security and security overhead analysis in the ieee 802.15. 4 wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, 2006(1):093830, 2006.

[51] Jonathan Hui, Pascal Thubert, et al. Compression format for ipv6 datagrams over ieee 802.15. 4-based networks. 2011.

[52] Gabriel Montenegro, Nandakishore Kushalnagar, Jonathan Hui, David Culler, et al. Transmission of ipv6 packets over ieee 802.15. 4 networks. *Internet proposed standard RFC*, 4944:130, 2007.

[53] Anhtuan Le, Jonathan Loo, Aboubaker Lasebae, Mahdi Aiash, and Yuan Luo. 6lowpan: a study on qos security threats and countermeasures using intrusion detection system approach. *International Journal of Communication Systems*, 25(9):1189–1212, 2012.

[54] Ahmed Raoof, Ashraf Matrawy, and Chung-Horng Lung. Secure routing in iot: evaluation of rpl secure mode under attacks. *arXiv preprint arXiv:1905.10314*, 2019.

[55] Robert Shirey. Internet security glossary, 2000.

[56] Pranav M Pawar, Rasmus H Nielsen, Neeli R Prasad, Shingo Ohmori, Ramjee Prasad, et al. Behavioral modeling of wsn mac layer security attacks: A sequential uml approach. *Journal of Cyber Security and Mobility*, 1(1):65–82, 2012.

[57] Okan Can and Ozgur Koray Sahingoz. A survey of intrusion detection systems in wireless sensor networks. In *2015 6th international conference on modeling, simulation, and applied optimization (ICMSAO)*, pages 1–6. IEEE, 2015.

[58] Abdur Rahaman Sardar, Rashmi Ranjan Sahoo, Moutushi Singh, Souvik Sarkar, Jamuna Kanta Singh, and Koushik Majumder. Intelligent intrusion detection system in wireless sensor network. In *Proceedings of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA) 2014*, pages 707–712. Springer, 2015.

[59] Yulia Cherdantseva and Jeremy Hilton. A reference model of information assurance & security. In *2013 International Conference on Availability, Reliability and Security*, pages 546–555. IEEE, 2013.

[60] Olalekan Adeyinka. Internet attack methods and internet security technology. In *2008 Second Asia International Conference on Modelling & Simulation (AMS)*, pages 77–82. IEEE, 2008.

[61] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.

[62] James P Anderson. Computer security threat monitoring and surveillance. *Technical Report, James P. Anderson Company*, 1980.

[63] L Vokorokos and A Baláž. Host-based intrusion detection system. In *2010 IEEE 14th International Conference on Intelligent Engineering Systems*, pages 43–47. IEEE, 2010.

[64] Sanjay Sharma and RK Gupta. Intrusion detection system: A review. *International Journal of Security and Its Applications*, 9(5):69–76, 2015.

[65] Rick Hofstede, Pavel Čeleda, Brian Trammell, Idilio Drago, Ramin Sadre, Anna Sperotto, and Aiko Pras. Flow monitoring explained: From packet capture to data analysis with netflow and ipfix. *IEEE Communications Surveys & Tutorials*, 16(4):2037–2064, 2014.

[66] Leonel Santos, Carlos Rabadão, and Ramiro Gonçalves. Flow monitoring system for iot networks. In *World Conference on Information Systems and Technologies*, pages 420–430. Springer, 2019.

[67] IEs. "ip flow information export (ipfix) entities", 2007. `https://www.iana.org/assignments/ipfix/ipfix.xhtml`, último acesso em 2019-03-17.

[68] Ganesh Sadasivan, Nevil Brownlee, Benoit Claise, Juergen Quittek, et al. Architecture for ip flow information export. *draft-ietf-ipfix-architecture-12 (work in progress)*, 2006.

[69] Christopher M Inacio and Brian Trammell. Yaf: yet another flowmeter. In *Proceedings of LISA10: 24th Large Installation System Administration Conference*, page 107, 2010.

[70] Luca Deri and NETikos SpA. nprobe: an open source netflow probe for gigabit networks. In *TERENA Networking Conference*, pages 1–4, 2003.

[71] Carrie Gates, Michael P Collins, Michael Duggan, Andrew Kompanek, and Mark Thomas. More netflow tools for performance and security. In *LISA*, volume 4, pages 121–132, 2004.

[72] Peter Haag. Watch your flows with nfsen and nfdump. In *50th RIPE Meeting*, 2005.

[73] Luca Deri and Stefano Suin. Ntop: Beyond ping and traceroute. In *International Workshop on Distributed Systems: Operations and Management*, pages 271–283. Springer, 1999.

[74] Anna Sperotto, Gregor Schaffrath, Ramin Sadre, Cristian Morariu, Aiko Pras, and Burkhard Stiller. An overview of ip flow-based intrusion detection. *IEEE communications surveys & tutorials*, 12(3):343–356, 2010.

[75] Leonel Santos, Carlos Rabadao, and Ramiro Gonçalves. Intrusion detection systems in internet of things: A literature review. In *2018 13th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–7. IEEE, 2018.

[76] Shahid Raza, Linus Wallgren, and Thiemo Voigt. Svelte: Real-time intrusion detection in the internet of things. *Ad hoc networks*, 11(8):2661–2674, 2013.

[77] Ilona Murynets and Roger Piqueras Jover. Anomaly detection in cellular machine-to-machine communications. In *2013 IEEE International Conference on Communications (ICC)*, pages 2138–2143. IEEE, 2013.

[78] Linus Wallgren, Shahid Raza, and Thiemo Voigt. Routing attacks and countermeasures in the rpl-based internet of things. *International Journal of Distributed Sensor Networks*, 9(8):794326, 2013.

[79] Doohwan Oh, Deokho Kim, and Won Woo Ro. A malicious pattern detection engine for embedded security systems in the internet of things. *Sensors*, 14(12):24188–24211, 2014.

[80] M Surendar and A Umamakeswari. Indres: An intrusion detection and response system for internet of things with 6lowpan. In *2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 1903–1908. IEEE, 2016.

[81] Leonel Santos, Ramiro Gonçalves, and Carlos Rabadão. A novel intrusion detection system architecture for internet of things networks. In *ECCWS 2019 18th European Conference on Cyber Warfare and Security*, page 428. Academic Conferences and publishing limited, 2019.

[82] Luís Canuto, Leonel Santos, Leandro Vieira, Ramiro Gonçalves, and Carlos Rabadâo. Coap flow signatures for the iot. In *2019 14th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6. IEEE, 2019.