Internet Of Things

Architecture, Technologies and Applications

Francisco Torrinha[A91691] , João Novais[A91671] e Pedro Sequeira[A91660]

Universidade do Minho, R. da Universidade, 4710-057 Braga

Neste relatório são abordados uma série de temas relacionados com IoT, desde a recolha de dados a partir de sensores a protocolos e tecnologias implementadas para facilitar e otimizar a transmissão de informação entre máquinas conectadas a uma rede. Serão aprofundadas as arquiteturas de IoT nomeadamente as suas camadas, apresentar algumas tecnologias utilizadas em IoT nomeadamente *Big Data Analytics* e alguns casos práticos. Serão também referidos alguns protocolos dos quais IoT depende desde o nível físico ao nível de aplicações.

Keywords: IoT, Protócolo, Camadas.

Índice

1 Introdução

Nos dias que que correm, as IoT representam uma parte fundamental do funcionamento económico, financeiro e até social da nossa atualidade. No entanto, olhando para as tendências dos últimos tempos, desde a implementação das IoT, e possível prever que o seu desenvolvimento vai crescer e que cada vez mais os serviços de utilidade dos quais dependemos vão ficar conectados entre si.

Para permitir esta intercomunicação entre aparelhos e necessário que os dados recolhidos por diferentes sensores sejam interpretados de forma eficiente tendo em conta as suas limitações nomeadamente poder de processamento e armazenamento. Para alem das insuficiências acima referidas também e importante ter em conta que a transmissão dos dados recolhidos por sensores tem a possibilidade de ser corrompida ou até mesmo perdida, nomeadamente em sistemas de comunicação sem fios.

Após os dados capturados pelos sensores terem sido recebidos e processados devidamente, estes podem ser utilizados numa abundância de formas desde provocar acontecimentos físicos como por exemplo, ligar um determinado aparelho mecânico através de corrente elétrica, a acontecimentos virtuais como a comunicação com outro qualquer aparelho digital capaz de receber informação.

Para que houvesse uma forma uniformizada e congruente de comunicação entre dispositivos foram criadas várias arquiteturas e protocolos cada um com utilidades especificas dependendo dos pontos fracos e fortes que apresentam.

Todo este sistema de transmissão e receção de dados tem aplicações em diversas áreas das quais dependemos diariamente nomeadamente a área da saúde, como exemplo atual e relevante referir aplicações como *stayaway-covid* , na educação com implementação de aplicações como a *blackboard* e na restauração, nomeadamente com aplicações semelhantes a popular aplicação *uber-eats,* entre outras, fazendo assim com que a qualidade de vida individual e social melhor.

2 Arquitetura

2.1 Níveis de arquitetura

Existem várias arquiteturas cada uma com níveis número de camadas diferentes e com camadas mais ou menos especificos. A primeira arquitetura apresntada neste relatório é chamada de Three-layer. Esta arquitetura, tal como o nome indica tem 3 níveis: Perception Layer(Sensores que recolhem informação e que identificam outros objetos IoT no meio), Network Layer(Este nível conecta os vários objetos IoT a dispositivos com ligação á rede, servidores e outros objetos IoT e processar os dados recolhidos pelos sensores) e Aplication Layer(Este nível é responsabilizado por fornecer serviços específicos da aplicação IoT ao utilizador). Apesar de resumir o essaencial das IoT não é mais adequada para desenvolver estudos em relação as IoT, pois é uma arquitetura muito genérica. A arquitetura Five-Layer é mais detalhada e mais especifica dividindo-se em 5 camadas: Object layer(Camada com a msm função que a Perception Layer), Object Management Layer (Transfere os dados recolhidos na Object Layer para a Service Management Layer através de canais seguros. Os dados são transferidos através RFID, 3G, WiFi, Bluetooth Low Energy, LAN entre outras), Service Managment *Layer*(Esta layer armazena, analisa e processa os dados transferidos pela Object Managment Layer. Nesta camada é onde se aplica as tecnologias as databases, cloud computing e Big data analysis), Aplication Layer(A mesma camada utilizada na Three-Layer) e Business Layer(Gere todo o sistema IoT, desde as suas aplicações aos seus modelos economicos). Para além destas arquiteturas existem outras como arquiteturas SOA-based que possuem uma camada Service *Composition*(Comunica com outros objetos IoT para integrar serviços requeridos pelo o utilizador) [6,8].

2.2 Elementos das IoT

Para assegurar o funcionamento adequado da IoT e necessário decompô-la em elementos, cada um composto por várias tecnologias e protocolos específicos a tarefa

a executar. Os seguintes elementos serão abordados neste relatório, identificação, sensores, comunicação e computação.

Tal como o nome sugere o nome, o elemento de identificação tem como objetivo estabelecer um método de reconhecimento entre os vários objetos ligados a determinada rede. Cada objeto tem um atributo de nome e *address*.

Vários objetos na mesma rede podem possuir o mesmo nome. Em termos de *address* existem dois principais tipos, IPv4 e IPv6.

As *addresses* IPv4 consistem em *integers* 32 bit tipicamente divididas por um ponto em cada byte sendo que cada divisão e representada no formato decimal. Ainda hoje são bastante utilizadas apesar de haver apenas um número muito limitado de *addresses* possíveis de atribuir a dispositivos comparativamente ao número de conexões que é possível gerar atualmente.

E importante também saber que dispositivos fora de uma rede tem de ter um IP único globalmente visível e dispositivos que são apenas visíveis dentro da rede tem que ter um IP local único.

Os endereços IPv4 podem ser incluídos em uma de cinco classes. A classe A cujo primeiro byte e o prefixo e os outros atribuem um valor único ao *host*. A classe B em que os dois primeiros bytes representam o prefixo e os últimos dois o valor único do *host*. A classe C tem 3 bytes como prefixo e o último como número do *host*. A classe D e reservada para *multi-catsing*. A classe E serve para futuras aplicações.

O formato IPv6 foi implementado devido a falta de flexibilidade de IPv4 face ao número atual de possíveis dispositivos conectados a uma rede. E, portanto, representado por uma 128bit *integer* em lugar da relativamente pequena 32bit *integer* utilizada em IPv4.

IPv6 suporta também uma variedade de tipos de *address*. *Unicast*, que especifica um identificador para um serie de interfaces que recebem pacotes. Este tipo de *address* tipicamente representa a maior parte de elementos de uma rede. *Multicast* e identificada por começar com 0xFF e apenas suporta tráfego de entrada e saída do *host*. *Anycast* e muito parecido com *unicast*, com a exceção de que suporta endereço de rota de *subnet*.

O elemento sensores e responsável por recolher dados sobre o ambiente que rodeiam o objeto de rede, alguns exemplos de sensores são câmaras, sensores atmosféricos e pressão e giroscópios.

O elemento de comunicação desempenha um papel fundamental da IoT pois permite a transmissão de dados entre elementos de uma rede. Existe uma grande variedade de tecnologias de comunicação que são utilizadas conforme a situação e tipo de dados presente.

O elemento de computação e responsável pelo processamento e validação dos dados recolhidos por sensores. Para envergar neste esforço computacional podem ser utilizadas uma variedade de hardware desde máquinas compactas, mas com especificações fracas tal como o *raspberry* pi, a server clusters capazes de processar uma quantidade relativamente elevada de dados simultaneamente.

Para complementar o hardware e necessário ter um sistema operativo capaz, sendo maior parte das vezes utilizado um *UNIX* ou *UNIX-Like based* os tal como Linux (*Arch*, *Gentoo*, *Debian*, *Open BSD* etc) [1,2].

2.3 Protocolos das IoT

Application Protocols

FTP significa File *Transfer Protocol* e tal como o nome indica, este protocolo permite a transferência de dados entre máquinas ligadas a mesma rede. A operação de FTP pode decorrer em uma de dois modos. No modo passivo o cliente conectasse a um servidor FTP a partir de um *port* aleatório, tipicamente, ao *port* 21 e o servidor responde com um outro *port* aleatório que foi disponibilizado para tranferencia de dados. No modo ativo um cliente conectasse a um servidor FTP a partir de um *port* aleatório, tipicamente, ao *port* 21 e envia o número do *port* ao qual o servidor se devera conectar para transferir os dados. FTP tem três modos de transferência de dados. O *stream mode* que permite com que os dados sejam envidos de forma continua e deixam todo o processamento de informação para FTP. O *block* insere cada linha de dados num bloco juntamente com o *header* e o tamanho do *block* em bytes. O modo de compressão e essencialmente igual ao modo, apenas passando os dados por um algoritmo de compressão [4,5].

HTTP e um protocolo *request-response* em que um cliente requer algo de um servidor, e de seguida esse mesmo servidor retorna ao cliente uma resposta, a maior parte das vezes estas respostas são *html*, *css*, etc. Devido a dependência de preservação de dados que HTTP herda, e mais comumente utilizado em cima de FTP, apesar de haver adaptações para UDP. Em termos de estrutura HTTP *packets* podem ser decompostos em duas partes o *header* e a secção de dados. O *header* normalmente contem uma serie de informações como a data de envio, o tipo de *client*, etc., mas a mais importante e a *request line*. Dentro da *request line* encontra-se o *request type*, o file *path*, e a versão HTTP. Existem vários tipos de *request* em HTTP, sendo três dos mais importantes GET, POST e DEL. GET informa o servidor que devera retornar ao cliente a informação contida no *path* especificado no *request*. POST informa o server que devera colocar no *path* especificado pelo *request* os conteúdos da secção de dados do *packet*. DEL informa o servidor que devera apagar a informação disponível no *path* especificado.[3]

Service Discovery Protocols

O protocolo DNS está acessível na camada TCP/IP, é um sistema de nomear para dispositivos conectados a uma rede e também é descentralizado e hierárquico. Este protocolo traduz os nomes de domínios para *ip's* necessários para os utilizadores na rede poderem aceder e localizar serviços computacionais e dispositivos com os protocolos de rede subjacentes. O DNS tem a responsabilidade de atribuir e mapear nomes de domínios para os recursos da internet designando Servidores de Nome Autoritários para cada. A internet mantém dois principais espaços de nome (*namespaces*), a hierarquia de nome de domínio e espaço de endereço para protocolo

de internet (internet *Protocol* (*ip*)). O Sistema de Nomes de Domínios mantém a hierarquia de nomes de domínios e providencia serviços de tradução entre si mesmo e o espaço de endereço. Os servidores de nomes da internet e o protocolo de comunicação implementam o Sistema de Nomes de Domínio. O Servidor de Nomes da DNS é um servidor que guarda os registos DNS de um domínio [1]. De uma maneira mais ilustrativa, quando um utilizador tenta pesquisar por um site, por exemplo, se o endereço já não é conhecido ao seu sistema operativo este envia um "*query*" ao Servidor de Resolver Nomes, por sua vez este servidor irá contactar os Servidores de Raiz de Nomes se tem o endereço do site em questão na sua cache, se não o Servidor de Resolver Nomes (SRN) vai contactar os Servidores de Nível Alto de Domínios, no caso de este não ter o endereço, também, o SRN irá, finalmente, contactar os Servidores Autoritário de Nomes que irá responder ao SRN o endereço do site em causa, finalmente o SRN comunica ao sistema operativo o *ip* e assim já é possível estabelecer uma ligação [13,14].

Infrastructer Protocol

Nesta secção será aprofundado o funcionamento de vários protocolos de infraestruturas proeminentes nas IoT.
Protocolos de infraestruturas são usados para estabelecer comunicações adjacentes requisitadas pelas aplicações de IoT.
Um dos protocolos mais influentes é o RPL (*Routing Protocol* for *Low Power and Lossy* Networks). Este protocolo é uma implementação do protocolo ROLL (*Routing over low-power and lossy links*) baseada em IPv6. Este protocolo foi criado para auxiliar requisitos mínimos de *routing* através da construção de topologias mais robustas sobre ligações com tendências de perda de informação.
O protocolo RPL é representado através do diagrama de nodos DODAG (*Destination Oriented Directed Acyclic Graph*). O DODAG possui apenas um nodo raiz, cada nodo esta ciente da existência do seu nodo pai, mas não possui informação sobre os seus nodos filhos. Para manter um nível de performance e eficiência alto o RPL armazena pelo menos o caminho mais rápido de cada nodo até a raiz.
Para manter a topologia de *routing* quando a informação é transferida e atualizada entre os nodos o RPL utiliza 4 tipos de mensagens de controlo: DODAG *Information Object*(DIO), guarda o nível do nodo, determina a distância do nodo á raiz e escolhe o caminho para o nodo pai mais eficiente, *Distination Advertisemenjet Object*(DAO) o RPL fornece trafego dos nodos com nível maiores para os de menor nível e vice versa para suportar as mensagens DAO que atualizam informação do nodo pai escolhido, DODDAG *Information Solicitation*(DIS) que são usadas para adquirir mensagens DIO provenientes de nodos adjacentes e DAO-*Acknoledgment*(DAO-Ack) uma mensagem de resposta as mensagens DAO enviadas pela raiz da DODAG ou nodos pais DAO.
A DODAG começa a formar-se quando o nodo raiz envia a sua localização através de mensagens DIO a todos os níveis de *Low-power Lossy* Network (LLN). Em cada nível routers registam um caminho para o nodo pai e caminhos para os nodos filhos. Esses

routers propagam as suas mensagens DIO até a DODAG ser formada. Depois desta ser construída cada router obtém um caminho para um nodo pai da forma mais eficiente tornando-se este caminho o *default* para chegar ao nodo raiz suportando assim tráfego dos nodos filhos para a raiz. Para suportar tráfego dos nodos pais para nodos filhos são utilizadas mensagens DAO.
Routers RPL tem dois modos de operação Non *storing*(As mensagens movem-se para os níveis mais baixos baseando-se em *source routing* IP) ou *Storing*(as mensagens movem-se para níveis mais baixos baseando-se nos endereços IPv6). A implementação deste protocolo é usualmente feita em ambientes *Contiki* entre outros [7,8].

O último protocolo de infraestrutura que será apresentado é o protocolo IEEE 802.15.4 foi criado para especificar uma subcamada para *Medium* Access *Control*(MAC) e uma camada física para *low-rate* wireless *pivate area* networks(LR-WPAN). Devido ao seu baixo consumo de energia, baixa taxa de carregamento de dados, baixo custo e alta transferência de mensagens este protocolo é muito utilizado nas IoT. Este protocolo fornece uma comunicação viável, um bom funcionamento em diversas plataformas e um nível alto de segurança, encriptação e serviços de autenticação. O IEEE 802.15.4 possui 3 canais de frequência de banda e o utiliza o método *direct sequence* spread *spectrum* (DSSS). Baseado nos canais de frequência a camada física transmite e recebe dados em 3 taxas de carregamento diferentes: 250 *kbps* a 2.4 GHz,40 *kbps* t 915 MHz, e 20 *kbps* a 868 MHz. Maiores frequências e largura de banda proporciona maior transferência de mensagens baixa latência(*ping*). Menores frequências cobre maiores distancias. O IEEE 802.15.4 suporta 2 tipos de nodos: Full Function Devices(FFDs) que são utilizados como coordenadores de *personal area* networks(*PANs*) ou funcionar como nodos normais. Um coordenador é responsavel pela criação, controlo e manutenção de uma rede. *Reduced Function Devices*(RFD) são nodos simples e com recursos limitados [8].

3 Técnologias

3.1 Big Data Analytics

Big Data significa equipar grandes quantidades de objetos físicos como humanos, animais, plantas, televisões, computadores, etc. com sensores conectando-os á internet. *A Big Data* precisa de um armazenamento eficiente e inteligente. Obviamente, os dispositivos conectados
necessitam de mecanismos para guardar, processar e recuperar. Mas a *Big Data*, como o nome implica, é tão enorme que excede a capacidade de ambientes hardware e ferramentas de software comuns para processar a informação num período aceitável. A imergência e o desenvolvimento de computação em nuvem é definida pela NIST como um modelo de acesso, para rede em procura por consumidores. Serviços da nuvem

permitem indivíduos e empresas usarem componentes de software e hardware remoto de terceiros [1]. Computação em nuvem permite a investigadores e negócios usarem e manterem muito recursos remotamente, confiante a um preço reduzindo. Os IoT empregam um enorme número de dispositivos embutidos, como sensores e atuadores que geram *Big Data* que por sua vez requer uma computação complexa para extrair conhecimento [2]. Por isso, o armazenamento e os recursos de computação da nuvem apresentam a melhor escolha para os IoT para guardar e processar a *Big Data* [8,12].

4 Referências

[1]-https://medium.com/datadriveninvestor/4-stages-of-iot-architecture-explained-in-simple-words-b2ea8b4f777f

[2]-https://www.hindawi.com/journals/jece/2017/9324035/

[3]-https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

[4]-https://en.wikipedia.org/wiki/File_Transfer_Protocol

[5]-https://www.jscape.com/blog/bid/80512/active-v-s-passive-ftp-simplified

[6]-https://www.researchgate.net/publication/312957467_Internet_of_Things_Architectures_Protocols_and_Applications

[7]-https://en.wikipedia.org/wiki/RPL_(IPv6_Routing_Protocol_for_LLNs)

[8]-Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications

[9]-Internet of Things Standards: Who Stands Out from the Crowd?

[10] – V. Gypta and jayaraghvandran "IoT Protocols War and the Way Forward", 28th Int'I Conf. VLSI Design, Invited Talk, 2015, pp 28.

[11] – March 2015- IoT market analysis: Sizing the opportunity, Knud Lasse Lueth

[12] R. Bryant, R. H. Katz, and E. D. Lazowska, "Big-data computing: Creating revolutionary breakthroughs in commerce, science, and society," Comput. Commun. Consortium (CCC),

Washington, DC, USA, 2008.

[13] - Wikipédia - https://en.wikipedia.org/wiki/Domain_Name_System

[14] – vídeo de DNS Made Easy Videos -

https://www.youtube.com/watch?v=72snZctFFtA&ab_channel=DNSMadeEasyVideos